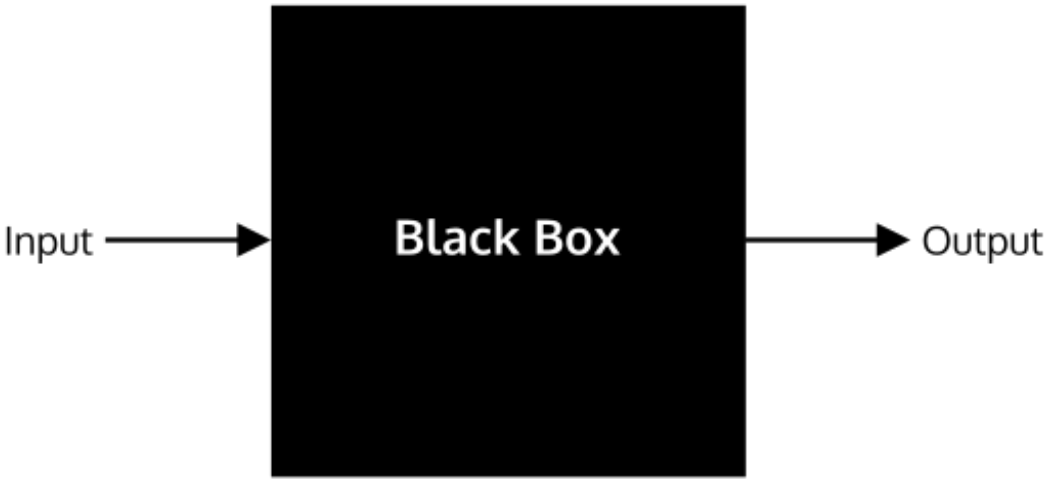


Interpretability and Explainability in Machine Learning

JOÃO PAULO CARRILHO CUNHA

(Licenciado)

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática e Multimédia

Orientador: Doutor Artur Ferreira

Júri:

Presidente: Doutor Pedro Jorge
Vogais: Doutor Artur Ferreira
Doutor Paulo Trigo

December 2024

Interpretability and Explainability in Machine Learning

JOÃO PAULO CARRILHO CUNHA

(Licenciado)

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática e Multimédia

Orientador: Doutor Artur Ferreira, DEETC/ISEL

Júri:

Presidente:	Doutor Pedro Jorge, DEETC/ISEL
Vogais:	Doutor Artur Ferreira, DEETC/ISEL
	Doutor Paulo Trigo, DEETC/ISEL

December 2024

Acknowledgements

I would like to express my deep gratitude to all the people and organizations who made this work possible.

First and foremost, I would like to thank my supervisor, Artur Jorge Ferreira, for his support and guidance throughout this process. His careful guidance and deep knowledge were instrumental to the success of this thesis. I would also like to thank ISEL for providing me with the opportunity to conduct this study and for all the support provided during the research period.

I thank my colleagues and friends who accompanied me on this journey and provided me with support, encouragement, and motivation to overcome the challenges that arose along the way.

Last but not least, I would like to thank my family, especially my parents Florbela Madalena Alves Carrilho and António José Dias Cunha, for their encouragement. Their support was essential for me to reach this milestone.

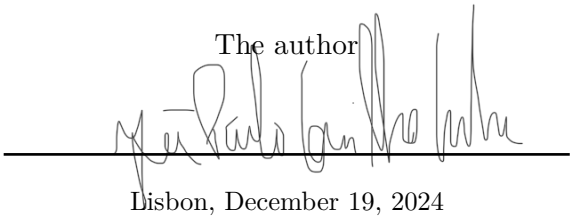
To all those mentioned and to all those who contributed in some way to the completion of this work, my sincere thanks. Your support and contributions were essential to this success.

This research was supported by Instituto Politécnico de Lisboa (IPL) under Grant `IPL/IDI&CA2024/ML4EP_ISEL`.

Statement of integrity

I declare that this **dissertation**t is the result of my personal and independent research. Its content is original, and all sources listed in the bibliographic references were consulted and are duly mentioned in the text. I further declare that all scientific and technical references relevant to the development of the work are duly cited and included in the bibliographic references.

The author



Lisbon, December 19, 2024

Abstract

As machine learning models become increasingly complex, the need for interpretability and explainability has grown significantly. This thesis investigates the use of machine learning and explainability techniques, namely DT (Decision Trees), SHAP (SHapley Additive exPlanations), EBM (Explainable Boosting Machines), and LIME (Local Interpretable Model-agnostic Explanations), to identify and analyze the most influential features in predictive models. On our study, we consider tabular and image data.

Using the Iris and Adult datasets for tabular data, we apply these algorithms to discern where models derive their predictive power and which features contribute most significantly to outcomes.

For image data, we address a multiclass classification problem using the Cats and Dogs dataset, employing saliency maps to pinpoint the most relevant pixels influencing model accuracy. Our findings demonstrate that these interpretability techniques effectively unveil the underlying mechanisms of machine learning models, enhancing our understanding and trust in their predictive capabilities.

Keywords: Decision Trees, EBM, Explainability, Feature Importance, Interpretability, LIME, Machine Learning, Saliency Maps, SHAP

Resumo

À medida que os modelos de aprendizagem automática se tornam cada vez mais complexos, a necessidade de interpretabilidade e explicabilidade tem crescido significativamente. Esta tese investiga o uso de técnicas de aprendizagem automática e explicabilidade, nomeadamente árvores de decisão, SHAP (SHapley Additive exPlanations), EBM (Explainable Boosting Machines) e LIME (Local Interpretable Model-agnostic Explanations), para identificar e analisar as características mais influentes em modelos preditivos. No nosso estudo, consideramos dados tabulares e de imagem.

Para dados de imagem, abordamos um problema de classificação multiclasse utilizando o conjunto de dados Cats and Dogs, utilizando mapas de saliência para identificar os píxeis mais relevantes que influenciam a precisão do modelo. As nossas descobertas demonstram que estas técnicas de interpretabilidade revelam eficazmente os mecanismos subjacentes dos modelos de aprendizagem automática, aumentando a nossa compreensão e confiança nas suas capacidades preditivas.

Palavras-chave: Aprendizagem Automática, Árvores de Decisão, EBM, Explicabilidade, Importância das Características, Interpretabilidade, LIME, Mapas de Saliência, SHAP

Contents

List of Figures

Acronyms

AI	Artificial Intelligence 5, 14, 20, 21, 24, 25, 26
ASM	Angular Second Moment 17
CAM	Class Activation Mapping 16
CAV	Concept Activation Vector 17
CM	Confusion Matrix 51, 91
CNN	Convolutional Neural Network 9, 21, 22, 23, 24, 25, 29, 30, 33, 34
CT	Computed Tomography 23
DNN	Deep Neural Network 8, 12, 14, 16, 17
DT	Decision Tree 2, 11, 12, 13, 25, 29, 30, 31, 32, 35, 36, 42, 51, 53, 54, 55, 65, 66, 67, 85, 86, 93
EBM	Explainable Boosting Machine 2, 11, 14, 15, 25, 29, 33, 35, 40, 48, 51, 53, 60, 61, 62, 63, 64, 65, 66, 77, 78, 79, 85, 86, 93
GANs	Generative Adversarial Networks 5
GRU	Gated Recurrent Unit 9
LIME	Local Interpretable Model-agnostic Explanations 2, 5, 11, 13, 14, 25, 33, 34, 92, 93, 94
LSTM	Long Short-Term Memory 9
ML	Machine Learning 1, 2, 5, 11, 12, 13, 14, 15, 20, 24, 25, 26, 27, 28, 29, 94
MRI	Magnetic Resonance Imaging 23
NLP	Natural Language Processing 23
NN	Neural Network 5, 6, 7, 8, 19
ReLU	Rectified Linear Unit 6, 21
ResNet	Residual Network 17, 23, 24, 29, 30, 94
RNN	Recurrent Neural Network 9

SGD Stochastic Gradient Descent 22

SHAP SHapley Additive exPlanations 2, 5, 11, 12, 13, 25, 29, 31, 32, 33, 35, 36, 37, 38, 39, 40, 43, 44, 45, 46, 47, 48, 51, 53, 55, 56, 57, 58, 59, 60, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 85, 86, 93

SSD Single Shot MultiBox Detector 23

TCAV Testing with Concept Activation Vectors 11, 17

VGG Visual Geometry Group 29

XAI Explainable Artificial Intelligence 5, 20

XGBoost Extreme Gradient Boosting 31, 32, 33

YOLO You Only Look Once 23

1 Introduction

1.1 Context and Problem Framework

Machine Learning (ML) [1] is a branch of Artificial Inteligence [27]. ML has served as a tool for large-scale data interpretation and analysis. Supervised ML techiques are organized into two methods, namely classification and regression [12], under which exciting insights can be drawn and effective predictions can be made. The use of explainability techiques and ML algorithms in problem domains, such as tabular classification and data analysis to arrive at interpretable and explainable models assisting human decision-making, is the focus of this thesis.

1.2 Motivation and Objectives of the Work

Classification evaluation is one of the primary tasks in ML. We should be confident with our models, and hence, we should compare and evaluate their performance using appropriate metrics. That would allow us to tell which models are most accurate and reliable on the dataset for the task at hand, by comparing different classifiers.

In most practical scenarios, however, accurate predictions are not good enough. Numerous customers would like to obtain interpretive and explainable models that provide insight into the algorithms' decision-making process. Especially in sensitive domains, such as the Health Business, the importance of understanding what factors and features drive the model's predictions directly impacts human lives.

To satisfy this need, this thesis advocates the integration of mechanisms that provide interpretability and explainability into the ML model, which will explain the relationship between input features, model decisions, and output predictions. Explanation techniques help in visualizing the model decision-making process and therefore give insight into how the model processes the data and comes up with an outcome.

The methodology proposed will be further systematic, consisting of a number of successive stages. First, identification and preprocessing of relevant datasets involve ensuring that quality and relevance to specific domains of application are of interest. Then, it applies ML algorithms, trains classifiers on the dataset, and obtains output predictions.

Finally, interpretable and explainable techniques shall be taken up for analyzing the output prediction so as to go in-depth into the decision-making process of the model.

Of these, the focus of this thesis, in particular, is on the application of these methods to tabular classification and data analysis problems. We will be applying ML algorithms and interpretable models to garner meaningful insights for human decision-makers in these domains.

This research work in summary aims to develop an approach to the techniques of ML with a focus on interpretability and explainability. Providing human-interpretable models we gain trust and confidence on the algorithms decisions.

1.3 Organization of the Document

This thesis addresses the pressing need for interpretability in machine learning models as they become increasingly complex and opaque. By employing interpretability algorithms such as Decision Tree (DT), SHapley Additive exPlanations (SHAP), and Explainable Boosting Machine (EBM), we explore how key features in predictive models can be highlighted to provide insights into model decision-making. Additionally, methods like Local Interpretable Model-agnostic Explanations (LIME) and saliency maps are used for image data to identify crucial pixels that impact classification results, making the models' behavior more comprehensible. Here's an outline of how this thesis is structured:

1. **Introduction**: The first chapter introduces the motivation for this research, focusing on the necessity of model interpretability in modern ML models. Here, we present the objectives of this study and an overview of the interpretability techniques and datasets used, setting the stage for the proposed solution.

2. **Explainability in Machine Learning**: This chapter reviews existing literature on interpretability in machine learning, providing a summary of various methods and algorithms that make ML models more understandable. The chapter compares different approaches to interpretability, including decision trees, SHAP, LIME, and saliency maps, contextualizing their use in past research and highlighting gaps this study aims to address.

3. **Proposed Solution**: This chapter details the methodology and experimental setup, describing how interpretability algorithms were applied across both synthetic and real-world datasets. Techniques used, such as DT, SHAP, and EBM for tabular data, and LIME and saliency maps for image data, are introduced, along with an explanation of their implementation. This chapter provides a roadmap of the models and tools employed.

4. **Synthetic Datasets**: This chapter introduces the synthetic datasets, detailing their structure, features, and relevance for evaluating the interpretability techniques. Here, we outline how each interpretability algorithm interacted with the synthetic features, shedding light on their effectiveness in identifying key features.

5. **Real-world Datasets**: This chapter covers the real-world datasets used in this study, namely the Iris and Adult datasets. The structure and features of each dataset are described, followed by an analysis of the interpretability techniques applied.

6. **Cats and Dogs dataset**: This chapter discusses the use of the ResNet architecture for image classification on the Cats and Dogs dataset. Techniques like LIME and saliency maps are applied to visualize the important regions in images, showing how CNNs focus on features such as facial structures and fur patterns. The interpretability techniques help elucidate the CNN decision paths, making the classification process more transparent.

7. **Conclusions**: The final chapter provides an overview of the findings, summarizing how interpretability techniques like DTs, SHAP, EBM, LIME, and saliency maps contribute to making ML models more transparent and trustworthy. Additionally, suggestions for future research are presented, such as refining these methods for more complex datasets, with an emphasis on advancing interpretable and ethical AI applications.

2 Explainability in Machine Learning

2.1 Machine Learning and Deep Learning

Deep learning, a subset of ML, has gained prominence due to its success in handling complex tasks. It utilizes artificial Neural Network (NN) with multiple layers to model intricate relationships in large datasets. This chapter explores the fundamentals of deep learning, emphasizing the importance of data and the growing need for interpretability and explainability. We also examine how ML techniques like classification and regression provide a foundation for making deep learning models more interpretable and comprehensible.

Classification assigns predefined labels to input data by identifying patterns, making it useful in applications like image recognition and medical diagnostics. Regression, on the other hand, predicts continuous values, such as house prices or stock trends. These techniques enhance the versatility and predictive power of deep learning models across various domains.

Incorporating these techniques into deep learning not only improves accuracy but also helps to make results more interpretable. Visual tools like bar charts, scatter plots, and decision boundaries help explain relationships and patterns identified by models, making the results more accessible to users. These visualizations are essential in improving understanding and trust in model predictions.

Recent advances in deep learning extend beyond classification and regression, with models like Generative Adversarial Networks (GANs) and reinforcement learning excelling in areas such as image generation and autonomous systems. However, these complex models pose additional challenges for interpretability, leading to the need to develop methods for explaining their behavior.

As deep learning evolves, the demand for Explainable Artificial Intelligence (XAI) [7] has grown, particularly in high-stakes areas like healthcare and finance. Tools such as SHAP (SHapley Additive exPlanations) and LIME offer insights into how models make decisions, helping users understand individual predictions. These methods highlight the importance of specific features in shaping model outputs, making Artificial Intelligence (AI) systems more transparent and trustworthy.

2.2 Neural Network

NN [20] are computational models inspired by the human brain's intricate network of interconnected neurons. Each neuron in a neural network processes information and makes computations. These networks are organized into layers, each consisting of numerous interconnected neurons. Information flows through the network, undergoing transformations at each layer. NN are adept at learning complex patterns and relationships from data, making them adequate tools for various tasks such as image and speech recognition, natural language processing, among others.

At the core of a neural network lies the neuron, a fundamental building block. The artificial neuron mimics the basic functionality of a biological neuron. It receives input signals from other neurons through dendrites, processes this information in the cell body (soma), and, based on certain activation rules, generates an output signal sent through the axon. The strength of connections (synapses) between neurons, akin to weights in an artificial neural network, determines the influence of one neuron's output on another, as illustrated in Figure 2.1.

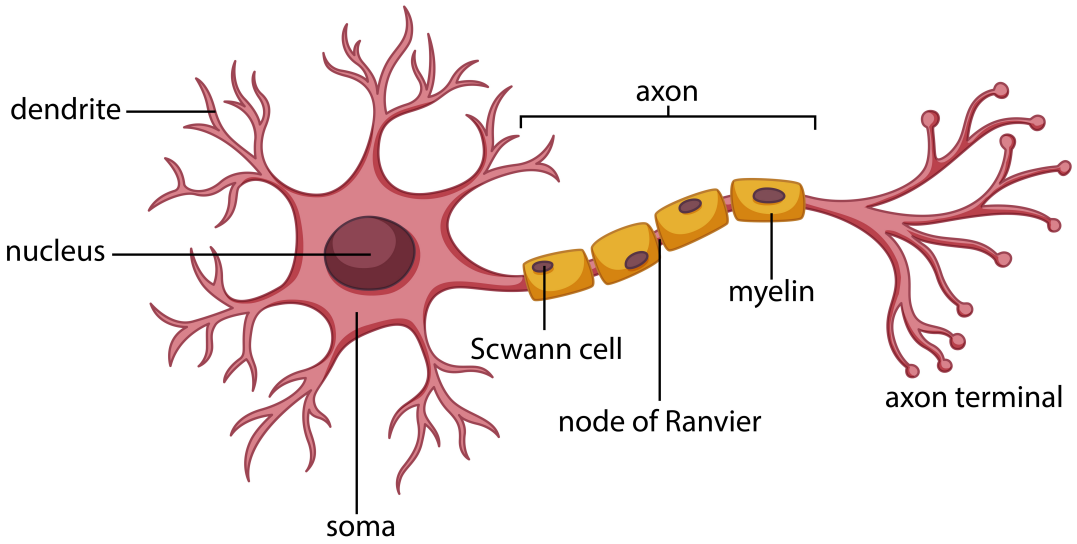


Figure 2.1: *Neuron Anatomy*

2.2.1 Activation Functions

Activation functions are crucial components of each neuron in a neural network. They introduce non-linearity into the neuron's output, allowing the network to learn complex patterns. Common activation functions include the sigmoid function, which squashes values into a range between 0 and 1, and the Rectified Linear Unit (ReLU), which outputs the input if it's positive and zero otherwise. Activation functions enable NN to model a wide range of relationships and make them more expressive and adaptable to various data types and tasks.

Figure 2.2 illustrates the structure of a neuron, which includes various components such as inputs, weights, a transfer function (summation), an activation function, and an output. Each input x_i is multiplied by a weight w_{ij}, and the weighted inputs are summed together

to form the net input (net_j). The activation function φ then processes this net input, resulting in the final output o_j.

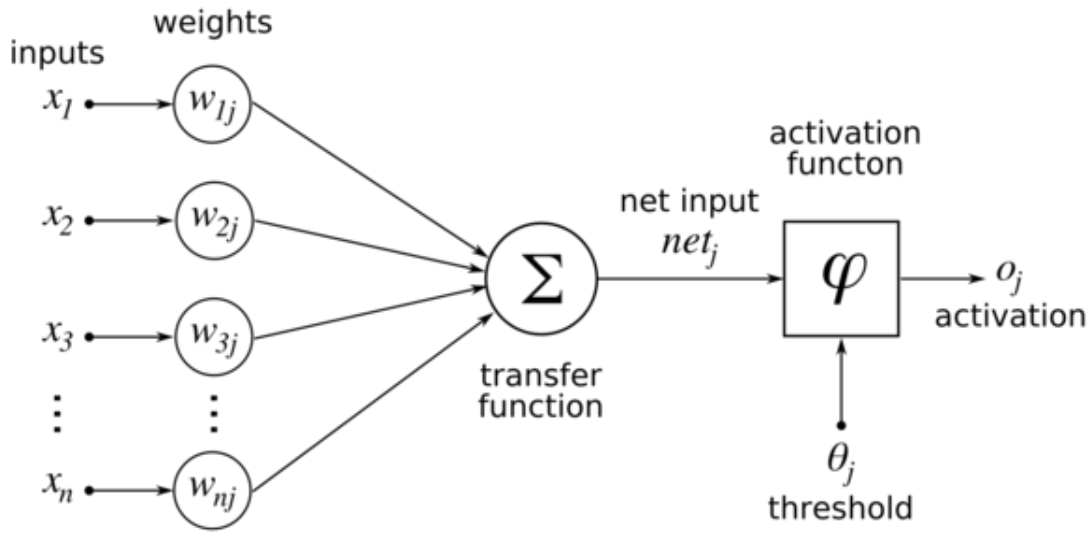


Figure 2.2: *Structure of a single neuron in a neural network*

2.2.2 Feedforward Neural Networks

A feedforward neural network [8] is the foundational architecture in deep learning. Information flows only in one direction, from the input layer through the hidden layers (if present) to the output layer. Each neuron in one layer is connected to every neuron in the subsequent layer. The input data is processed layer by layer, with each neuron's output serving as the input to the neurons in the following layer. This sequential flow allows the network to make predictions or classifications based on the input data.

2.2.3 Backpropagation Algorithm

The backpropagation algorithm [2] is a fundamental method for training NN. During the training process, the network makes predictions, and the resulting errors between these predictions and the actual target values are computed. Backpropagation involves calculating these errors layer by layer in reverse order, starting from the output layer and moving backward to the input layer. The algorithm then adjusts the model's parameters (weights and biases) iteratively to minimize these errors. This iterative process enhances the network's predictive accuracy, allowing the NN to approximate a given function.

2.2.4 Types of Neural Layers

NN consist of different types of layers, each serving a specific purpose in information processing. As shown in Figure 2.3, the input layer receives the initial data, the hidden layers (if present) process and transform this data through various computations, and the output layer produces the final predictions or classifications. Hidden layers allow the network to learn complex features and patterns, enabling it to make accurate predictions based on the provided input.

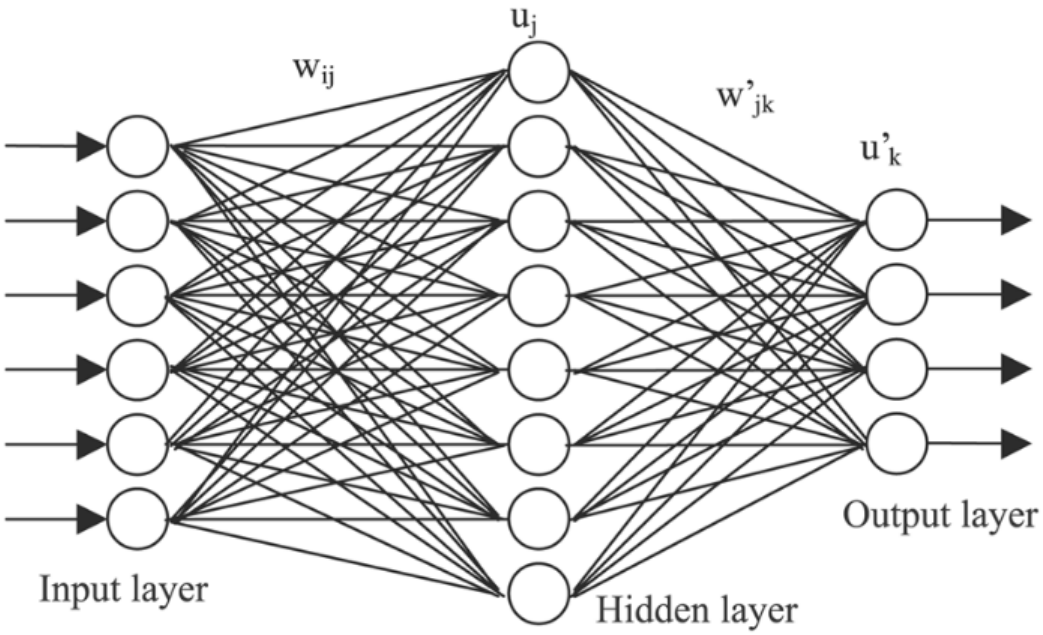


Figure 2.3: *Multi-layer neural networks (multi-layer perceptron - MLP)*

2.2.5 Deep Neural Networks

Deep Neural Network (DNN) [6] extend the concept of NN to include multiple hidden layers between the input and output layers. As illustrated in Figure 2.4, these additional layers enable DNN to learn intricate hierarchies of features from the input data, making them capable of representing highly complex relationships. The depth of the network (meaning the number of hidden layers) enhances the network's ability to understand and learn intricate patterns, contributing to its effectiveness in solving complex problems across various domains.

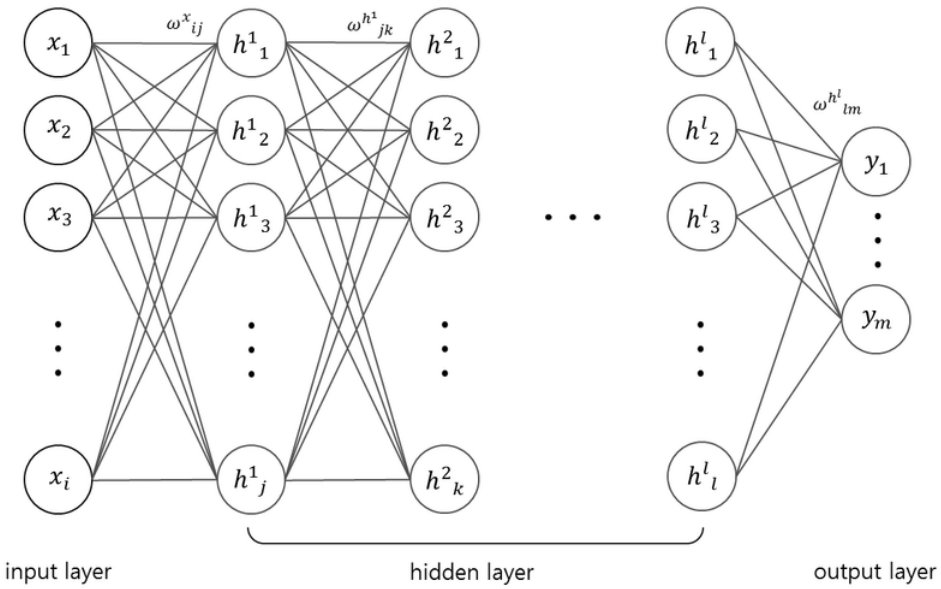


Figure 2.4: *Deep Neural Network Architecture Overview*

2.2.6 Convolutional Neural Networks

Convolutional Neural Network (CNN) [4] are specialized neural network architectures designed for processing grid-like data, such as images and videos. They consist of three main types of layers: convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply filters to the input data to extract features such as edges, shapes, and textures. Pooling layers downsample the spatial dimensions, reducing computational complexity. Fully connected layers perform classification based on the extracted features. CNN use weight sharing and spatial hierarchies to efficiently capture patterns, making them highly effective in image recognition, object detection, and other computer vision tasks, as illustrated in Figure 2.5.

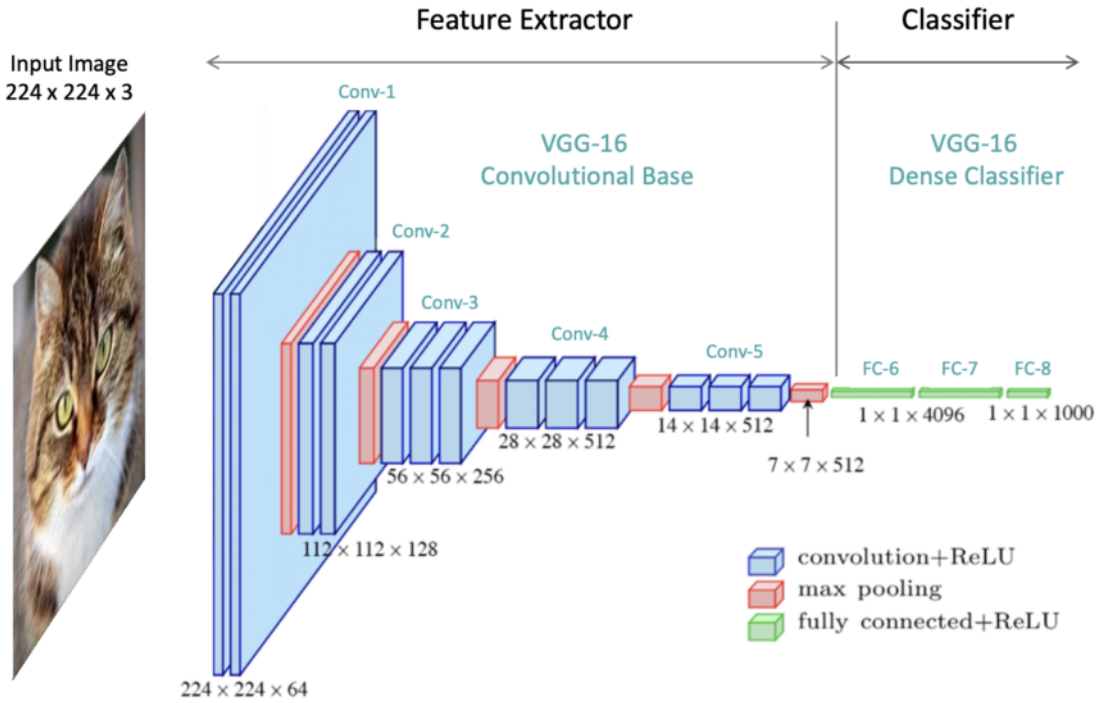


Figure 2.5: *Structure of a Convolutional Neural Network (CNN)*

2.2.7 Recurrent Neural Networks

Recurrent Neural Network (RNN) [22] are a type of neural network well-suited for processing sequential or time-series data. Unlike feedforward networks, RNN possess loops, allowing information to be passed from one step of the sequence to the next. This looped structure enables RNN to maintain memory of previous inputs, making them ideal for tasks such as natural language processing, speech recognition, and handwriting recognition. However, traditional RNN face challenges, namely the vanishing or exploding gradient problem, which led to the development of recent variants, such as Long Short-Term Memory (LSTM) [16] and Gated Recurrent Unit (GRU) [10], to address these issues and better capture long-term dependencies in sequences, as shown in Figure 2.6.

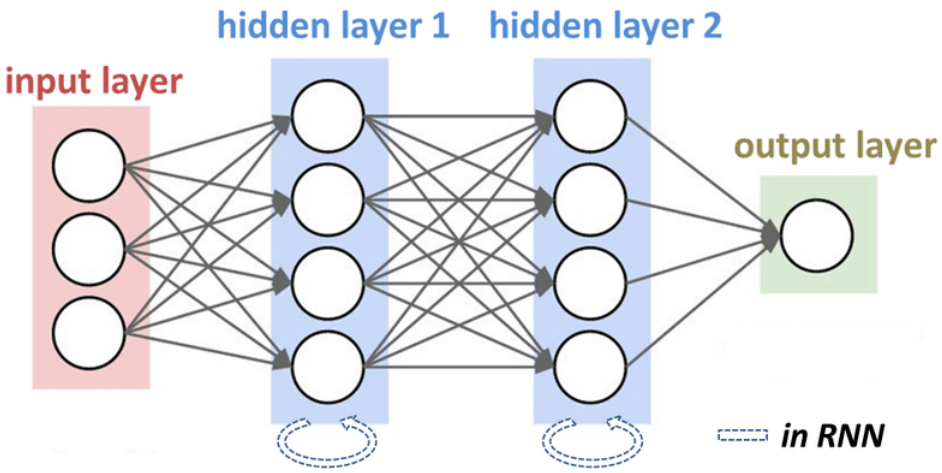


Figure 2.6: *Structure of a Recurrent Neural Network (RNN)*

2.2.8 Transfer Learning

Transfer learning [28] is a technique in deep learning that involves leveraging pre-trained neural network models for a new, related task. Instead of starting from scratch and training a model from the ground up, a pre-trained model is used as a starting point. The knowledge and features learned during the pre-training phase are transferred to the new task, either by fine-tuning the existing model's weights or extracting features from certain layers of the model and training a new classifier. Transfer learning is valuable when there is a scarcity of labeled data for the target task, allowing for significant reduction in training time and computational resources. It has shown remarkable success in various domains, including image classification, natural language processing, and speech recognition.

2.3 Interpretability and Explainability

Explainable artificial intelligence is an emerging field [7]. The distinction between interpretability and explainability is critical [26], as is the understanding of model interpretability techniques [25]. The success of deep learning relies not only on the amount of data available but also on its quality. While more data allows models to learn better and generalize more effectively, it's essential to consider two key factors: redundancy and relevance. Redundancy refers to similar or duplicate information within the dataset, which can cause overfitting and bias in predictions. Relevance, on the other hand, is about the significance of specific features or data points in contributing to accurate outcomes. By focusing on selecting relevant data, the performance and efficiency of deep learning models can be improved.

Interpretability and explainability are crucial for understanding and trusting deep learning models. Interpretability refers to how easily a human can grasp the decision-making process of a model. It is concerned with models being inherently understandable and transparent, where each part of the process is clear. Explainability goes a step further, offering tools to explain `why` and `how` a model makes its decisions, helping non-experts

to understand the model's behavior. Together, these concepts ensure that humans can trust and collaborate with ML models, making them essential for real-world applications in which models are deployed.

Interpreting and explaining deep learning models can be challenging due to their complexity and non-linearity. However, a range of methods has been developed to address this. Visualization techniques, for instance, highlight important features in the model's decisions. Other methods, such as model distillation and rule extraction, focus on simplifying complex models to make them easier to interpret while maintaining their accuracy. These techniques help to bridge the gap between complex ML models and human understanding.

Interpretability is crucial because it allows us to know how a model makes predictions. Understanding these processes can help improve the model, uncover biases, and better address its limitations. Explainability is equally important as it offers clear reasoning for the model's decisions, fostering trust and enabling communication of these decisions to others. This is especially relevant for regulatory and compliance purposes.

There are two primary approaches to enhancing interpretability and explainability in ML. The first is using simple, transparent models like DT or linear models. These models have clear and intuitive structures, being easy to interpret. The second approach involves using techniques that visualize and explain model predictions, such as feature importance plots, DT visualizations, and sensitivity analysis. These methods provide valuable insights into the factors driving model decisions and help identify potential biases or limitations in the model.

However, it's important to note that there is often a trade-off between interpretability and model performance. Complex, opaque models tend to deliver higher accuracy but are more difficult to explain. This trade-off requires careful balancing, especially in situations where interpretability is critical, such as in healthcare or finance.

In summary, interpretability and explainability are key aspects of ML models. Both can be achieved through transparent models or techniques that visualize and explain predictions. These factors are crucial for building trust and transparency in ML applications.

In addition to these methods, there are advanced tools and libraries that further improve interpretability. SHAP and LIME provide detailed explanations for individual predictions. EBM offer both accuracy and interpretability by using an additive modeling approach. Additionally, methods like saliency maps, Testing with Concept Activation Vectors (TCAV) [14], and distillation [15] help simplify complex models and offer explanations of predictions. Counterfactual explanations also allow for understanding how slight changes to input data could alter a model's prediction.

These advanced techniques, such as SHAP, LIME, and EBM, provide insights into both individual and global feature importance, allowing users to understand the driving factors behind model predictions. Saliency maps highlight the critical parts of input data, while TCAV tests a model's ability to understand specific concepts. Knowledge distillation transfers the learning from a complex model into a simpler, more interpretable model. Counterfactual explanations offer a powerful way to explain and understand predictions by showing the minimal changes required to alter a model's decision.

2.4 Feature Selection and Redundancy/Relevance

Feature selection plays a crucial role in deep learning models, as it determines the subset of features that are most informative for the task at hand. Various feature selection algorithms exist, aiming to keep the most relevant features and eliminate redundant or irrelevant ones. These algorithms can help streamline the learning process, reduce the computational burden, and improve the interpretability of the models by focusing on the most important features.

In addition to feature selection algorithms, frameworks have been developed to assess the redundancy and relevance of features within a dataset. These frameworks provide systematic methodologies to evaluate and rank the importance of features based on their contribution to the model's performance and interpretability. By incorporating redundancy/relevance frameworks, we can enhance the efficiency and interpretability of deep learning models.

2.5 Techniques for Extracting Knowledge from Models

2.5.1 SHapley Additive exPlanations

SHAP [17] is a popular method for interpreting and explaining the predictions of ML models. It is based on the concept of Shapley values, which are a way of assigning credit to each feature in a prediction, based on its contribution to the prediction. SHAP provides a way to compute Shapley values efficiently, even for complex models such as DNN.

The basic idea behind SHAP is to measure the impact of each feature on a prediction, by comparing the model's output for that prediction with its output for a `baseline` prediction. The baseline prediction is typically chosen as the average prediction for the entire dataset, or the prediction for a specific reference point.

To compute the SHAP value for a feature, the model's output is first computed for all possible combinations of features, including the feature of interest and all other features. The difference between the output for the full set of features and the output for the set of features that excludes the feature of interest is then computed, and the average of this difference is taken over all possible combinations of features that include the feature of interest. This average is the SHAP value for the feature.

The SHAP values can be used to generate a summary plot that shows the most important features for a given prediction, along with their contributions to the prediction. The SHAP values can also be used to generate feature importance rankings for the entire dataset, which can be useful for feature selection and feature engineering.

One of the strengths of SHAP is that it provides a unified framework for interpreting and explaining the predictions of different types of models, including linear models, DT, and DNN. It also provides a way to handle complex interactions between features, by computing the SHAP values for all possible combinations of features.

However, SHAP can be computationally expensive, especially for large datasets and complex models. There are also some limitations to SHAP, such as the fact that it only

provides local explanations for individual predictions, and does not necessarily capture the global behavior of the model. Nonetheless, SHAP is a powerful tool for interpreting and explaining the predictions of ML models, and it has become widely used in the ML community.

The SHAP graph in Figure 2.7 shows an example, with the average impact of features on the model's predictions for two classes it shows that perimeter, compactness, and area are the most influential features, while other factors like symmetry, smoothness, radius, fractal dimension and texture play smaller roles in distinguishing between the classes.

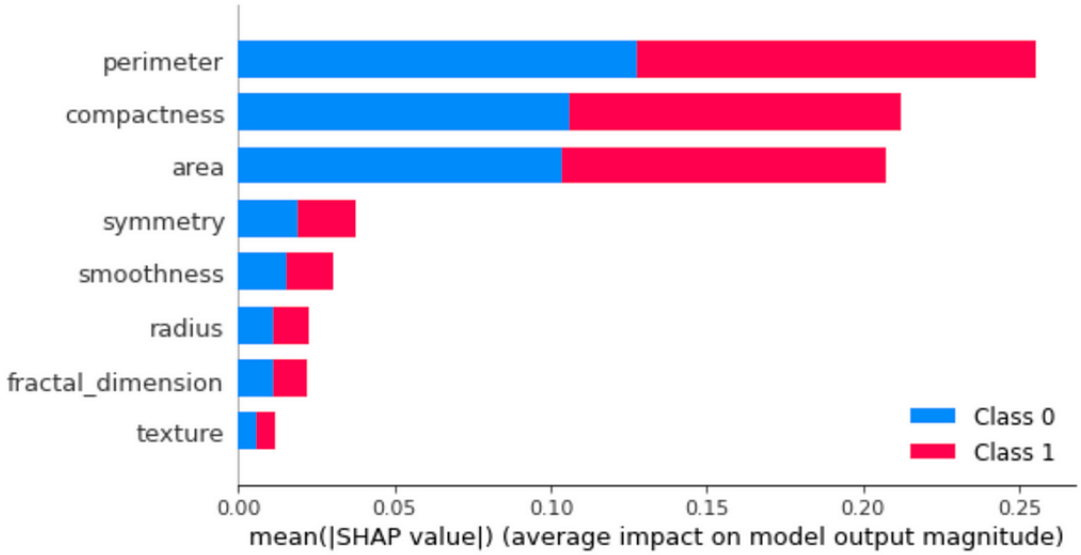


Figure 2.7: *Example of a SHAP graph output*

2.5.2 Local Interpretable Model-agnostic Explanations

LIME [24] is a method for interpreting ML predictions, akin to SHAP. It generates local explanations for individual predictions by approximating the model's behavior with a simpler and interpretable model.

The core concept of LIME involves creating `perturbed` versions of the input data for a given prediction. A local, interpretable model is then fitted to these perturbed data points, elucidating the prediction by highlighting the most crucial features. The perturbed data points are generated through `perturbation sampling`, which involves random sampling from the original input data and introducing noise or altering certain features.

After generating perturbed data points, LIME employs a straightforward, interpretable model (e.g., linear regression or DT), fitting it to the perturbed data using the model's outputs as labels. The weights of the features in the local model indicate the importance of those features for the prediction.

LIME boasts versatility, as it can be applied to any model, irrespective of complexity or data type. It offers a transparent summary of a model's behavior for individual predictions and allows users to specify perturbation levels and local model complexity, balancing accuracy and interpretability.

Despite these advantages, LIME has limitations. Selecting an appropriate perturbation distribution is challenging and can impact the quality of the local model and explanation accuracy. Moreover, LIME provides only local explanations, potentially missing the global model behavior. Nevertheless, LIME is widely used in interpreting and explaining ML models, making significant contributions to explainable AI.

The Figure 2.8 illustrates a LIME explanation, showing that the model predicts the mushroom as poisonous with 100% confidence. The features influencing this prediction, such as `odor=foul` and `stalk-surface-above-ring=silky`, contribute significantly to classifying the mushroom as poisonous, while the feature `gill-size=broad` weakly supports an edible classification.

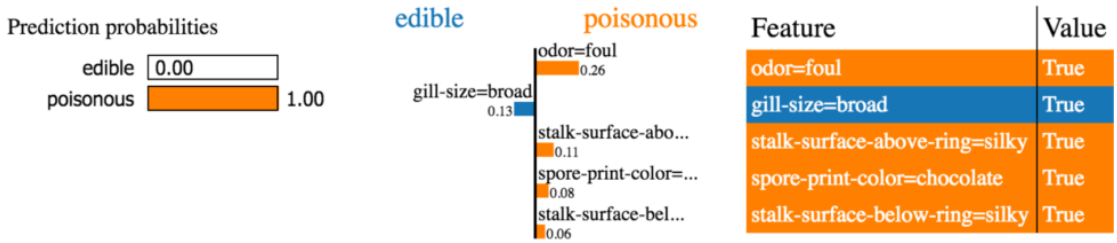


Figure 2.8: *Example of a LIME graph output*

2.5.3 Explainable Boosting Machine

EBM [3] is a method for creating interpretable models using the framework of gradient boosting. EBM was developed by Microsoft Research and is designed to provide a transparent and easy-to-understand alternative to more complex ML models such as DNN. The basic idea behind EBM is to create a set of simple, additive models that can be easily interpreted by humans. Each model is created using a single feature, and all the models are combined using gradient boosting to create a final prediction.

In contrast to other gradient boosting methods, EBM uses a `greedy` algorithm to select the next feature to add to the model, based on its ability to improve the model's accuracy. This approach allows EBM to create models that are both accurate and interpretable, by prioritizing the most important features for the prediction.

One of the strengths of EBM is its ability to handle high-dimensional data with a large number of features. EBM can handle both categorical and continuous features, and can automatically handle interactions between features.

EBM also provides a way to interpret the model's behavior, by generating feature importance plots that show the relative importance of each feature in the model. These plots can help users understand which features are the most important for the prediction, and how the model is making its decisions.

However, as with any method for creating interpretable models, EBM has some limitations. One of the main challenges with EBM is selecting the appropriate number of models and features to include in the final model, which can affect the accuracy of the model and the interpretability of the results. Additionally, EBM may not be as accurate as more complex ML models such as DNN, especially for very large and complex datasets.

Despite these limitations, EBM has become a popular method for creating interpretable ML models, and it has been used in a variety of applications, including healthcare, finance, and marketing.

Figure 2.9 illustrates the behavior of an EBM, an interpretable ML model, in relation to the feature F_N. The top part of the figure represents the model's score or contribution across different values of F_N. The Y-axis displays the score, indicating how much a specific range of F_N influences the final prediction, either positively or negatively. The four colored lines—blue, red, green, and purple—correspond to distinct classes or target outcomes. Variations in these lines signify that the impact of F_N changes depending on its value and the class it influences. When the score increases, it suggests that the corresponding range of F_N positively affects the likelihood of a particular class, whereas a decrease indicates a negative effect. Flat lines imply that the feature has little influence on predictions for that specific class in the given range.

The bottom part of Figure 2.9 provides a histogram of the distribution of F_N values within the dataset. The height of the bars represents the density of data points for each range of F_N, offering context for understanding the model's decisions. For example, if a portion of the score plot exhibits significant variation and corresponds to a high-density region in the histogram, it indicates that the model is making decisions based on a substantial portion of the data. Conversely, regions with low density suggest that the model's behavior in those areas might be less certain due to fewer data points.

Overall, Figure 2.9 provides valuable insights into the local and global effects of the feature F_N on the model's predictions. It demonstrates how the EBM model captures the relationship between F_N and the target variable by adjusting the score accordingly. This plot showcases EBM's transparency and interpretability, making it a reliable tool for understanding feature influence. Such visualizations are crucial for verifying that the model's behavior aligns with domain knowledge and for explaining the decision-making process to both technical and non-technical audiences alike.

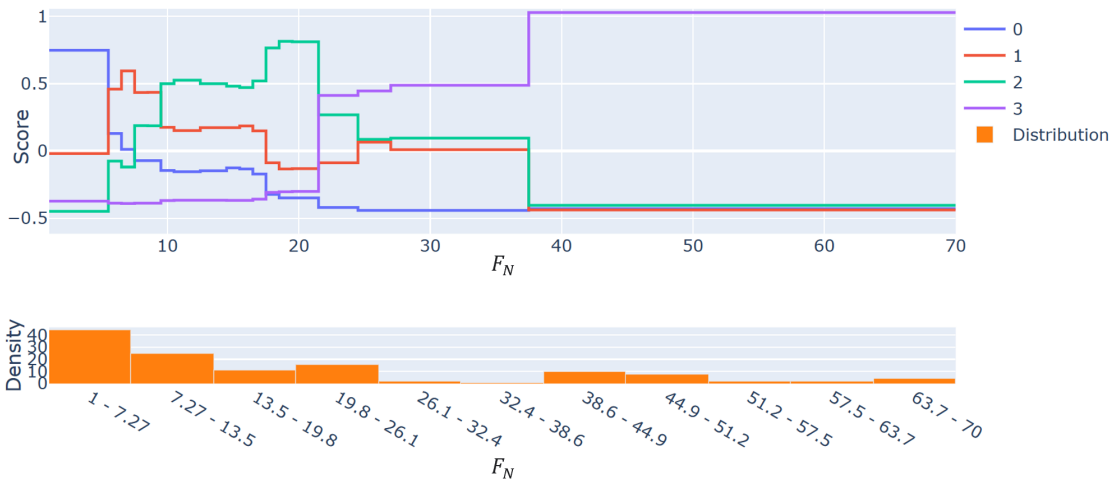


Figure 2.9: *Example of an EBM graph output*

2.5.4 Saliency maps

Saliency maps [19] are a method for interpreting the predictions of DNN by highlighting the most important regions of an input image that contribute to the prediction. Saliency maps are often used in computer vision tasks, such as image classification and object detection, to help users understand why the model is making a particular decision.

The basic idea behind saliency maps is to compute the gradient of the output with respect to the input image, and then use this gradient to generate a heat map that highlights the most important regions of the image. The gradient indicates the sensitivity of the output to changes in the input, and can be used to identify the regions of the image with the greatest impact on the prediction.

One popular method for computing saliency maps is `gradient-based saliency`. In this method, the gradient of the output with respect to the input image is computed using backpropagation. The magnitude of the gradient is then used to generate a heat map that indicates the importance of each pixel in the image. Another method for computing saliency maps is called Class Activation Mapping (CAM). CAM is a technique for generating class-specific heat maps that highlight the most important image regions for a particular class. CAM works by computing the activation maps of the final convolutional layer of the neural network, and then using these activation maps to generate the class-specific heat maps.

Saliency maps have several advantages as a method for interpreting DNN. First, they provide a visual representation of the most important regions of an image, which help users understand why the model is making a particular prediction, as shown in Figure 2.10. Second, saliency maps can be computed quickly and efficiently, making them suitable for real-time applications.

However, saliency maps also have some limitations. One of the main challenges is that they may not always accurately reflect the true importance of each pixel in the image, especially for complex and highly variable datasets. Additionally, saliency maps only provide a local interpretation of the model's behavior for individual predictions, and may not capture the global behavior of the model. Despite these limitations, saliency maps are a widely-used tool for interpreting and explaining the predictions of DNN in computer vision tasks.

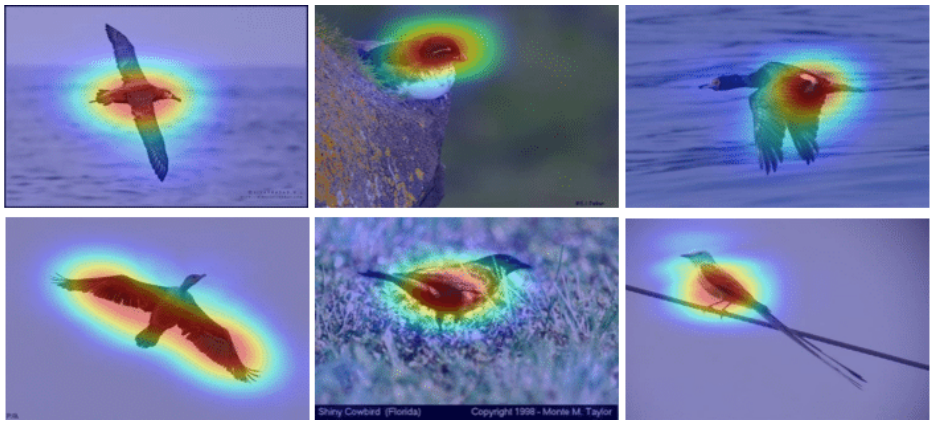


Figure 2.10: *Example of a Saliency Map output*

2.5.5 Testing with Concept Activation Vectors

TCAV [14] is a method for evaluating the interpretability of DNN. Developed by researchers at Google, TCAV helps users understand which concepts or features are important for the model's predictions. The basic idea is to test whether a given concept is learned by the model, by analyzing the sensitivity of the model's predictions to variations in that concept. For instance, in an image classification task, TCAV could test the model's sensitivity to concepts like `stripes` or `texture`.

To perform a TCAV analysis, a user selects a concept of interest and a set of images representing that concept. The user then generates Concept Activation Vector (CAV) that capture the variation of the concept across the images. CAV are computed using a linear classifier trained to classify images as either belonging to the concept or not. TCAV uses these CAV to analyze the model's sensitivity to the concept by computing the dot product between the CAV and the gradients of the model's hidden layers. This reveals how much the model's predictions are influenced by variations in the selected concept.

TCAV scores can be computed for different layers of the model, providing insight into the interpretability of various layers. Additionally, TCAV can be used to compare different models by analyzing and comparing their TCAV scores. A key strength of TCAV is its ability to quantify how much a model has learned a concept and to provide a numerical measure of interpretability. It also helps identify which layers and features are more important for predicting a concept.

However, TCAV has some limitations. The selection of appropriate concepts and representative images for analysis is crucial and can affect accuracy. Moreover, TCAV may not fully capture the complexity of a model's behavior, especially in large datasets. Despite these challenges, TCAV remains a valuable tool for interpreting DNN and has been applied in various fields such as healthcare, finance, and marketing.

Figure 2.11 shows the relationship between different Residual Network (ResNet) layers and the R^2 determination coefficient, measuring the explanatory power of each layer for concepts like correlation, Angular Second Moment (ASM), eccentricity, area, and contrast. The R^2 determination coefficient (ranging from -0.2 to 0.8) represents the strength of the connection between the concepts and the network layers, with higher values indicating stronger explanatory power. Each concept is represented by a color-coded line, and the shaded regions indicate variability across layers. Trends reveal that concepts like ASM have increasing explanatory power in deeper layers, while others like eccentricity show more variability and lower R^2 values.

This graph provides a visual representation of how different ResNet layers correlate with these concepts, highlighting the increasing or decreasing trends in explanatory capacity as the network deepens.

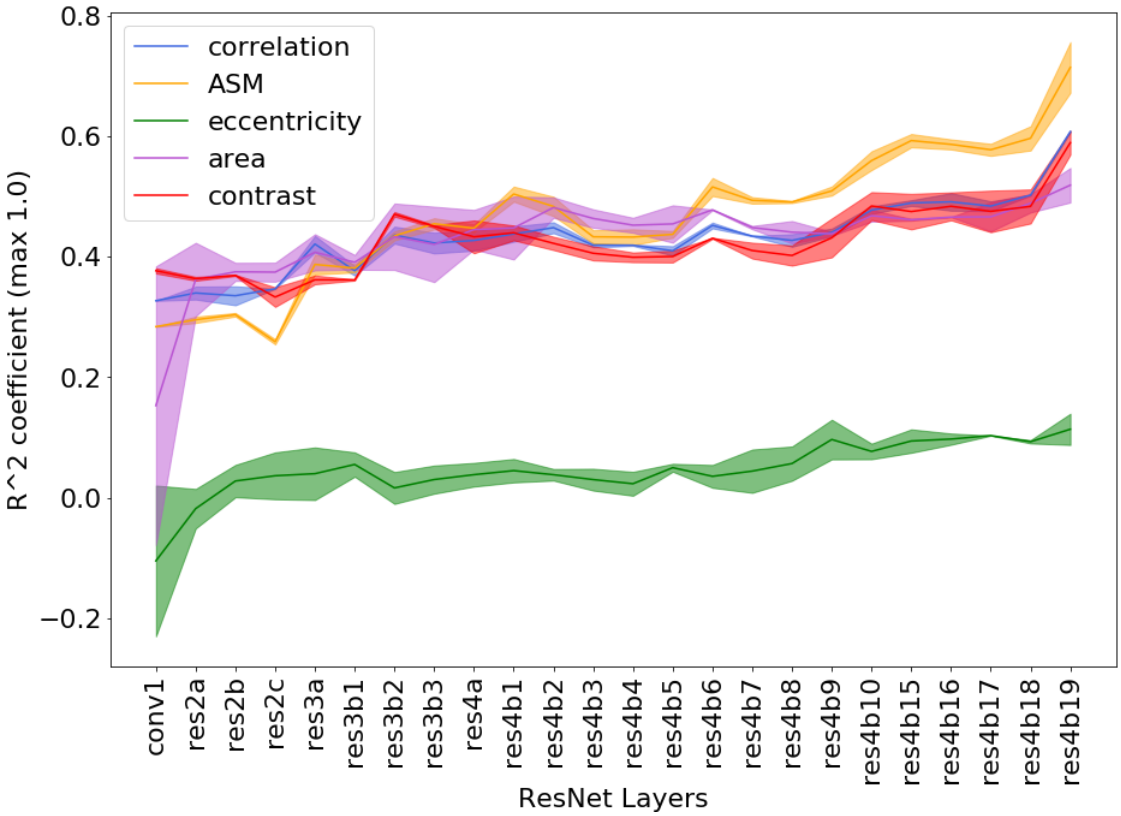


Figure 2.11: *Example of a TCAV graph output*

2.5.6 Distillation

Distillation [15], also known as knowledge distillation, is a technique for compressing the knowledge in a large and complex neural network into a smaller and more compact model. The basic idea behind distillation is to train a smaller `student` model to mimic the behavior of a larger `teacher` model, by transferring the knowledge learned by the teacher to the student.

The process of distillation involves two stages: training the teacher model and training the student model. The teacher model is typically a large and complex neural network that has been trained on a large dataset, and is capable of achieving high accuracy for the task at hand. The student model, on the other hand, is a smaller and simpler neural network that is designed to replicate the behavior of the teacher model, but with fewer parameters.

In the first stage of distillation, the teacher model is trained on the dataset using the standard training procedure. In the second stage, the student model is trained using a modified loss function that includes two terms: one that measures the difference between the predictions of the teacher model and the student model, and another that encourages the student model to produce confident and smooth predictions.

By training the student model in this way, the knowledge learned by the teacher model is transferred to the student model, allowing the student model to achieve similar accuracy to the teacher model, but with fewer parameters and faster inference time. Additionally, the student model can be trained on smaller datasets than the teacher model, which can

be useful in scenarios where data is limited.

Distillation has several advantages as a technique for compressing NN. First, it can reduce the size of a model without sacrificing accuracy, which can be useful for deployment on resource-constrained devices. Second, it can speed up inference time by reducing the number of parameters that need to be computed. Third, it can improve the generalization performance of the model, by encouraging the student model to learn more robust representations of the data.

However, distillation also has some limitations. One of the main challenges with distillation is selecting the appropriate hyperparameters, such as the temperature parameter that controls the softness of the targets. Additionally, distillation may not always capture the full complexity of the teacher model, especially for very large and complex datasets. Despite these limitations, distillation is a valuable technique for compressing NN and has been used in a variety of applications, including computer vision, natural language processing, and speech recognition.

Figure 2.12 provides a visualization of a knowledge distillation process combined with multiple choice learning. On the left hand side, the diagram begins with independent ensemble learning, in which a dataset is used to train several base models, denoted as B_1, B_2, and B_3. These models are trained independently on the same data to archive different predictions. In the middle, the process transitions to knowledge distillation, where the knowledge from each base model is transferred to specialized models, S_1, S_2, and S_3. This transfer is represented by the arrows labeled $P_{\phi_1,T}$, $P_{\phi_2,T}$, and $P_{\phi_3,T}$, with the flow of knowledge from the base to the specialized models. On the right hand side of the figure, multiple choice learning is applied. Each specialized model, S_1, S_2, and S_3, specializes in different parts of the data, D_1, D_2, and D_3, respectively. The circles indicate how the data is organized among the specialized models, with each model being responsible for a specific subset of the data, aiming to improve prediction accuracy by focusing on distinct regions of the input space.

The Figure 2.12 shows how knowledge distillation and multiple choice learning can be combined to leverage the strengths of an ensemble of base models and specialize them for different data subsets.

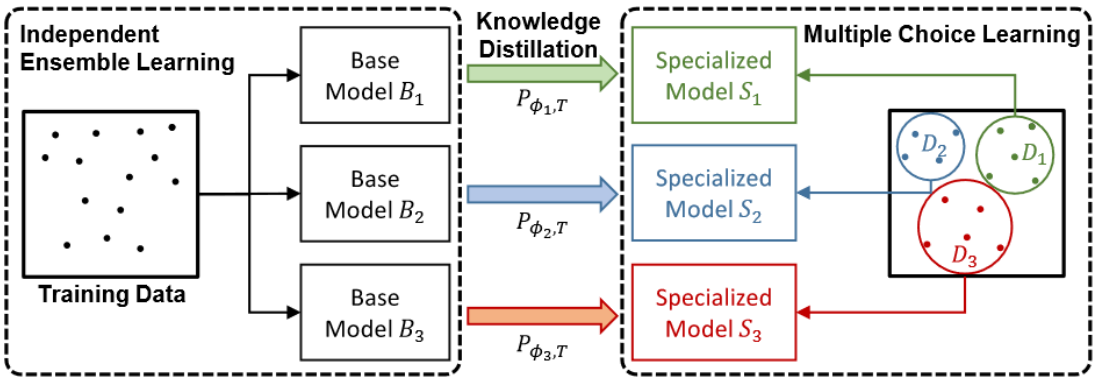


Figure 2.12: *The knowledge Distillation framework*

2.5.7 Counterfactual

Counterfactuals [18] are a type of explanation that attempt to provide an understanding of how changing a particular input to a model would affect its output. Counterfactual explanations are important in situations where we want to understand how a model's decision would change if the input was different, or to understand the factors that are driving the model's prediction.

The basic idea behind counterfactuals is to create a hypothetical example that is similar to the original input, but with one or more changes to specific features or attributes. This new example is called a counterfactual, and is designed to show how the model's output would change if the input had been different. For example, in a loan approval model, a counterfactual might show how changing an applicant's income or credit score would affect their likelihood of being approved for a loan.

Counterfactual explanations can be generated using a variety of techniques, such as optimization-based methods, gradient-based methods, or adversarial attacks. The specific technique used will depend on the type of model being explained, the nature of the input features, and the desired level of interpretability.

One of the key benefits of counterfactual explanations is that they provide a clear and actionable understanding of how a model works, and can be used to identify potential biases or errors in the model. By generating counterfactuals for different inputs, we can see which features or attributes are driving the model's predictions, and identify cases where the model is making predictions based on factors that are unrelated to the desired outcome.

However, counterfactual explanations also have some limitations. One of the main challenges with counterfactuals is that they can be difficult to generate for complex models with high-dimensional inputs, especially if the input features are highly correlated or interact in complex ways. Additionally, counterfactuals may not always be unique, and there may be multiple possible explanations for a particular output.

Despite these limitations, counterfactual explanations are an important tool for understanding and interpreting ML models, and can be used to improve the transparency, fairness, and accountability of AI systems.

Figure 2.13 illustrates the process of generating counterfactual explanations to interpret AI model's prediction. The process begins with an instance, represented by a set of features (f_1, f_2, f_3), which is fed into the AI model. The model processes this input and makes a prediction, in this case, classifying the instance into the negative class (shown in red in the bar graph).

Following this prediction, an XAI library is employed to generate counterfactual explanations. These counterfactual explanations suggest how the input features (f_1, f_2, f_3) need to change in order to flip the model's prediction from the negative class to the positive class (shown in green). The counterfactual changes are depicted in a bar graph where the changes to each feature are listed, showing how each one impacts the model's decision.

Finally, by applying these suggested changes to the features, the model's prediction shifts from the negative class to the positive class, achieving the desired outcome. This

counterfactual process is a valuable tool for understanding what minimal changes could alter a model's decision, providing insights into the underlying behavior of the AI system.

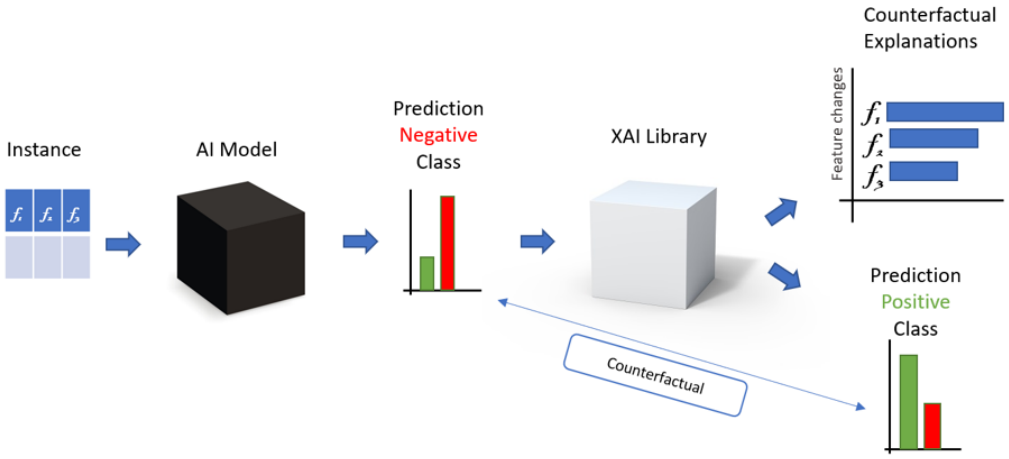


Figure 2.13: *The Conterfactual approach*

2.6 Image Analysis with convolutional Neural Networks

2.6.1 Image analysis with convolutional Neural Networks

The architecture of CNN is inspired by the visual processing mechanisms of the human brain. It is composed of several types of layers, each one playing a crucial role in the network's functionality:

- **Convolutional Layers** - The core component of a CNN is the convolutional layer. It applies convolution operations to the input data, using a set of filters (kernels) to extract features such as edges, textures, and patterns. These features become more complex as the network deepens.

- **Pooling Layers** - Pooling layers follow convolutional layers to reduce the spatial dimensions of the feature maps. This process, known as down-sampling, helps in reducing the computational load and controlling overfitting by making the network invariant to small transformations.

- **Activation Functions** - Activation functions introduce non-linearity into the network, enabling it to learn complex patterns. The most commonly used activation function in CNN is the ReLU, which replaces all negative pixel values in the feature map with zero.

- **Fully Connected Layers** - In the final stages of the CNN, fully connected layers are used to combine the features extracted by the convolutional and pooling layers to make predictions. These layers are typically found at the end of the network and are responsible for mapping the high-level features to the output classes.

- **Dropout Layers** - Dropout is a regularization technique used in CNN to prevent overfitting. During training, a fraction of the neurons is randomly turned off, forcing the network to learn more robust features.

2.6.2 Key concepts

The Key Concepts in Convolutional Neural Network are:

- **Filters and Kernels** - Small matrices used in convolution operations to extract features from the input image. The size and number of filters determine the network's ability to capture different aspects of the image.

- **Stride and Padding** - The step size by which the convolution filter moves across the image. Padding involves adding extra pixels around the image's borders to control the spatial dimensions of the output feature maps.

- **Feature Maps** - The outputs of the convolutional layers, representing the activation of different filters applied to the input image. Feature maps provide a detailed representation of the image's essential features at various levels of abstraction.

2.6.3 Training

Training a CNN involves adjusting the network's weights through backpropagation and optimization algorithms such as the Stochastic Gradient Descent (SGD). During training, the network minimizes a loss function, such as cross-entropy loss for classification tasks, by iteratively updating the weights to improve its predictions.

Regarding training, we also have the following relevant concepts:

- **Learning Rate** - Controls the step size during weight updates. Choosing an appropriate learning rate is crucial for ensuring that the network converges to an optimal solution without overshooting.

- **Batch Normalization** - This technique normalizes the inputs of each layer, which can speed up training and improve the network's stability by reducing the risk of internal covariate shifts.

2.6.4 Applications

CNN have revolutionized various fields by providing state-of-the-art solutions for a wide range of applications, as follows:

- **Image Classification** - CNN are widely used in image classification tasks, where they automatically categorize images into predefined classes. Notable examples include ImageNet classification, where CNN have achieved superhuman performance.

- **Object Detection** - In object detection, CNN classify objects within an image and also locate them by drawing bounding boxes. Architectures such as Faster R-CNN, You Only Look Once (YOLO), and Single Shot MultiBox Detector (SSD) are prominent in this domain.
- **Semantic Segmentation** - CNN are also employed for pixel-wise classification tasks such as semantic segmentation, where each pixel in an image is labeled with a corresponding class. This is crucial for applications such as autonomous driving.
- **Medical Image Analysis** - CNN are increasingly used in medical imaging to assist in diagnosing diseases from X-rays, Magnetic Resonance Imaging (MRI), and Computed Tomography (CT) scans. Their ability to detect subtle patterns makes them valuable in identifying conditions such as tumors, fractures, and organ anomalies.
- **Natural Language Processing (NLP)** - While CNN are primarily known for image processing, they are also used in NLP tasks, such as text classification and sentiment analysis, by treating text as a sequence of spatially arranged features.

2.6.5 Challenges and Future Directions

Despite their success, CNN face several challenges:

- **Data Requirements** - CNN require large labeled datasets for training, which can be a limitation in domains where data is scarce or expensive to obtain.
- **Computational Resources** - Training deep CNN is computationally intensive, requiring powerful GPU and significant memory. This can be a barrier for those with limited access to high-end hardware.
- **Explainability** - CNN are often criticized for being `black boxes`, meaning their decision-making process is not easily interpretable. Researchers are actively exploring methods to make CNN more transparent and explainable.
- **Generalization** - Ensuring that CNN generalize well to new, unseen data remains a challenge. Techniques such as transfer learning, data augmentation, and adversarial training are used to address this issue.

The future of CNN looks promising, with ongoing research focused on improving their efficiency, robustness, and applicability to new domains. Innovations such as capsule networks, attention mechanisms, and neural architecture search are paving the way for the next generation of CNN.

2.6.6 ResNet Models and Saliency Maps

ResNet, or Residual Network [13], is a type of convolutional neural network that is commonly used for image classification tasks. It is known for its deep architecture and the use of shortcut connections, which help to mitigate the problem of vanishing gradients during training.

A pre-trained ResNet model is a model that has already been trained on a large dataset, such as ImageNet. Using a pre-trained model can save a lot of time and computational resources, as the model has already learned a lot of useful features from the training data.

In this thesis, we used a pre-trained ResNet model to generate saliency maps for a set of images. The saliency maps were then used to visualize which parts of the images were more important for the model's predictions. This helped us to gain insights into how the model was making decisions and which features it was focusing on.

The results showed that the model was able to correctly identify the most important regions of the images, such as the faces of people and the shapes of objects. This demonstrated that the model was able to learn meaningful features from the training data and use them to make accurate predictions. The saliency maps provided a useful visualization of the model's decision-making process and helped to interpret its predictions.

Overall, saliency maps are a powerful tool for understanding how deep learning models work and which features they are focusing on when making predictions. They can provide valuable insights into the inner workings of the model and help to identify areas for improvement. By visualizing the important regions of an image, saliency maps can help to build trust in the model and ensure that it is making decisions based on meaningful features.

In the context of image classification, a common task is to distinguish between different categories of images. For instance, you might want to train a model to differentiate between images of cats and dogs. This is a binary classification problem, but the same principles can be extended to multiclass classification problems, with more than two categories.

To use a pre-trained ResNet model for a cats vs dogs classification task, we first need to fine-tune the model on your specific dataset. This involves replacing the final layer of the model with a new layer that has the correct number of output nodes (two in this case), and then training the model on your dataset while keeping the weights of the pre-trained layers fixed.

2.6.7 Final Remarks

CNN have dramatically transformed the landscape of ML and AI, particularly in the field of computer vision. Their ability to learn and extract complex features from raw input data has enabled breakthroughs in numerous applications. As research continues to advance, CNN are likely to become even more powerful, driving innovation across diverse industries and domains.

3 Proposed solution

This deals with the emerging necessity of model explainability in ML models since they become complex, sometimes incomprehensible. This turns around such interpretability algorithms as DT [5], SHAP [17], and EBM [3] that can underline key features in predictive models and disclose how they come up with their insights. For image data, LIME [24] and saliency maps [19] are used to identify the important pixels, thus providing an insight into how models manage to achieve such high accuracies on image classification.

In this work, authors use Iris [9] and Adult [23] datasets comprising tabular data. These datasets train the models where algorithms of interpretability would help in unearthing critical features behind predictions. DT offer inherent interpretability, while SHAP and EBM provide both global and local explanations of feature importance.

EBM balances accuracy and interpretability with additive modeling. SHAP is used similarly as for additive models of Gradient Boosting combining over many trees. Cats and Dogs dataset was used as the image data multiclass classification using CNN [4] to classify. Saliency maps highlight the important pixels, while LIME generates augmented images to help further explain.

It is shown here that SHAP, DT, and EBM are able to find salient features on the Iris and Adult datasets, their intuitive interaction with the features, and their contribution to model outputs. For image classification, visual explanations using LIME and saliency maps are able to indicate that the CNN models pick regions around key facial features and fur patterns as discriminative for classes. Pixel visualizations provide insight into the model's high accuracy. The key takeaway from such interpretability algorithms is that a much better insight into ML models is obtained through the decision-making process disclosed. In other words, the ability to define a set of features and understand in what way such models obtain their predictive power enables us to build more transparent and trustworthy AI systems. This research emphasizes the importance of interpretability when it comes to developing a model and proves that it leads to more effective and ethical applications in ML.

3.1 Synthetic Dataset

3.1.1 Generating and Leveraging Synthetic Data

Synthetic data offers significant advantages beyond just research and controlled experimentation. One of the main benefits is its ability to overcome challenges related to data privacy and security. In industries namely healthcare, finance, or law enforcement, access to real-world data may be restricted due to sensitive personal information. Synthetic data allows researchers and developers to create datasets that are free from such constraints while preserving the essential characteristics needed for meaningful analysis. This enables organizations to test ML models, develop new technologies, and train systems without risking exposure to confidential information. Furthermore, synthetic data can simulate rare or extreme conditions that may not be sufficiently represented in real-world data, providing more robust testing environments for AI systems in safety-critical applications, such as autonomous driving or medical diagnosis.

Another compelling aspect of synthetic data is its scalability. Collecting, cleaning, and labeling real-world datasets can be time-consuming and expensive. In contrast, synthetic data generation can be automated, allowing researchers to quickly produce large datasets tailored to their specific needs. This opens up opportunities to train models on a broader range of scenarios, leading to more generalized and adaptable ML systems. Additionally, synthetic data can help address the common issue of bias in ML models by allowing for the creation of balanced and diverse datasets, ensuring that algorithms are tested on a wide variety of controlled inputs.

As synthetic data generation techniques continue to improve, they offer a promising path toward building more reliable, scalable, and ethical AI systems.

3.1.2 Dataset Features and Relevance

In this study, two synthetic datasets were created to explore the role of feature relevance in modeling. Both datasets contained a total of 8 features, differing in the number of important features related to the target variable. In the first dataset, the challenge was to identify the single crucial feature among several irrelevant ones. The second dataset, with three important features, required balancing the influence of multiple key variables. This setup allowed for an analysis of how models handle both isolated and interacting relevant features, providing insights into how feature relevance impacts predictions.

These synthetic datasets offer a controlled environment to understand how models prioritize important information and manage irrelevant features. By comparing the two scenarios, this approach highlights the importance of feature relevance in predictive performance and interpretability.

3.2 Iris Dataset

3.2.1 Overview of the Dataset

The Iris dataset is one of the most well-known datasets in the ML community, often used as a benchmark for testing and demonstrating various algorithms. Introduced by the British biologist and statistician Ronald A. Fisher in 1936 [9], the dataset contains 150 samples of iris flowers, divided equally among three species: Iris setosa, Iris versicolor, and Iris virginica. Each sample is characterized by four features: sepal length, sepal width, petal length, and petal width, all measured in centimeters. These features allow for the differentiation between the species based on their morphological differences.

The simplicity and small size of the Iris dataset make it an excellent starting point for those new to ML, providing a clear and intuitive example of supervised classification tasks. Despite its simplicity, the dataset presents a challenge in terms of distinguishing between Iris versicolor and Iris virginica, as these two species have overlapping feature values. This characteristic makes it particularly useful for testing the effectiveness of classification algorithms and feature selection methods. The Iris dataset continues to be a valuable educational resource and a benchmark for evaluating new algorithms in the field of pattern recognition and ML.

3.2.2 Dataset Features and Relevance

The Iris dataset features four distinct measurements: sepal length, sepal width, petal length, and petal width. Each of these features provides critical information that aids in the classification of the three iris species. Among these, petal length and petal width are particularly significant, as they exhibit greater variability between species compared to sepal measurements. This variability makes them highly discriminative, especially for differentiating between Iris setosa and the other two species, as well as between Iris versicolor and Iris virginica to a lesser extent. Sepal length and width, while also important, generally offer less discriminative power on their own but contribute valuable information when combined with petal measurements. Understanding the relevance of these features is essential for developing accurate classification models and provides insights into the morphological distinctions between the species, demonstrating the importance of feature analysis in pattern recognition and ML tasks.

3.3 Adult Dataset

3.3.1 Overview of the Dataset

The Adult dataset, also known as the `Census Income` dataset, is a popular dataset used for ML and data mining tasks, particularly in the realm of classification. Collected from the 1994 U.S. Census [23], it contains information about 48,842 individuals, each described by 14 attributes. These attributes include both continuous and categorical features: age, workclass, fnlwgt, education, education-num, marital-status, occupation, relationship, race, sex, capital-gain, capital-loss, hours-per-week, and native-country. The

primary task associated with this dataset is to predict whether an individual's income exceeds 50,000 dolars per year, formulated as a binary classification problem.

The Adult dataset is widely used to evaluate the performance of classification algorithms and to explore issues related to feature selection, data preprocessing, and the handling of imbalanced classes. The mix of categorical and continuous variables presents a realistic challenge for preprocessing and model training. Additionally, the dataset includes inherent biases that make it a useful tool for studying fairness and ethics in ML, as it reflects real-world disparities that can affect model outcomes. As such, the Adult dataset serves as a comprehensive resource for developing and testing ML models, while also encouraging the examination of broader societal impacts.

3.3.2 Dataset Features and Relevance

The Adult dataset comprises 14 features that provide a comprehensive snapshot of an individual's demographic and economic characteristics. These features include both continuous variables, such as age, fnlwgt (final weight), education-num (number of years of education), capital-gain, capital-loss, and hours-per-week, as well as categorical variables such as workclass, education, marital-status, occupation, relationship, race, sex, and native-country. Each feature plays a crucial role in predicting whether an individual's income exceeds 50,000 dolars per year. For instance, continuous features in particular age and education-num are often directly correlated with income, reflecting the impact of experience and education on earning potential. Categorical features such as occupation and workclass are also highly relevant, as they directly pertain to an individual's employment status and job type, which are strong indicators of income level. Understanding the relevance and interplay of these features is essential for building accurate and robust classification models, and for drawing meaningful insights from the data regarding socioeconomic patterns and disparities.

3.4 Cats and Dogs Dataset

3.4.1 Overview of the Dataset

The Cats and Dogs dataset [11] is a widely used dataset in ML and computer vision tasks, particularly in image classification problems. It consists of images of various cat and dog breeds, making it ideal for exploring multiclass classification challenges. The dataset contains images representing 37 distinct classes, with 12 cat breeds and 25 dog breeds. These classes include cat breeds in particular Abyssinian, Bengal, Birman, Bombay, British Shorthair, Egyptian Mau, Maine Coon, Persian, Ragdoll, Russian Blue, Siamese, and Sphynx. The dog breeds include American Bulldog, American Pit Bull Terrier, Basset Hound, Beagle, Boxer, Chihuahua, English Cocker Spaniel, English Setter, German Shorthaired Pointer, Great Pyrenees, Havanese, Japanese Chin, Keeshond, Leonberger, Miniature Pinscher, Newfoundland, Pomeranian, Pug, Saint Bernard, Samoyed, Scottish Terrier, Shiba Inu, Staffordshire Bull Terrier, Wheaten Terrier, and Yorkshire Terrier. Each image is labeled with its respective breed, providing a structured dataset for training and evaluating

ML models.

The primary task associated with the Cats and Dogs dataset is to accurately classify each image into one of the 37 categories. This dataset is commonly used to test the performance of CNN and other deep learning architectures. The dataset also allows researchers to experiment with data augmentation techniques, explore the impact of breed diversity on model accuracy, and assess the effectiveness of transfer learning using pre-trained models. Its wide variety of images, which include different orientations, backgrounds, and scales, makes it a valuable resource for advancing image recognition tasks in ML.

3.4.2 Dataset Features and Relevance

The Cats and Dogs dataset includes images that exhibit unique visual features relevant to each breed. These features include patterns, colors, and textures that distinguish different breeds of cats and dogs. For the 12 cat breeds, visual features such as fur patterns, body size, and ear shape help distinguish breeds such as the Abyssinian, Bengal, and Persian. The unique facial structure of the Sphynx or the thick coat of the Maine Coon are also key features that a model might use to make accurate predictions. Similarly, for the 25 dog breeds, the dataset captures a variety of distinctive traits. For example, the small body size and short fur of a Chihuahua, the characteristic wrinkles of a Pug, and the large, muscular build of a Saint Bernard provide vital information for classification.

In model training, techniques such as edge detection, texture analysis, and color recognition play a critical role in identifying breed-specific features. Models leverage these visual attributes to learn patterns that distinguish breeds from one another. For instance, the long, flowing fur of a Samoyed, the short, sleek coat of a Boxer, or the compact frame of a Miniature Pinscher are all key characteristics that help define their breed.

Advanced model namely ResNet and Visual Geometry Group (VGG) have been shown to perform exceptionally well on this dataset, leveraging deep learning to extract complex hierarchical features. These models highlight the importance of feature extraction and the role of convolutional layers in processing intricate visual data. The Cats and Dogs dataset serves as an essential tool for evaluating ML models while also offering insights into the intricacies of breed classification.

3.5 Methodology and Experimental Setup

3.5.1 Data Acquisition and Sources

The experimental setup involves three datasets for the analysis: the Iris and Adult datasets for tabular data, and the Cats and Dogs dataset for image data. Using Python, the datasets Iris and Adult were both loaded from the SHAP library and the library `scikit-learn` performed preprocessing and splitting the data. For image data, the `tensorflow` and `keras` libraries were used to handle and preprocess the Cats and Dogs dataset.

In this study, DT, EBM, and gradient boosting trees were employed for model training. These tree-based models do not require feature scaling, as they split the data based on feature values rather than relying on distance metrics. As a result, feature scaling

techniques like `StandardScaler` were not applied, allowing the features to retain their original scales and units, which enhances interpretability.

The image dataset used in this study consists of images of cats and dogs, with labels corresponding to 37 different breeds. These images were downloaded from an online repository, specifically from the Oxford-IIIT Pet Dataset [21]. Along with the images, a pickle file (`Oxford-IIIT-Pet_Dics.p`) was provided, with metadata such as the breed labels, file names, and additional information required for organizing the dataset into training, validation, and testing sets.

After loading the pickle file, the class and file name dictionaries were extracted, allowing for the organization of the images into appropriate directories. A custom function, `createFolders`, was used to automatically create directories for each breed within the training, validation, and testing folders. Another function, `splitData`, was responsible for moving the images to the correct directories based on their fold assignment in the metadata, which was obtained from the pickle file.

The dataset was organized into three distinct sets: training, validation, and testing, ensuring that each breed had its corresponding images properly allocated. This structured organization was crucial for feeding the images into the model during training and evaluation.

Once the dataset was organized, the Keras image data generators were employed to load the images into memory for training, validation, and testing. The generators were configured to load images in batches, resize them to a predefined target size, and convert the labels into categorical format for the multi-class classification task. This setup ensured that the images were efficiently loaded and fed into the model for training, validation, and testing.

To facilitate model training, a pre-trained ResNet50V2 was used as the base model, combined with a custom CNN architecture based on MobileNetV2. The model was designed for multi-class classification, with a final softmax layer used to predict the probability distribution across the 37 breeds. After training, the model achieved an accuracy of approximately 87% on the test set.

To ensure faster loading and reproducibility in future use cases, the trained model's weights were saved using the `custom_cnn_model.weights.h5` file. This approach allows the model to be reloaded and used without being retrained, ensuring consistent performance and saving computational resources in future sessions.

3.5.2 Decision Tree

A DT classifier was implemented using the `ClassificationTree` from the `interpret` library, which is designed with interpretability as the core focus. This model, part of the InterpretML toolkit, ensures that the decision-making process is transparent and easy to follow. The model was trained on the synthetic, Iris, and Adult datasets with default hyperparameters to highlight feature importance and interpretability.

The `DecisionTreeClassifier` from the `scikit-learn` library, while widely used, was not employed in this study. One key reason is that `DecisionTreeClassifier` focuses

more on performance and flexibility rather than interpretability. It offers extensive options for tuning, such as adjusting tree depth and controlling node splitting, which can optimize for accuracy and computational efficiency. However, its outputs are not inherently focused on producing explanations that are easy to interpret, making it less suited for applications where interpretability is a priority.

The primary advantage of `ClassificationTree` over `DecisionTreeClassifier` lies in its status as a glassbox model. Glassbox models, such as `ClassificationTree`, are designed for complete transparency, allowing users to fully understand the decision-making process. Every split and decision within the tree can be traced back to specific features, ensuring that the model's behavior is easy to explain and interpret. In contrast, `DecisionTreeClassifier`, while flexible and performance-oriented, does not inherently prioritize this level of transparency, which can result in a more opaque model. Given the focus of this study on explainability and interpretability, `ClassificationTree` was chosen for its clear, glassbox nature, ensuring that all decisions made by the model are fully understandable.

3.5.3 SHapley Additive exPlanations

SHAP was applied in this study to interpret the predictions made by the Extreme Gradient Boosting (XGBoost) [29] model. SHAP values provide insight into how each feature contributes to the prediction by calculating the contribution of each feature relative to the average prediction. This method ensures both global and local interpretability, making it easier to understand how the model makes decisions for individual instances as well as across the entire dataset.

XGBoost is a scalable and efficient implementation of gradient boosting designed for high performance in predictive modeling tasks. It builds an ensemble of DT in a sequential manner, where each tree attempts to correct the errors of the previous one. The core idea behind XGBoost is to combine the predictions of multiple weak learners (typically DT) to form a strong prediction model.

Each tree in the XGBoost model is trained to minimize a loss function, typically using gradient descent to optimize performance. At each iteration, a new tree is added, and it focuses on reducing the residuals (errors) of the previous trees. This allows the model to iteratively improve, with each new tree correcting the mistakes made by earlier ones.

XGBoost includes several key features that enhance its performance and flexibility. One such feature is `regularization`, which helps prevent overfitting by adding penalties (via parameters like `lambda` and `alpha`), allowing the model to generalize better to unseen data. Another important feature is `weighted quantile sketch`, which enables XGBoost to handle weighted datasets efficiently, improving its performance on imbalanced data. Additionally, `parallelization` is built into XGBoost, allowing tree construction to be executed across multiple cores, making the model significantly faster than traditional gradient boosting implementations. Finally, XGBoost is `sparsity aware`, meaning it can handle missing values or sparse data by treating them in a way that minimizes their impact on the loss function.

The XGBoost model consists of a set of DT where each tree is built to minimize a predefined loss function, such as mean squared error for regression or log-loss for classification. During training, XGBoost optimizes the model's performance by adjusting parameters such as the learning rate, number of estimators (trees), and maximum tree depth. These parameters control how aggressively the model fits the data and how complex the individual trees are.

In this study, XGBoost was selected for its ability to handle large datasets efficiently and provide robust performance across a variety of tasks. Its combination of DT-based learning and gradient boosting makes it an ideal choice for generating accurate predictions, while its regularization mechanisms ensure that the model does not overfit the data.

After training the XGBoost model, using `shap.TreeExplainer`, SHAP values were computed to interpret the model's predictions. These values were then used to generate various visualizations that offer detailed insights into feature importance and interactions. The following SHAP visualizations were employed:

- `Bar Plot` - The SHAP bar plot provides a global view of feature importance, ranking features by their average absolute SHAP values. This plot highlights which features contribute the most to the model's predictions overall, helping to quickly identify the most impactful variables.

- `Beeswarm Plot` - The beeswarm plot offers a more detailed visualization, showing both the magnitude and direction of SHAP values for individual data points. This plot reveals the distribution of SHAP values for each feature, giving insight into how different features affect predictions for different instances.

- `SHAP Force Plot` - The SHAP force plot visualizes individual predictions by showing how each feature pushes the prediction higher or lower compared to the average prediction. This plot provides a localized view of how the model made a decision for a specific instance, making it useful for understanding predictions on a case-by-case basis. In addition to individual predictions, the SHAP force plot was also applied to visualize the cumulative impact of features across multiple predictions. The plot was generated using `shap.plots.force` with the first 100 SHAP values, displaying how features influenced the model's decisions for these instances. This cumulative force plot provided a broader overview of feature contributions across a sample of the dataset, highlighting patterns in how certain features consistently affected predictions. The cumulative force plot was particularly useful for identifying trends and understanding the model's behavior across multiple instances.

- `Waterfall Plot` - The SHAP waterfall plot breaks down individual predictions, showing how each feature cumulatively contributes to the model's final output. This visualization allows for a clear breakdown of the path from the base value (mean prediction) to the actual prediction.

- `Dependence Plot` - The dependence plot illustrates the relationship between a specific feature and the SHAP values. It shows how the SHAP value of a feature changes

as the feature itself changes. This plot is particularly useful for identifying interaction effects between features, as it can highlight nonlinear dependencies on the data.

The combination of these visualizations provided both a global and local understanding of the model's predictions. By applying SHAP to the XGBoost model, the study was able to offer detailed insights into feature importance and interaction, ensuring a comprehensive explanation of how the model arrived at its predictions.

3.5.4 Explainable Boosting Machine

In this study, the EBM was implemented to provide an inherently interpretable model. EBM is a type of Generalized Additive Model (GAM) that learns feature contributions in a transparent and understandable manner.

The model was trained using the `ExplainableBoostingClassifier` from the `interpret` library, with no feature interactions (`interactions=0`) to ensure the simplest form of the model, where each feature contributes independently to the prediction.

By using EBM without interactions, the model isolates the effects of individual features, making it easier to understand their direct influence on the outcome. Each feature is fitted with a set of learned parameters, and these parameters reflect how different values of the feature affect the target variable. This configuration is particularly beneficial when transparency is required, as it allows for a clear visualization of how each feature impacts the predictions. The EBM model was trained on the dataset, and the resulting explanations were visualized through various plots, such as score plots and feature importance charts. These visualizations highlight the specific contribution of each feature, making it easier to trace the model's decision-making process. The lack of interactions between features ensures that the model remains interpretable at every level, with each feature's impact on the prediction being easily understood.

Overall, the use of EBM in this study provides both global and local explainability and interpretability. The model's structure, based on additive feature contributions, ensures that predictions are made in a transparent manner. This setup allows for a straightforward explanation of how each feature influences the outcome, making EBM an ideal choice for applications where model transparency is crucial.

3.5.5 Local Interpretable Model-agnostic Explanations

LIME was employed to provide local interpretability for image classification predictions made by a custom CNN model. The process began by selecting an individual image from the test set, which was normalized for use with LIME. To generate an explanation, the `LimeImageExplainer` was initialized to analyze the model's predictions at the pixel level. The custom CNN model's prediction function was passed to LIME as the model to explain, with the number of top classes to explain set to 5 (`top_labels=5`). Although the cats and dogs dataset contains 37 classes, explaining the top 5 classes is generally sufficient to capture the most relevant predictions. This approach focuses on the highest-confidence predictions made by the model, which helps in understanding its decision-making process without overcomplicating the explanation by including all possible classes. Limiting the

number of classes in this way is also more computationally efficient, as it reduces the number of explanations that need to be generated, while still offering valuable insights into the model's behavior.

The explainer generated 1,000 perturbed samples from the original image, altering specific pixel regions (superpixels) to understand how they influenced the prediction. Each of these samples was evaluated using the CNN model, and the explainer identified which pixel regions contributed most to the model's decision.

For the class explanation, the label with the highest prediction confidence was selected, and LIME generated an explanation using a superpixel-based approach. The explanation highlights the areas of the image that positively contributed to the prediction, by decomposing the image into interpretable segments (superpixels). The final result displayed both the original image and the superpixels that contributed to the model's decision, with positive contributions clearly visualized. This approach helped provide localized interpretability for individual predictions by showing which areas of the image had the greatest influence on the CNN's classification. By isolating specific superpixels and explaining how they affected the model's output, LIME made it easier to understand the decision-making process of the CNN, offering clear insights into how the model interprets visual features.

3.6 Project Repository

The code for this project are available on GitHub. The repository contains Jupyter Notebooks used for data analysis, model training, and interpretability techniques, along with a README file that provides setup and usage instructions.

You can access the repository at: https://github.com/sexyfafe/thesisA45412.

4 Synthetic Datasets

In this chapter, we present the results obtained from applying the DT, SHAP and EBM methods to the synthetic dataset. The results include performance metrics such as accuracy, alongside visualizations generated by DT to aid in the interpretation of feature importance and interactions. These visualizations includes SHAP various plots, helping to explain the impact of individual features on model predictions, while EBM's transparent modeling provides further insight into how the features contribute to the final outcomes.

4.1 Eight features with one important feature

4.1.1 Decision Trees

Figure 4.1 shows the DT generated using the `ClassificationTree` from the `interpret` library, applied to the synthetic dataset with 8 features, where 1 feature is particularly important. The tree structure highlights the primary feature's dominance in splitting decisions, while other features contribute less. The node weights represent the number of data points that follow each branch, offering insights into how the model distributes importance and makes predictions based on the data distribution.

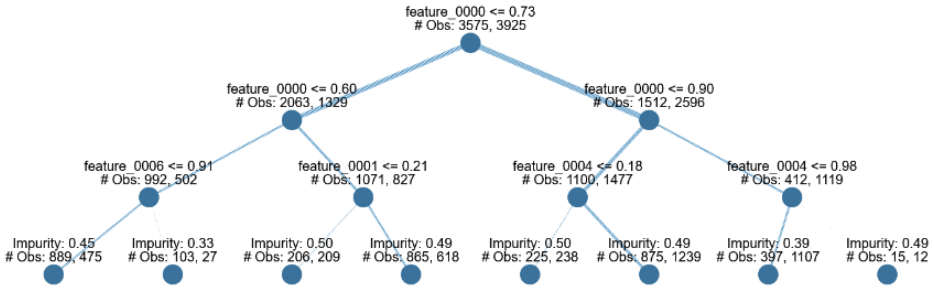


Figure 4.1: *Decision Tree for the Synthetic dataset with 8 features 1 important*

Figure 4.2 illustrates the same DT generated using the `ClassificationTree` from the `interpret` library, applied to the synthetic dataset. This figure specifically highlights the path taken by the model to correctly classify an instance with the label `False`. The path is marked in orange, showing the sequence of splits that led to the correct decision. The visualization also reflects the weights at each node, representing the number of data points that followed the same decision path. This graphical representation offers a clear understanding of how the model arrived at the correct classification for this instance by focusing on the most important feature and subsequent less significant ones.

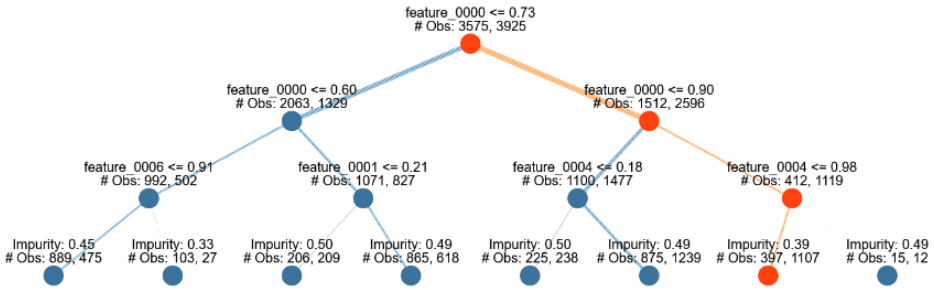


Figure 4.2: *Decision Tree for the Synthetic dataset, highlighting label False*

Figure 4.3 illustrates the DT from the `ClassificationTree` applied to the synthetic dataset, highlighting a path that led to a correct classification with the label `True`. The path, shown in orange, traces the decision-making process from the root to the final node, demonstrating how the model navigates through the key feature and other less significant features. The weights along the path represent the number of data points processed at each decision point, providing a clear view of how the model arrived at the correct classification. This visualization reinforces the model's interpretability by showing the importance of the primary feature and the path it follows to achieve accurate results.

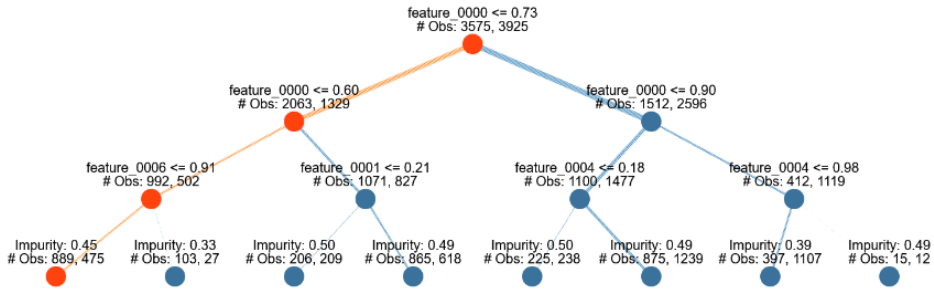


Figure 4.3: *Decision Tree for the Synthetic dataset, highlighting label True*

4.1.2 SHapley

Figure 4.4 displays the SHAP bar plot, which illustrates the contribution of each feature to the model's predictions in terms of SHAP values. Feature 0 stands out with the highest SHAP value, confirming its status as the most important feature in the synthetic dataset. The remaining features show smaller contributions, with their SHAP values indicating less influence on the model's predictions. This provides a clear ranking of feature importance based on their impact on the model, reinforcing the dominant role of Feature 0.

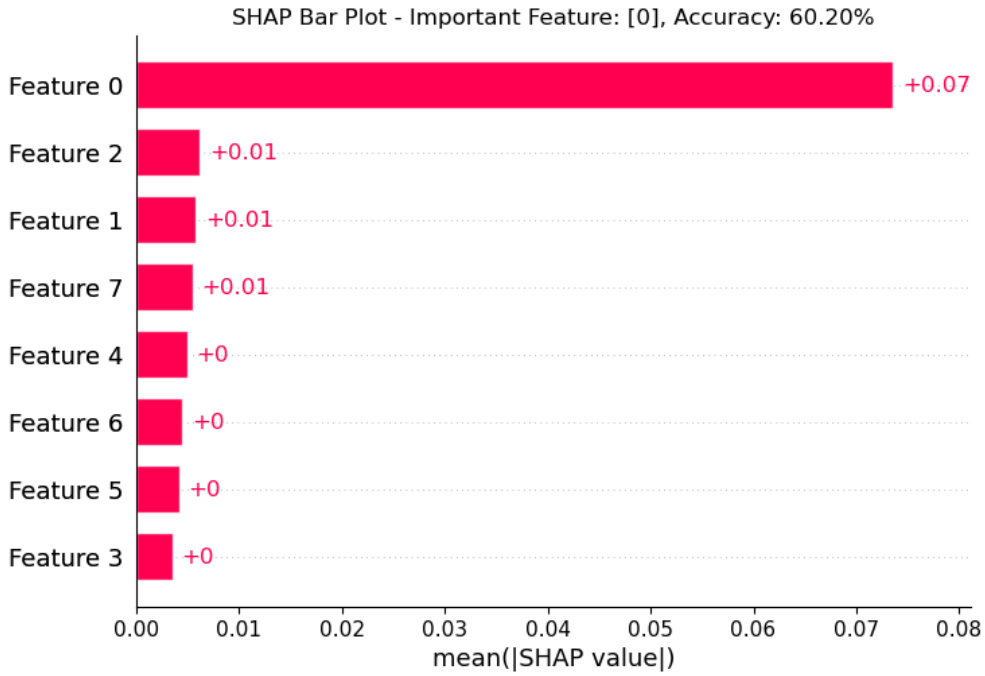


Figure 4.4: *SHAP bar plot for the Synthetic dataset with 8 features 1 important*

Figure 4.5 presents the SHAP beeswarm plot, showing the distribution of SHAP values for each feature across all instances in the synthetic dataset. Feature 0 clearly has the most significant impact, as indicated by its wide range of SHAP values, affirming its role as the most important feature. The remaining features contribute less to the model's predictions, with narrower distributions of SHAP values. The plot also reveals a substantial amount of noise, which was intentionally added to simulate real-world scenarios. This noise highlights the complexity of feature interactions and their varying contributions across different instances, further illustrating the challenges of model interpretation in realistic settings.

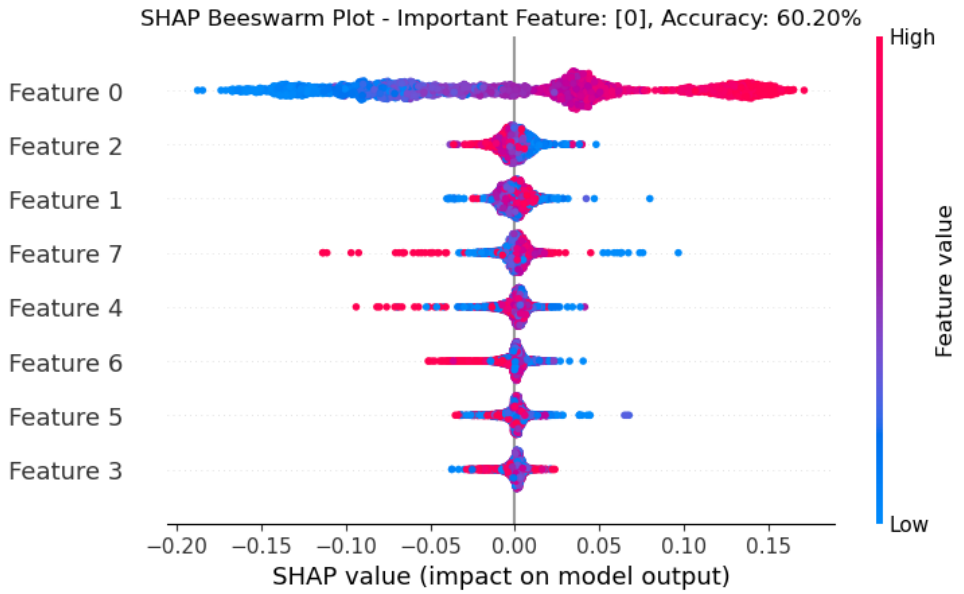


Figure 4.5: *SHAP beeswarm plot for the Synthetic dataset with 8 features 1 important*

Figure 4.6 illustrates the SHAP force plot for Feature 0, the most important feature in the synthetic dataset. In this plot, contributions above the baseline are highlighted in pink, while those below are shown in blue. The significant movement in the plot reflects the varying impact of Feature 0 across different instances, demonstrating its strong influence on the model's predictions. The contrasting pink and blue regions indicate how Feature 0 can either positively or negatively affect the predicted outcome, reinforcing its role as the primary driver of the model's behavior.

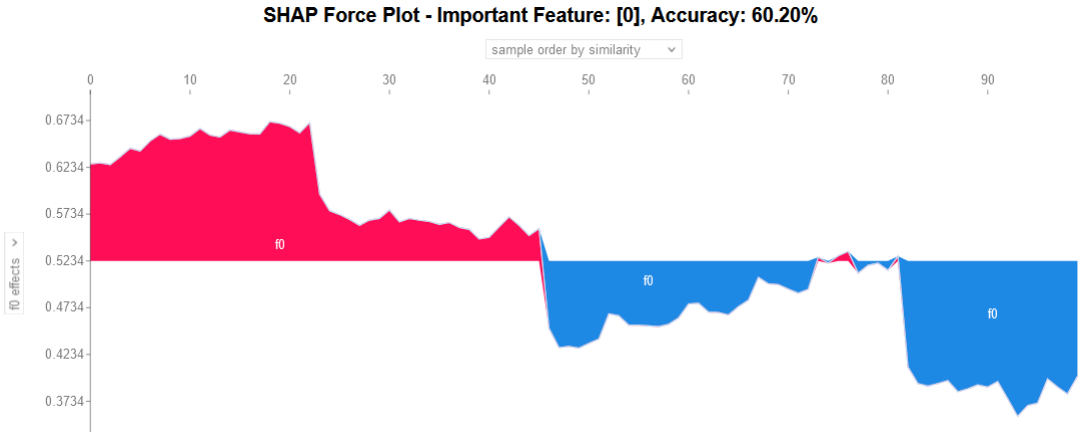


Figure 4.6: *SHAP force plot 1 for the Synthetic dataset with 8 features 1 important*

Figure 4.7 displays the SHAP force plot for Feature 6, a less important feature in the synthetic dataset. In this plot, contributions above the baseline are shown in pink, while those below are represented in blue. Unlike the force plot for more influential features, this graph exhibits minimal movement, indicating that Feature 6 has a limited impact on the model's predictions. The relatively flat distribution of pink and blue regions suggests that changes in Feature 6 do not significantly alter the predicted outcomes, highlighting its lower relevance compared to other features like Feature 0.

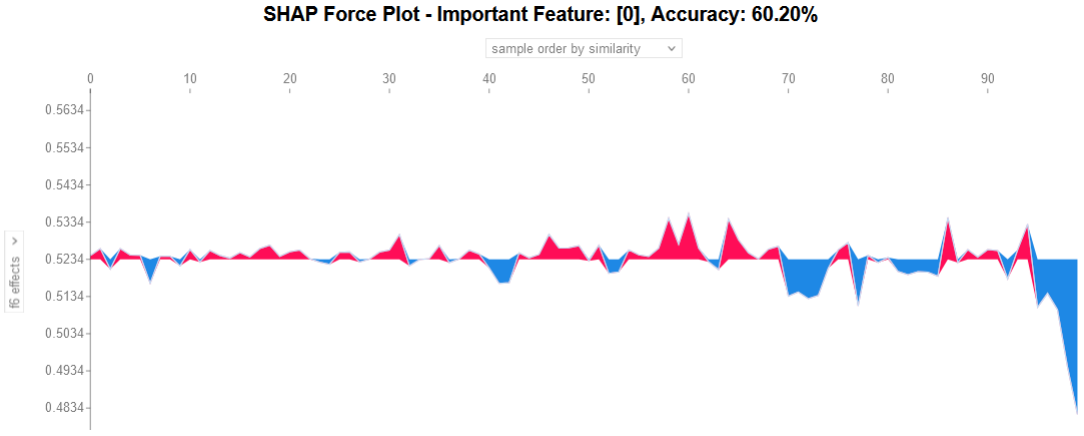


Figure 4.7: *SHAP force plot 2 for the Synthetic dataset with 8 features 1 important*

Figure 4.8 presents a SHAP force plot displaying the contributions of all features in the synthetic dataset. Feature 0 stands out with the highest base value, indicating its dominant role in influencing the model's predictions. The contributions of the other features are shown with smaller shifts from the baseline, reflecting their relatively lower impact on

the overall prediction. The visualization highlights how Feature 0 consistently has the greatest effect, while the remaining features provide more subtle adjustments, reinforcing the importance of Feature 0 in driving the model's decisions.

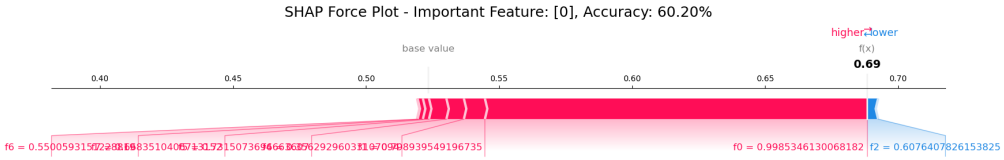


Figure 4.8: *SHAP force plot 3 for the Synthetic dataset with 8 features 1 important*

Figure 4.9 shows the SHAP waterfall plot for an instance in the synthetic dataset, displaying the contributions of all 8 features to the final prediction. Notably, Feature 0 has a score of -0.14, indicating a significant influence on the prediction, while all other features display zero contribution. This suggests that the model heavily relies on Feature 0 when making predictions for this instance, with the remaining features having no impact. The clear dominance of Feature 0 in this plot may indicate that the synthetic dataset lacks sufficient noise or complexity, resulting in a scenario where only one feature consistently drives the model's decisions. A more varied dataset might distribute the influence across multiple features, providing a more balanced contribution landscape.

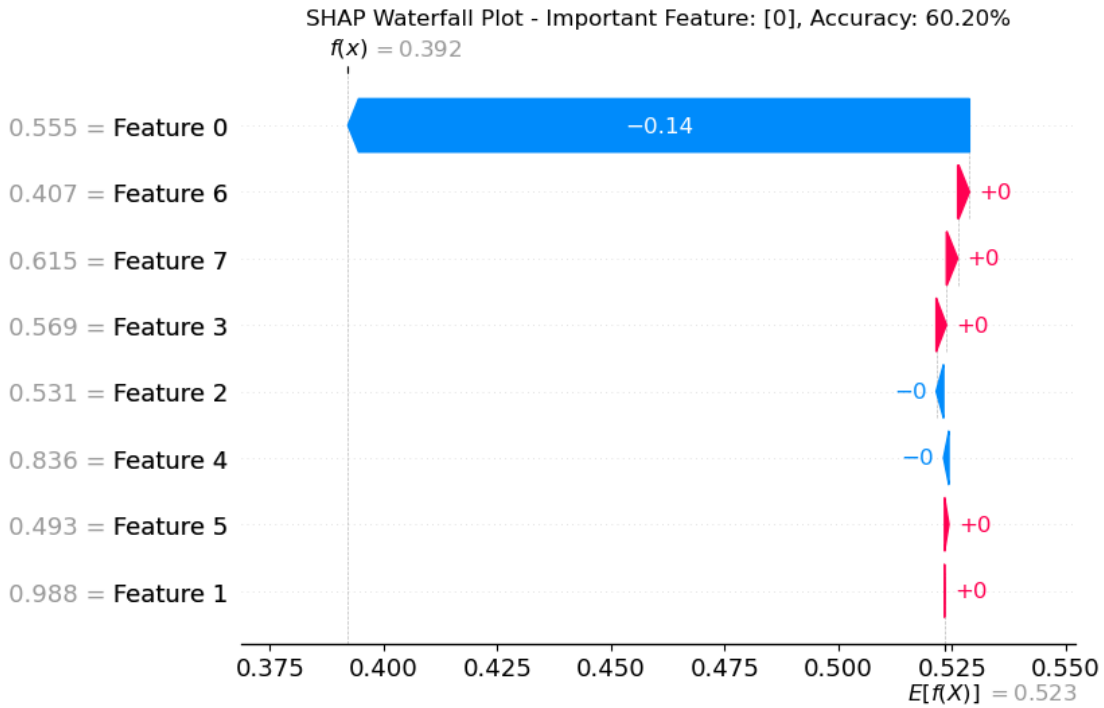
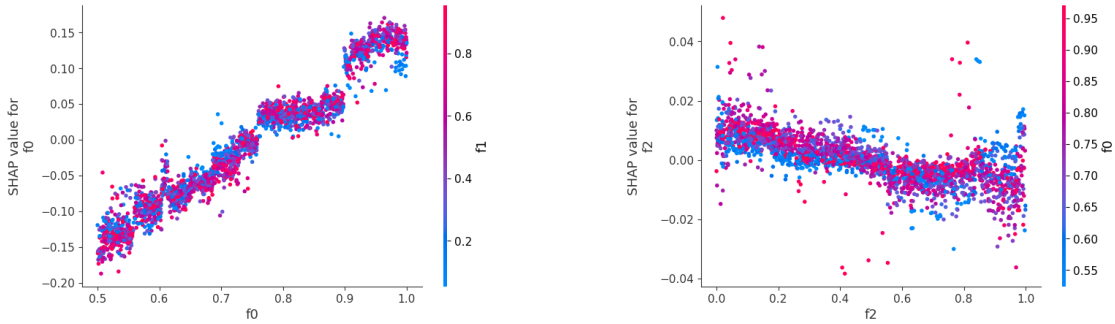


Figure 4.9: *SHAP waterfall plot for the Synthetic dataset with 8 features 1 important*

Figure 4.10 contains two subfigures illustrating SHAP dependence plots for Feature 0 and Feature 2 from the synthetic dataset. Subfigure 4.10a shows the dependence plot for Feature 0, which resembles an X-shaped function, with SHAP values ranging from -0.15 to 0.15. This pattern indicates that variations in Feature 0 significantly impact the model's

predictions, with both positive and negative contributions depending on the feature's values. In contrast, Subfigure 4.10b displays the dependence plot for Feature 2, which is nearly flat, resembling a $y = 0$ line, with SHAP values averaging between -0.02 and 0.02. This suggests that Feature 2 has minimal influence on the predictions, as changes in its value do not lead to substantial shifts in the model's output. The comparison between these plots highlights the strong impact of Feature 0 compared to the negligible effect of Feature 2 in the model's decision-making process.



(a) *Feature 0* (b) *Feature 2*

Figure 4.10: *SHAP dependence plot for the Synthetic dataset with 8 features 1 important*

4.1.3 Explainable Boosting Machine

Figure 4.11 shows the EBM feature weight plot, highlighting the influence of each feature on the model's predictions. Feature 0 has a weight above 0.4, making it the most significant contributor, while all other features have weights below 0.05, indicating minimal impact. This stark difference suggests that the model relies heavily on Feature 0, with the other features playing a much smaller role, likely due to the structure of the synthetic dataset.

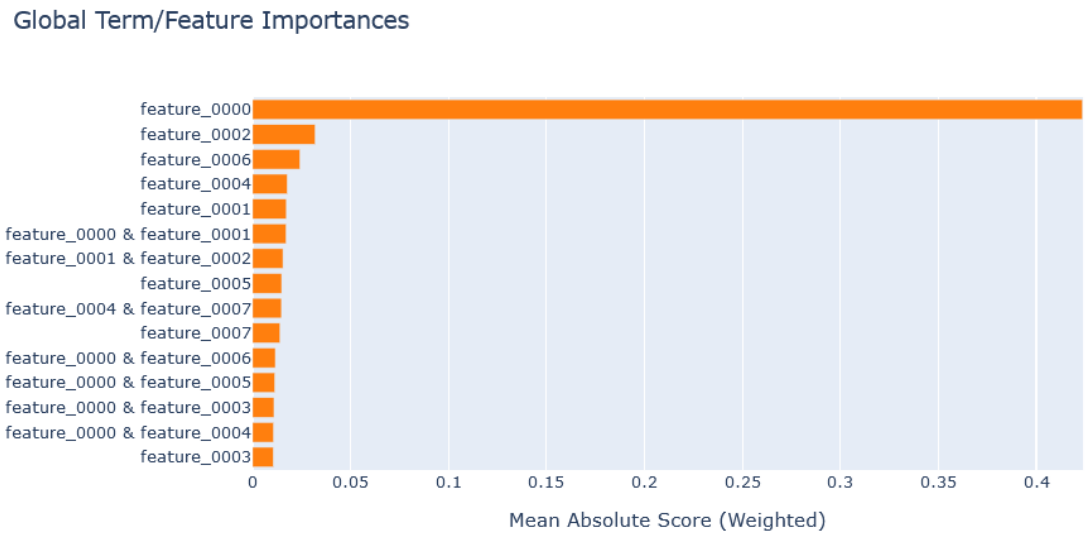


Figure 4.11: *EBM features for the Synthetic dataset with 8 features 1 important*

Figure 4.12 presents the score and density plots for Feature 0, the most important feature in the synthetic dataset. The score plot follows a linear pattern resembling a $y = x$ line, ranging from approximately -0.7 to 0.7, indicating a direct relationship between Feature 0's values and the model's predictions. The density plot, shown as vertical bars, reveals a high concentration of data points across the entire range of Feature 0, suggesting that the model frequently encounters these values when making predictions. This combination highlights both the strong influence of Feature 0 on the model and the broad representation of its values in the dataset.

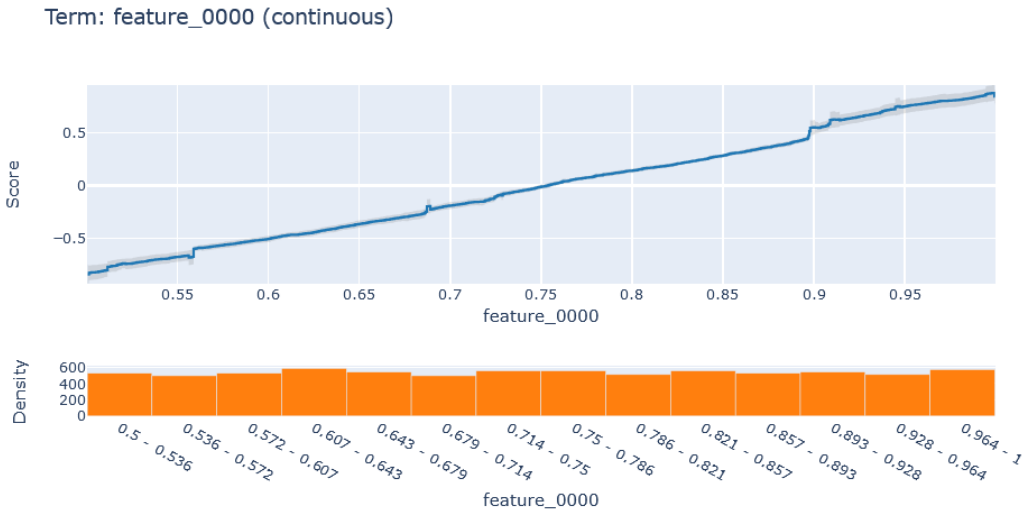


Figure 4.12: *EBM feature 0 for the Synthetic dataset with 8 features 1 important*

Figure 4.13 presents the score and density plots for Feature 1. The score plot resembles a flat $y = 0$ line with minimal variation, indicating that changes in Feature 1 have little to no impact on the model's predictions. In contrast, the density plot, displayed as vertical bars, shows a sparse distribution of data points, with low concentrations across the entire range of Feature 1. This suggests that not only is Feature 1 less influential in the model's decision-making, but it is also less represented in the dataset, further reducing its significance.

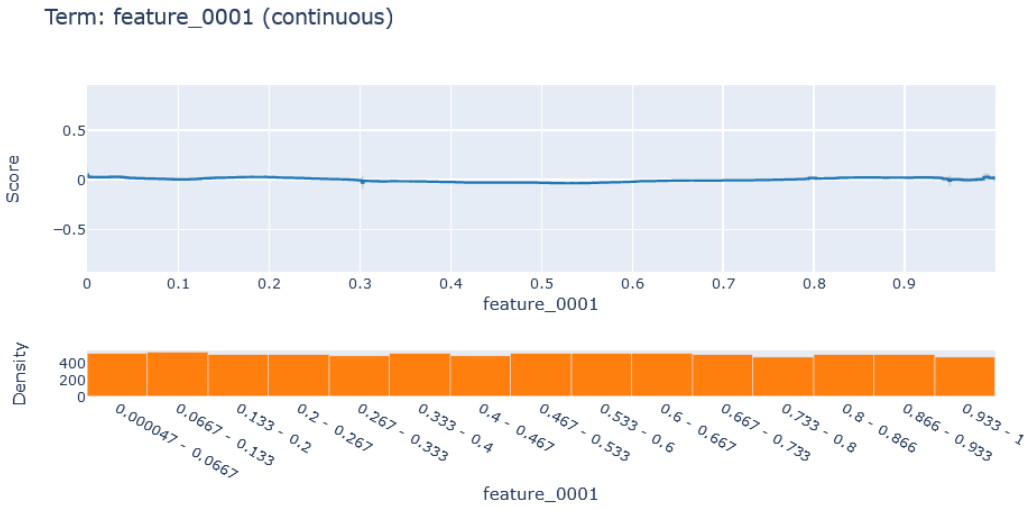


Figure 4.13: *EBM feature 1 for the Synthetic dataset with 8 features 1 important*

4.2 Eight features with three important feature

4.2.1 Decision Trees

Figure 4.14 shows the DT generated using the `ClassificationTree` from the `interpret` library, applied to the synthetic dataset with 8 features, where 3 features (Feature 0, 6, and 7) are particularly important. The tree begins with a split on Feature 7 at the root node, suggesting that this feature offers the best initial separation of data points. This choice indicates that Feature 7 helps the model differentiate instances early in the process, with subsequent splits involving Features 0 and 6 refining the decisions. The node weights represent the number of data points following each branch, providing insight into how the model utilizes feature importance for prediction.

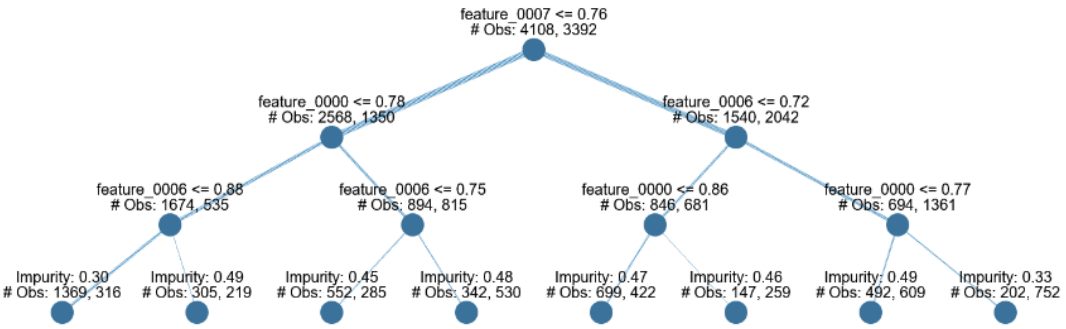


Figure 4.14: *Decision Tree for the Synthetic dataset with 8 features 3 important*

Figure 4.15 highlights the path taken by the model to correctly classify an instance with the label `False`. The path is marked in orange, beginning with Feature 7 as the initial split, indicating its crucial role in separating the data early on. As the path progresses, Features 0 and 6 contribute to further refining the decision. The node weights along this path show the distribution of data points through each decision, helping to understand the classification process.

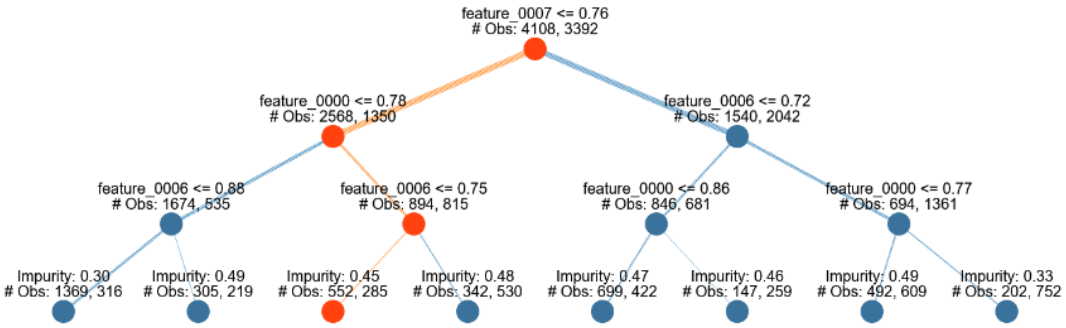


Figure 4.15: *Decision Tree for the Synthetic dataset, highlighting label False*

Figure 4.16 illustrates a path that led to a correct classification with the label `True`. The path, shown in orange, starts with a split on Feature 7, emphasizing its importance in the initial decision-making. Subsequent splits by Features 0 and 6 provide additional refinement. The weights at each decision point reveal how the model processes data through this path, offering a clear view of the contributions of the key features to the final prediction.

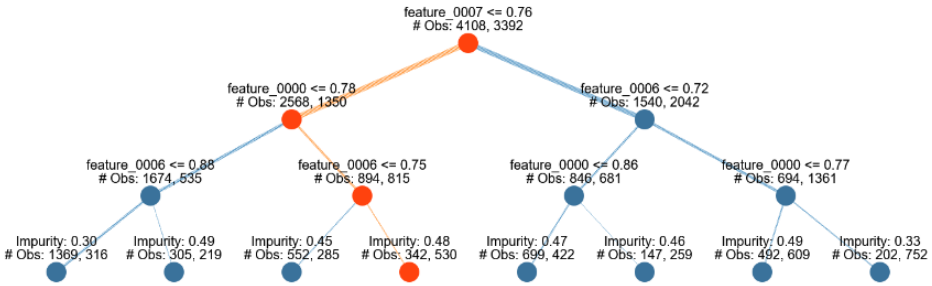


Figure 4.16: *Decision Tree for the Synthetic dataset, highlighting label True*

4.2.2 SHapley

Figure 4.17 displays the SHAP bar plot, illustrating the contribution of each feature to the model's predictions. Feature 0 has the highest SHAP value, indicating its dominance as a key feature in the synthetic dataset. Features 6 and 7 also contribute to the model's predictions but with notably lower SHAP values compared to Feature 0. The remaining features have minimal impact, as reflected in their smaller SHAP values. This plot ranks feature importance, highlighting how Feature 0, followed by Features 6 and 7, drives the model's decision-making process.

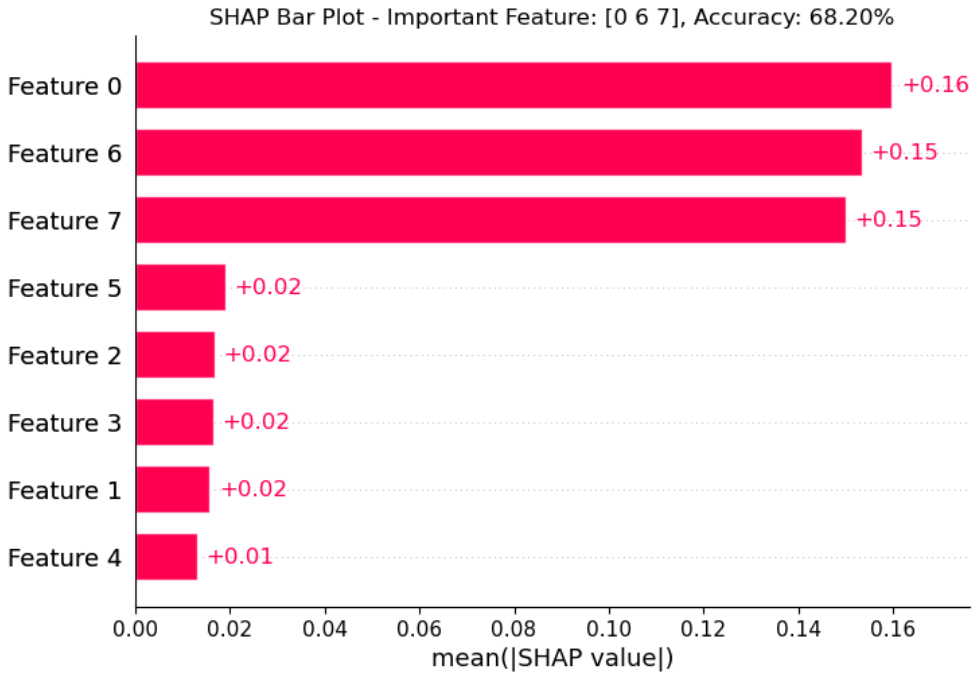


Figure 4.17: *SHAP bar plot for the Synthetic dataset with 8 features 3 important*

Figure 4.18 presents the SHAP beeswarm plot, showing the spread of SHAP values for each feature across all instances in the synthetic dataset. Feature 0 has the broadest range of SHAP values, underscoring its substantial influence on the model's predictions. Features 6 and 7 exhibit smaller but still notable contributions, while the other features show narrower SHAP value distributions, indicating lesser impact. The plot also reveals a degree of noise, added to simulate realistic conditions. This noise reflects the complexity of interactions among features and the variation in their contributions across instances, making the interpretability of the model more nuanced.

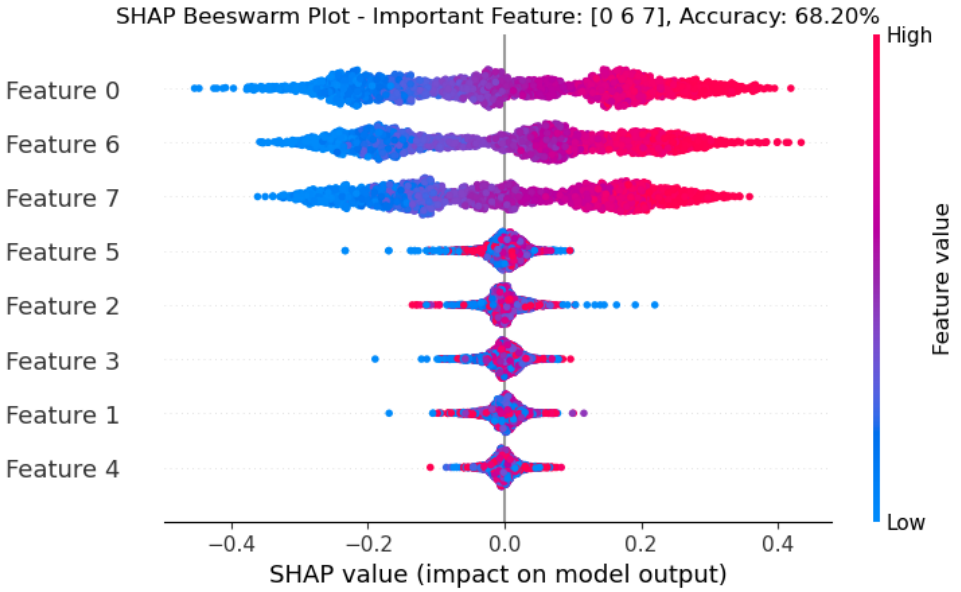


Figure 4.18: *SHAP beeswarm plot for the Synthetic dataset with 8 features 3 important*

Figure 4.19 illustrates the SHAP force plot for Feature 6, one of the key features in the synthetic dataset. The plot shows substantial movement, with contributions above the baseline highlighted in pink and those below in blue, indicating how variations in Feature 6 strongly influence the model's predictions. The dynamic shifts in the SHAP values across different instances emphasize the importance of Feature 6 in shaping the prediction outcomes, as changes in this feature can both positively and negatively impact the model's decisions. This visual representation underscores the role of Feature 6 as a critical factor in the model's decision-making process.

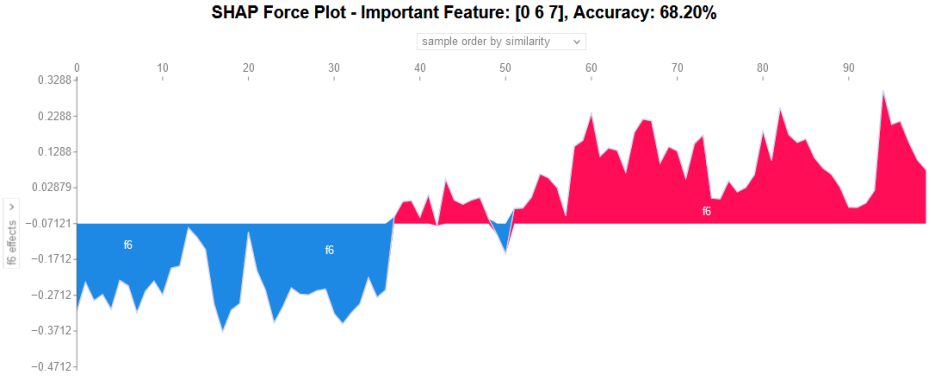


Figure 4.19: *SHAP force plot 1 for the Synthetic dataset with 8 features 3 important*

Figure 4.20 displays the SHAP force plot for Feature 0, highlighting its strong influence on the model's predictions in the synthetic dataset. The plot shows marked shifts with significant movement, with positive contributions in pink and negative ones in blue, demonstrating how Feature 0 substantially affects prediction outcomes. Its impact is evident as changes in its values consistently lead to significant adjustments in the model's decision process, reflecting its role as a primary driver of prediction variability.

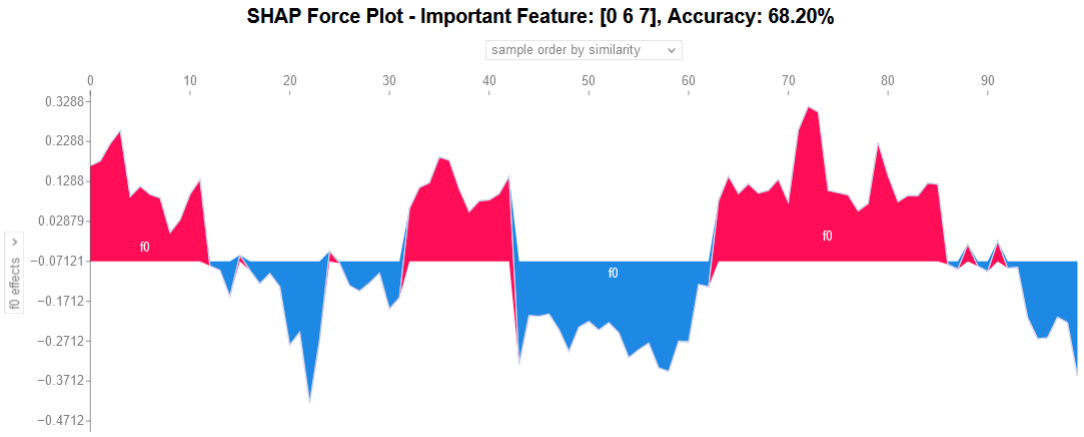


Figure 4.20: *SHAP force plot 2 for the Synthetic dataset with 8 features 3 important*

Figure 4.21 presents the SHAP force plot for Feature 7, another influential feature in the dataset. While the movement in the plot is less pronounced than for Feature 0, it still shows notable contributions above and below the baseline. The pink and blue regions indicate how Feature 7's variations alter the predictions, with a more balanced distribution between positive and negative impacts. This suggests that while Feature 7 is crucial, its influence is more context-dependent, contributing to nuanced adjustments in the model's output.

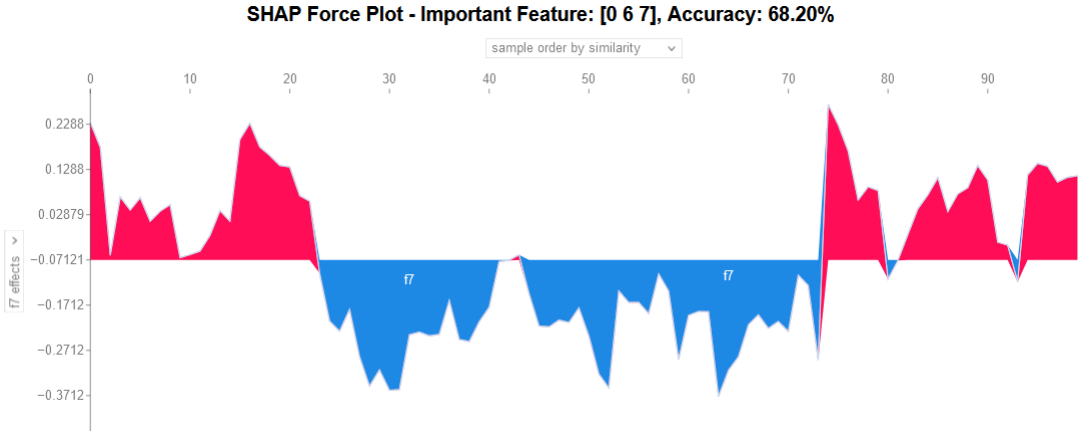


Figure 4.21: *SHAP force plot 3 for the Synthetic dataset with 8 features 3 important*

Figure 4.22 displays the SHAP force plot for Feature 1, which shows a nearly flat line with minimal variation. Unlike the more influential features, the contributions of Feature 1 remain close to zero, with only slight shifts above and below the baseline. This flatness indicates that changes in Feature 1 have little impact on the model's predictions, contributing minimally to the decision-making process. Its limited influence suggests that Feature 1 plays a relatively insignificant role in shaping the model's output.

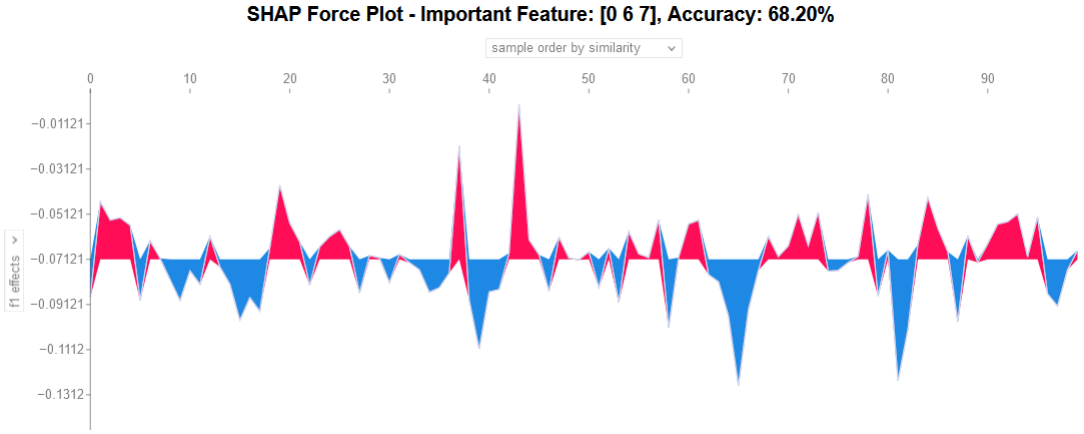


Figure 4.22: *SHAP force plot 4 for the Synthetic dataset with 8 features 3 important*

Figure 4.23 shows a SHAP force plot displaying the combined contributions of all features in the synthetic dataset. Feature 7 stands out with the highest base value, reflecting its dominant role in influencing the model's predictions. Feature 6 also contributes significantly, albeit to a lesser extent than Feature 7, while Feature 0 shows a moderate impact. The remaining features have minimal contributions, with their SHAP values remaining close to zero, indicating a negligible effect on the overall predictions. This visualization emphasizes the primary influence of Features 0, 6, and 7, while the other features play a minor role in shaping the model's output.

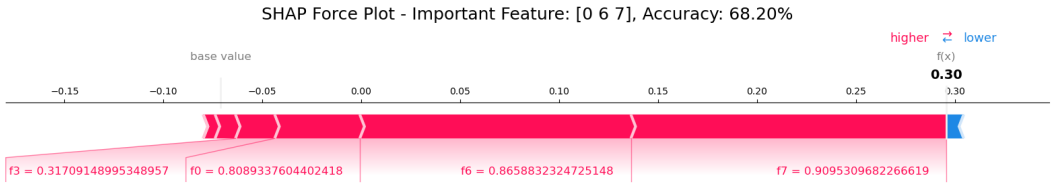


Figure 4.23: *SHAP force plot 5 for the Synthetic dataset with 8 features 3 important*

Figure 4.24 presents the SHAP waterfall plot, summarizing the contributions of all features in the synthetic dataset to a specific prediction. Feature 0 has the largest impact, with a substantial contribution that shifts the base value significantly, reflecting its dominant role in the model's decision. Feature 6 also contributes notably, though less than Feature 0, while Feature 7 adds a moderate adjustment. The remaining features have nearly zero contributions, indicating that their influence on the prediction is minimal. This waterfall plot highlights the cumulative effect of each feature, showing how Features

0, 6, and 7 play key roles in determining the model's output, while the other features have a negligible impact.

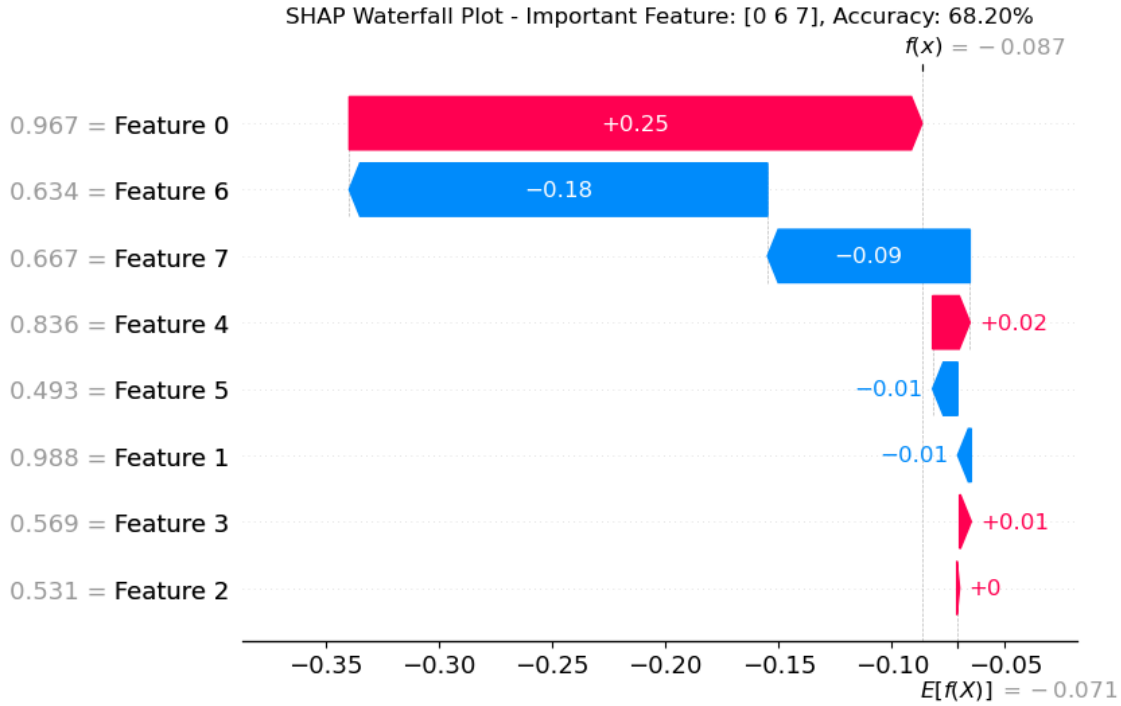
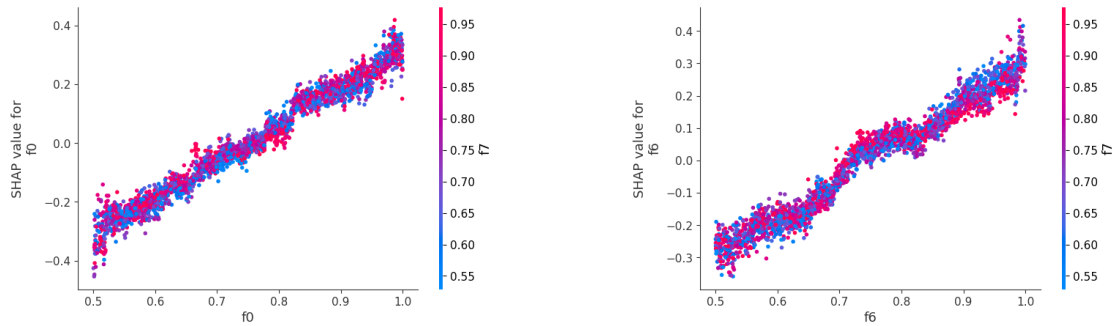


Figure 4.24: *SHAP waterfall plot for the Synthetic dataset with 8 features 3 important*

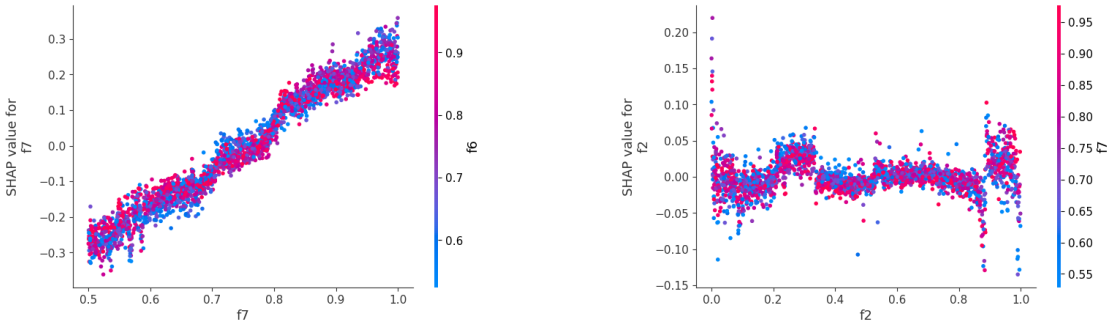
Figure 4.25 contains two subfigures, illustrating SHAP dependence plots for Feature 0 and Feature 6. The plot for Feature 0, shown in Subfigure 4.25a, exhibits a linear pattern, with SHAP values ranging from -0.4 to 0.4, indicating a consistent relationship between changes in Feature 0 and its impact on the model's predictions. In contrast, Subfigure 4.25b shows the dependence plot for Feature 6, which also follows a generally linear trend but includes more variability, with SHAP values ranging from -0.3 to 0.4. The slight noise in Feature 6's plot suggests a more complex relationship with the prediction outcomes compared to the smoother influence of Feature 0. Together, these plots demonstrate how both features significantly impact the model's behavior, with Feature 0 showing a clearer pattern of influence and Feature 6 displaying a more nuanced effect.



(a) *Feature 0* (b) *Feature 6*

Figure 4.25: *SHAP dependence plot 1 for the Synthetic dataset with 8 features 3 important*

Figure 4.26 contains two subfigures, illustrating SHAP dependence plots for Feature 7 and Feature 2. Subfigure 4.26a displays the plot for Feature 7, which follows a roughly linear pattern similar to an x function, with SHAP values ranging from -0.3 to 0.3. The plot includes some noise, indicating slight variability in how changes in Feature 7 affect the model's predictions. Subfigure 4.26b shows the dependence plot for Feature 2, which is nearly flat, reflecting a minimal relationship between Feature 2 and the model's outputs. The flatness of Feature 2's plot suggests that changes in this feature have little impact on prediction outcomes. Together, these plots demonstrate that while Feature 7 has a more dynamic influence on the model, Feature 2 contributes very little to the predictive process.



(a) *Feature 7* (b) *Feature 2*

Figure 4.26: *SHAP dependence plot 2 for the Synthetic dataset with 8 features 3 important*

4.2.3 Explainable Boosting Machines

Figure 4.27 displays the EBM feature weight plot, illustrating the influence of each feature on the model's predictions. Features 0, 6, and 7 each have weights above 0.4, highlighting their significant contributions, while the remaining features have weights close to zero, indicating minimal impact. This distribution shows that the model relies heavily on these three important features, reflecting their importance in the synthetic dataset.

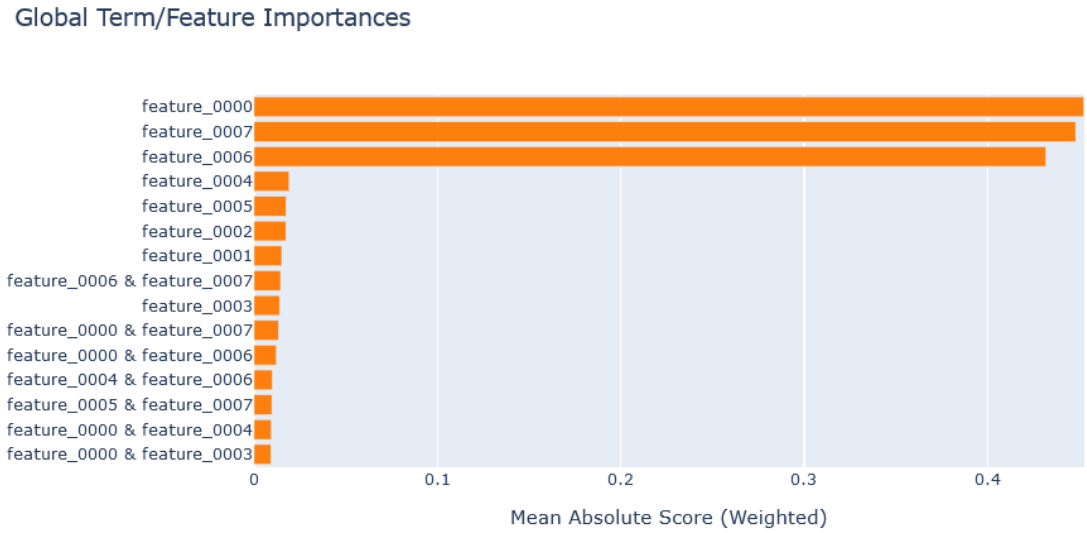


Figure 4.27: *EBM features for the Synthetic dataset with 8 features 3 important*

Figure 4.28 presents the score and density plots for Feature 0, the most important feature in the synthetic dataset. The score plot follows a linear $y = x$ pattern, ranging from approximately -1 to 0.7, indicating a direct relationship between Feature 0's values and the model's predictions. The density plot, displayed as vertical bars, shows a high concentration of data points across the entire range of Feature 0, suggesting that the model frequently encounters these values when making predictions. This combination highlights both the strong influence of Feature 0 on the model's behavior and the broad representation of its values in the dataset.

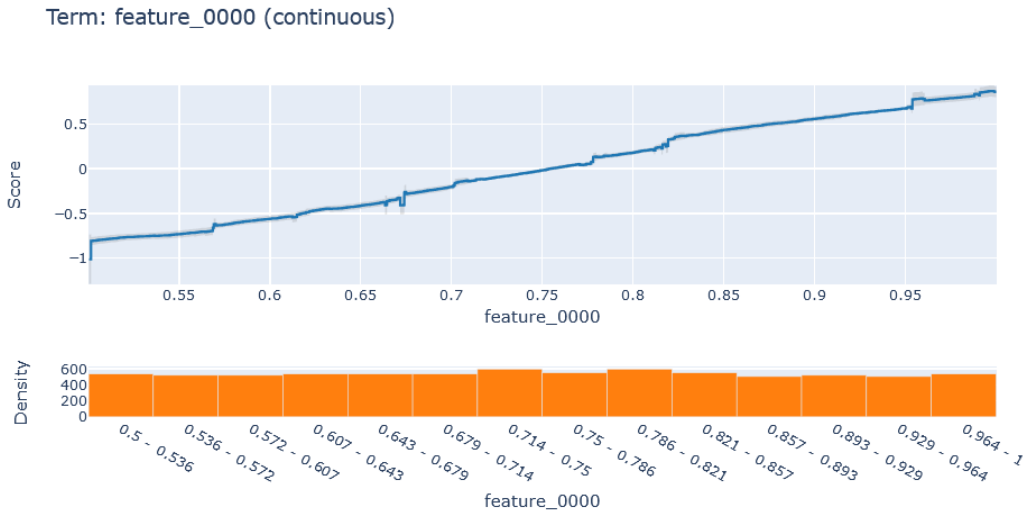


Figure 4.28: *EBM feature 0 for the Synthetic dataset with 8 features 3 important*

Figure 4.29 displays the score and density plots for Feature 7, another key feature in the synthetic dataset. The score plot exhibits a near-linear trend, ranging from -0.7 to 0.7, showing how variations in Feature 7 directly influence the model's output. The density plot reveals a high concentration of data points throughout the range, indicating that Feature 7's values are well-represented in the dataset. This suggests that Feature 7 plays a consistent role in shaping the model's predictions, complementing the influence of other important features like Feature 0.

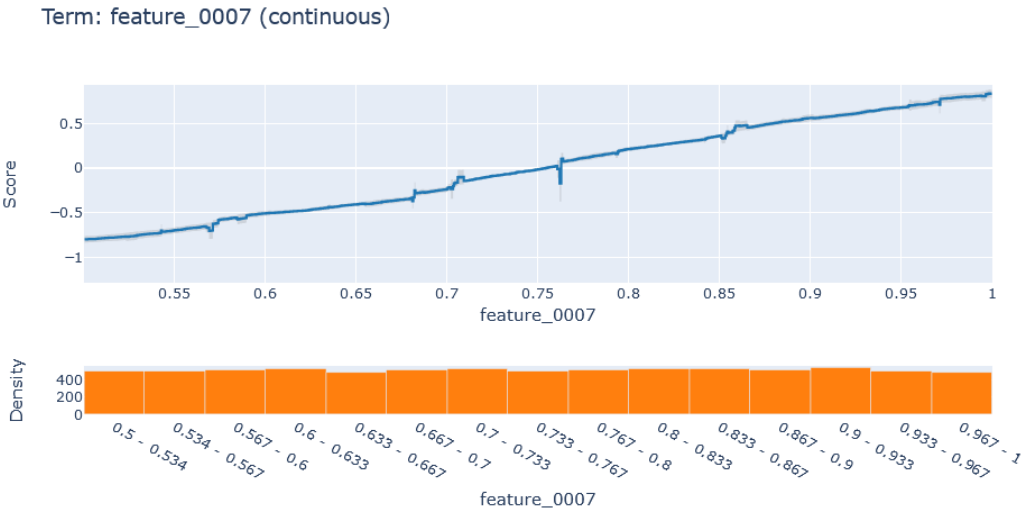


Figure 4.29: *EBM feature 7 for the Synthetic dataset with 8 features 3 important*

Figure 4.30 illustrates the score and density plots for Feature 6, a crucial feature in the synthetic dataset. The score plot follows a linear trajectory, extending from -0.7 to 0.7, reflecting Feature 6's direct effect on model predictions. The density plot indicates a uniform spread of data points across the feature's range, showing that the model frequently encounters a broad range of values for Feature 6. This consistent data distribution underscores Feature 6's role in the model, contributing steadily to its predictive behavior alongside other key features.

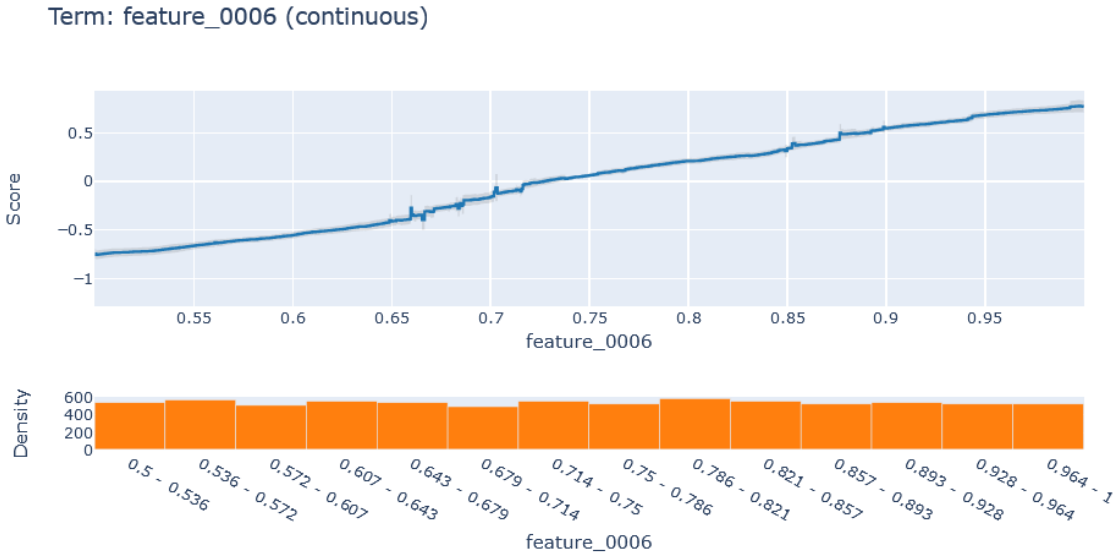


Figure 4.30: *EBM feature 6 for the Synthetic dataset with 8 features 3 important*

Figure 4.31 presents the score and density plots for Feature 3. The score plot is nearly flat, centered around zero with minimal variation, indicating that changes in Feature 3 have little to no effect on the model's predictions. The density plot reveals a sparse distribution, with low concentrations of data points across the feature's range. This suggests that Feature 3 is infrequently encountered by the model, further diminishing its influence compared to more impactful features like Feature 0, 6, and 7.

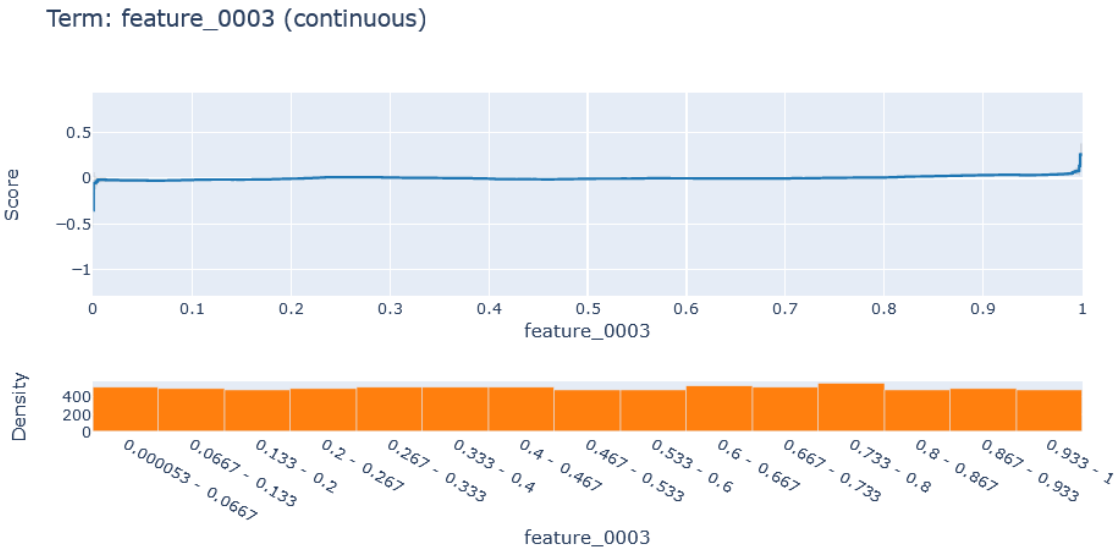
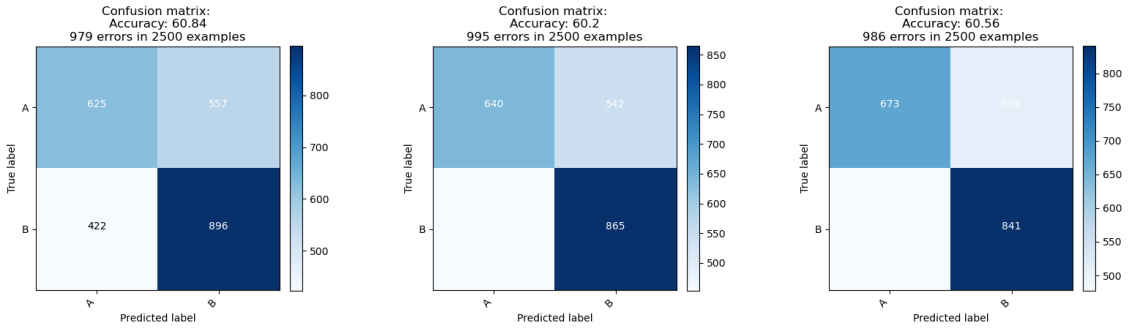


Figure 4.31: *EBM features 3 for the Synthetic dataset with 8 features 3 important*

4.3 Discussion

In this final analysis of the synthetic dataset, Figure 4.32 and Figure 4.33 presents three Confusion Matrix (CM), each corresponding to a different model used throughout the study: a DT, SHAP, and EBM. While the performance metrics across the models appear similar, the primary focus of this analysis is not on their scores but rather on the interpretability and explainability provided by each approach.

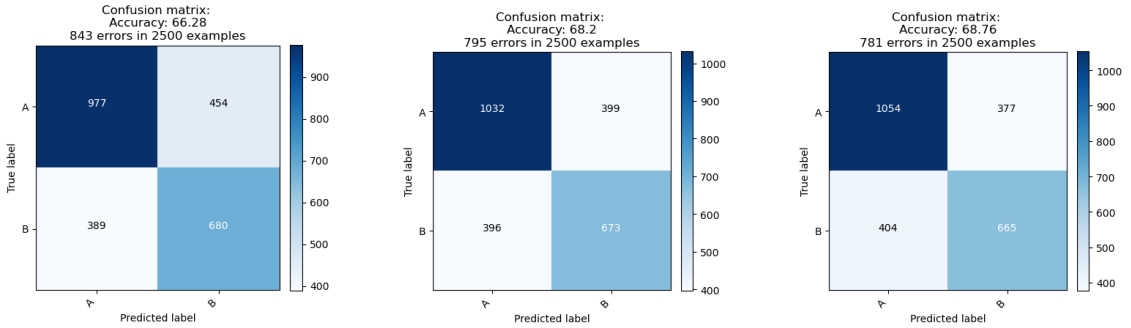
4.3.1 Eight features with one important feature



(a) *DT* (b) *SHAP* (c) *EBM*

Figure 4.32: *Confusion Matrix for the Synthetic dataset with 8 features 1 important*

4.3.2 Eight features with three important feature



(a) *DT* (b) *SHAP* (c) *EBM*

Figure 4.33: *Confusion Matrix for the Synthetic dataset with 8 features 3 important*

4.3.3 Key Takeaways

The DT model offered a straightforward visualization of how the synthetic dataset's features influenced predictions. Relevant features consistently emerged as key determinants in decision paths and splits, aligning with insights from SHAP visualizations. The bar plot, beeswarm plot, and force plots all underscored the influence of important features, distinguishing them with higher SHAP values compared to less relevant features.

Similarly, EBM reinforced these findings through its feature weight analysis, where relevant features received substantially higher weights than others, further confirming their primary role in the model's decision-making process. The EBM visualizations provided

clear, additive contributions, showing how relevant features directly influenced the model's outcomes, while irrelevant features played minimal roles.

Overall, despite using different classification methods, all three models consistently identified key features as primary influencers in the predictions. This convergence across different interpretability tools underscores the robustness of the findings and demonstrates the effectiveness of these methods in uncovering the underlying patterns in the synthetic dataset. By focusing on interpretability rather than solely on performance metrics, the study provides a comprehensive understanding of how each algorithm highlights the contributions of both relevant and irrelevant features, enhancing confidence in the models' decision-making processes.

5 Real-world Datasets

5.1 Iris dataset

In this chapter, we present the results obtained from applying the DT, SHAP and EBM methods to the Iris dataset. The results include performance metrics such as accuracy, alongside DT generated visualizations to aid in interpreting feature importance and interactions. SHAP plots are used to explain the impact of individual features, such as petal length and petal width, on the model's predictions. EBM's transparent modeling further provides insights into how these features contribute to the classification of the different iris species, ensuring that the decision-making process remains interpretable and aligned with domain knowledge.

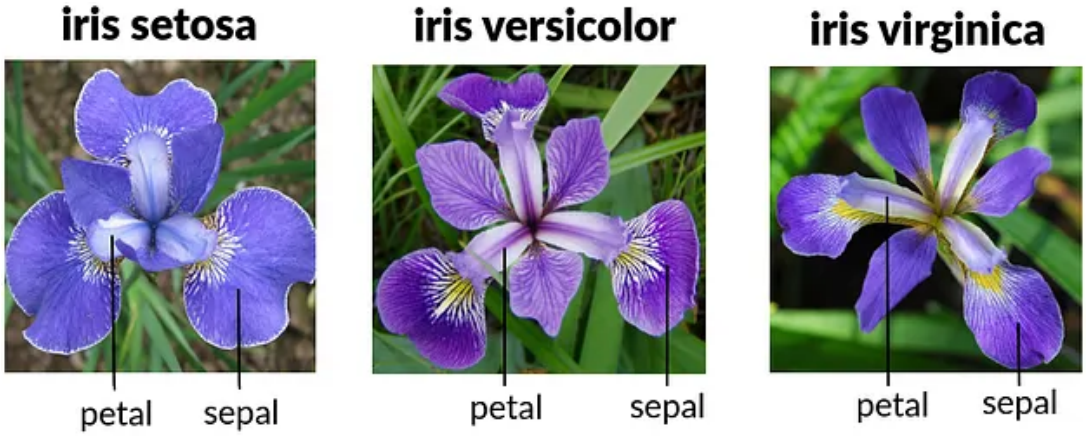


Figure 5.1: *Example Images of Each Iris Species in the Dataset*

5.1.1 Decision Trees

Figure 5.2 shows the DT generated using the `ClassificationTree` from the `interpret` library, applied to the Iris dataset. The tree's initial split is based on petal width, a choice that reflects its critical role in differentiating between species, particularly in distinguishing `Iris setosa` from the others. Although petal length is often considered the most important feature overall, the selection of petal width as the first node suggests that it provides a strong initial separation between species. Subsequent splits don't involve sepal length and

sepal width, using only petal length to refine the model's predictions. The node weights represent the number of data points that follow each branch, offering insights into how the model leverages feature importance to make accurate classifications.

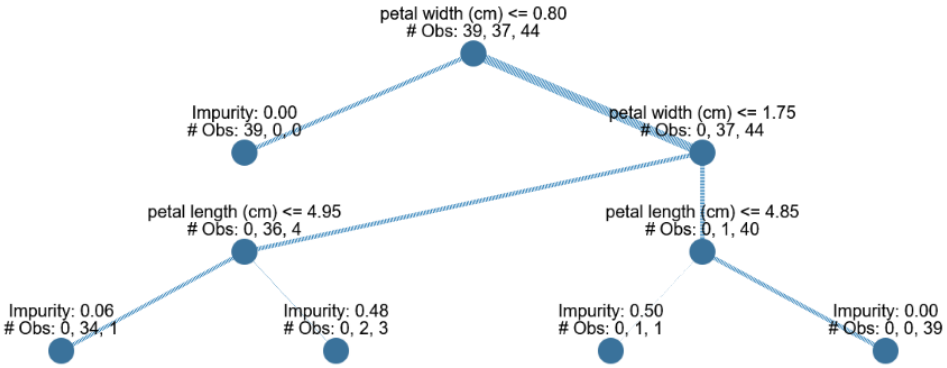


Figure 5.2: *Decision Tree for the Iris dataset*

Figure 5.3 illustrates, using the `ClassificationTree` from the `interpret` library, the DT generated applied to the Iris dataset, highlighting a path that led to a correct classification of an instance with the label `0`. The path, shown in orange, begins with a split on petal width, which is critical for differentiating `Iris setosa` from the other species. Subsequent splits involve petal length, refining the classification decision. The highlighted path reflects the sequence of decisions made by the model, with the orange segments representing the node weights that followed each decision. This visualization helps to understand how the model relies on key features to accurately identify instances of `Iris setosa`.

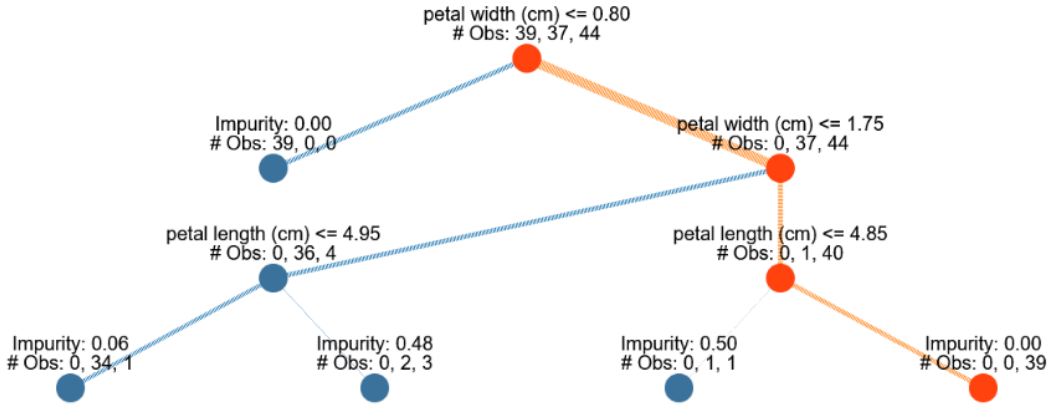


Figure 5.3: *Decision Tree for the Iris dataset highlighting label Iris setosa*

Figure 5.4 shows the DT from the `ClassificationTree` applied to the Iris dataset, focusing on a path that led to a correct classification of an instance with the label `1`. The path, highlighted in orange, starts with petal width at the root node, providing the initial distinction between species. The model then uses further splits based on petal length to differentiate `Iris versicolor` from the other species. The orange highlights indicate the number of data points processed at each decision node along the path, offering insight into the model's reasoning for classifying this instance correctly.

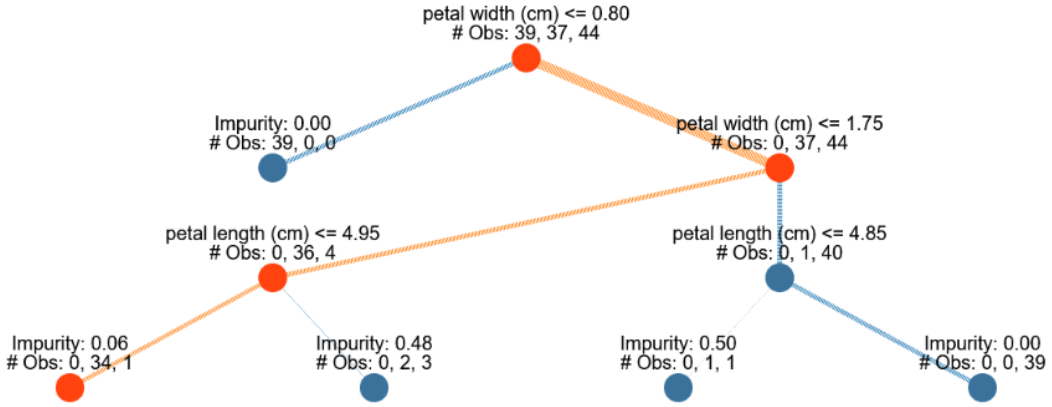


Figure 5.4: *Decision Tree for the Iris dataset highlighting label Iris versicolor*

Figure 5.5 depicts the DT generated using the `ClassificationTree` from the `interpret` library, applied to the Iris dataset, with a focus on a path that resulted in the correct classification of an instance with the label `2`. The orange path starts with a split on petal width, which helps in the initial separation of species. As the decision-making progresses, the model incorporates splits based on petal length characteristics to accurately classify `Iris virginica`. The orange segments along the path represent the node weights, providing an understanding of how many data points followed this sequence of decisions. This graphical representation showcases how the model navigates through key features to distinguish `Iris virginica` from other species.

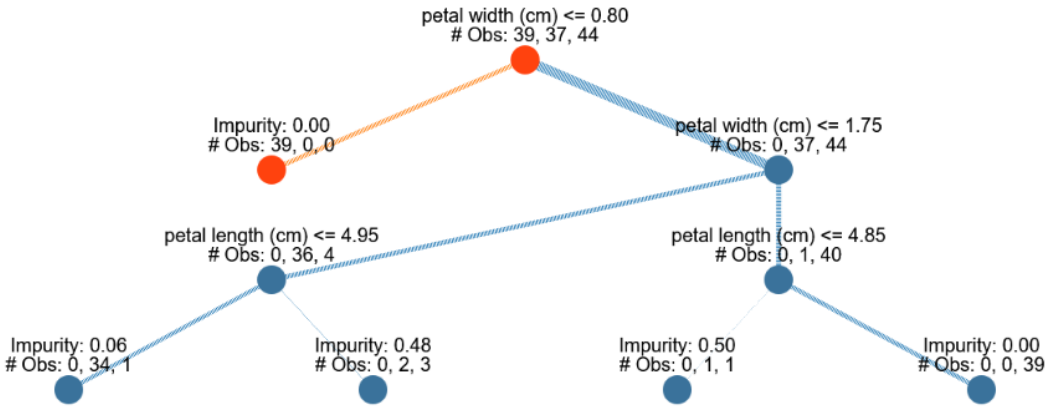


Figure 5.5: *Decision Tree for the Iris dataset highlighting label Iris virginica*

5.1.2 SHapley

Figure 5.6 displays the SHAP bar plot for the Iris dataset, illustrating the average contribution of each feature to the model's predictions. Petal length has the highest SHAP value, indicating that it is the most influential feature in determining the classification of iris species. Petal width follows closely, also contributing significantly to the model's decisions. The contributions of sepal length and sepal width are notably smaller, reflecting their lesser impact on the classification process. This plot provides a clear ranking of feature importance, with petal measurements playing a dominant role in distinguishing between the three species.

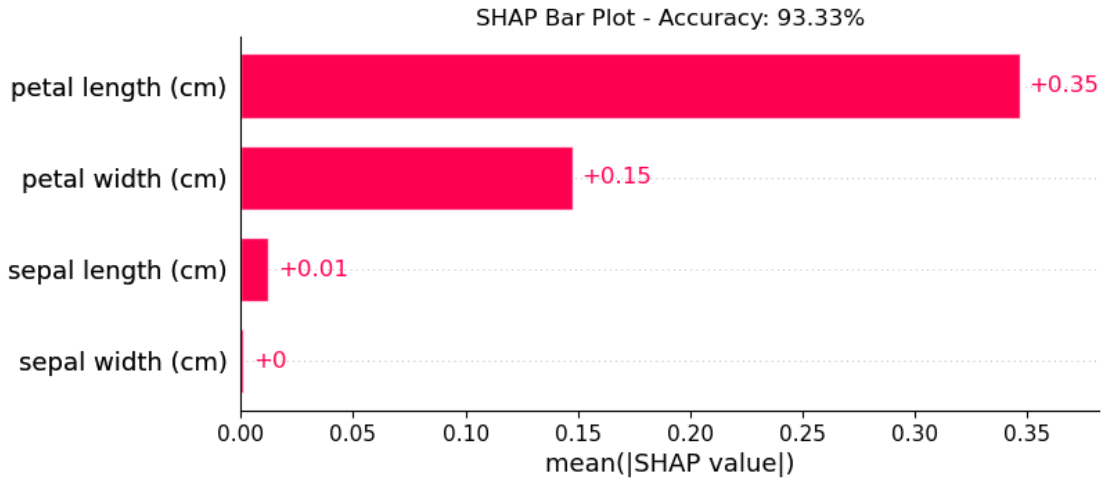


Figure 5.6: *SHAP barplot for the Iris dataset*

Figure 5.7 presents the SHAP beeswarm plot for the Iris dataset, showing the distribution of SHAP values for each feature across different instances. The plot includes a few distinct lines, each representing a feature, with dots along the lines indicating SHAP values for individual data points. Petal length and petal width stand out, with a wider spread of dots, reflecting their greater variability and influence on the model's predictions. In contrast, sepal length and sepal width show a narrower distribution, with fewer dots clustered around zero, indicating their smaller impact on the model. This visualization highlights how petal measurements play a crucial role in classification, while sepal measurements contribute less significantly.

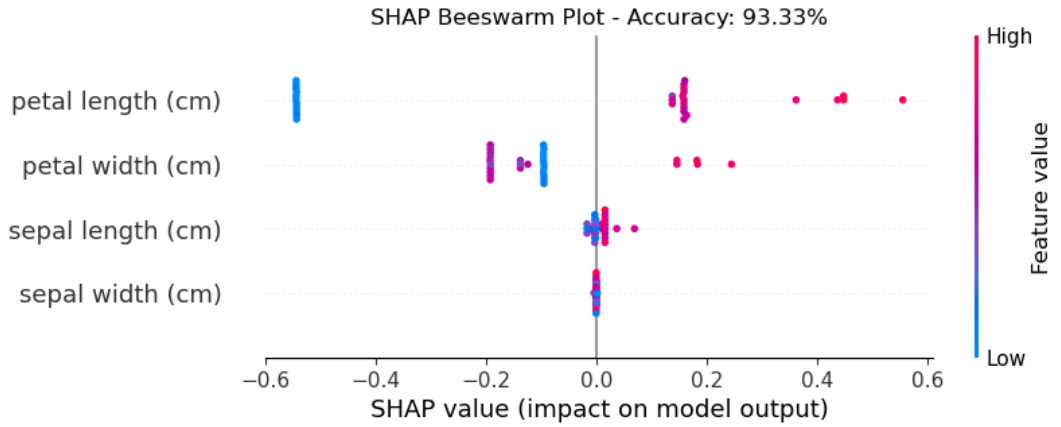


Figure 5.7: *SHAP beeswarm for the Iris dataset*

Figure 5.8 displays the SHAP force plot for petal length. This plot highlights the substantial impact of petal length on the model's predictions, with significant movements both above and below the baseline. Positive contributions, shown in pink, indicate instances where longer petal lengths increase the likelihood of predicting species like `Iris virginica`, while negative contributions in blue suggest shorter petal lengths that align with predictions for `Iris setosa`. The dynamic shifts in this plot underscore petal length's role as a primary driver in differentiating between the species.

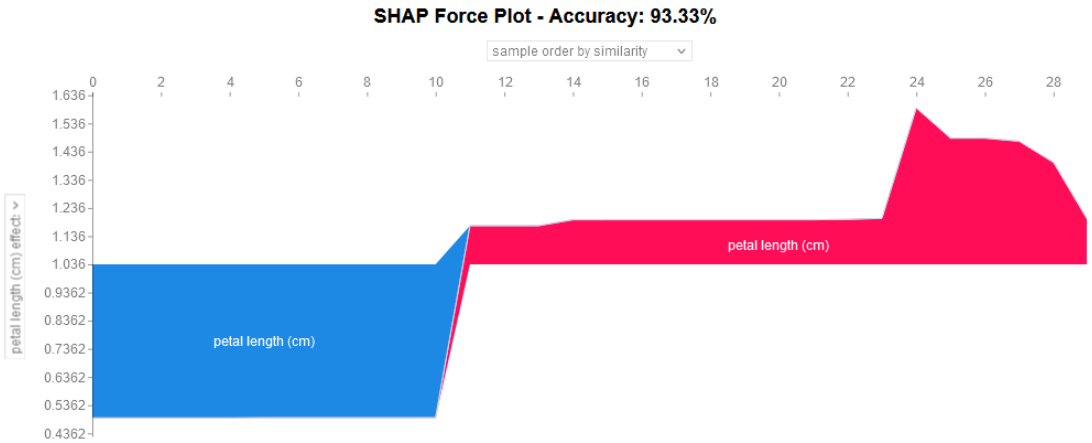


Figure 5.8: *SHAP force plot 1 for the Iris dataset*

Figure 5.9 shows the SHAP force plot for petal width, another key feature in the Iris dataset. The plot reveals significant variation in SHAP values, with contributions displayed in both pink and blue. Higher petal width values positively influence the prediction of species like `Iris virginica`, while lower values contribute negatively, favoring classifications such as `Iris versicolor` or `Iris setosa`. The balance between positive and negative contributions reflects how petal width helps refine the model's predictions, complementing the influence of petal length.

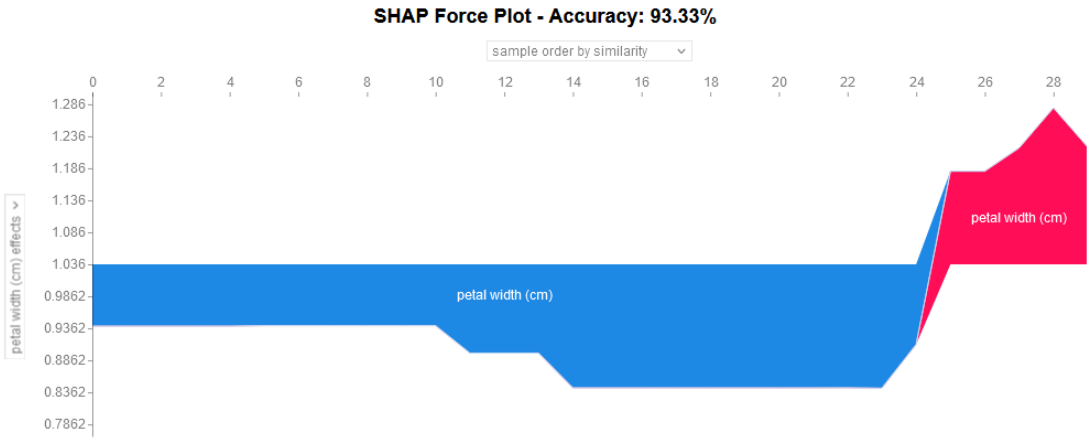


Figure 5.9: *SHAP force plot 2 for the Iris dataset*

Figure 5.10 presents the SHAP force plot for sepal length, showing its more moderate impact on the Iris dataset's model predictions. The movements above and below the baseline are less pronounced compared to petal measurements, indicating that sepal length plays a secondary role in classification. Positive contributions, highlighted in pink, correspond to instances where longer sepal lengths contribute to predictions of species like `Iris virginica`, while shorter sepal lengths in blue support predictions for species such as `Iris setosa`. While not as dominant as petal length, sepal length adds nuance to the overall model behavior.

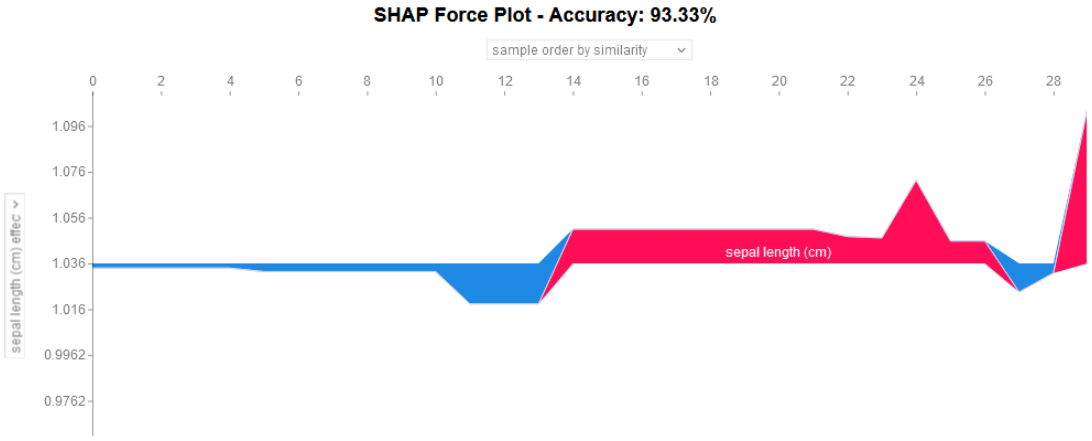


Figure 5.10: *SHAP force plot 3 for the Iris dataset*

Figure 5.11 illustrates the SHAP force plot for sepal width, which shows minimal variation in SHAP values compared to other features in the Iris dataset. The plot is relatively flat, with subtle contributions below the baseline, indicating that changes in sepal width have a limited influence on the model's predictions. Positive contributions in pink and negative contributions in blue are concentrated close to the baseline, suggesting that sepal width is not a major differentiator among the three species. Its role is complementary, providing some additional context but not driving the primary decisions of the model.

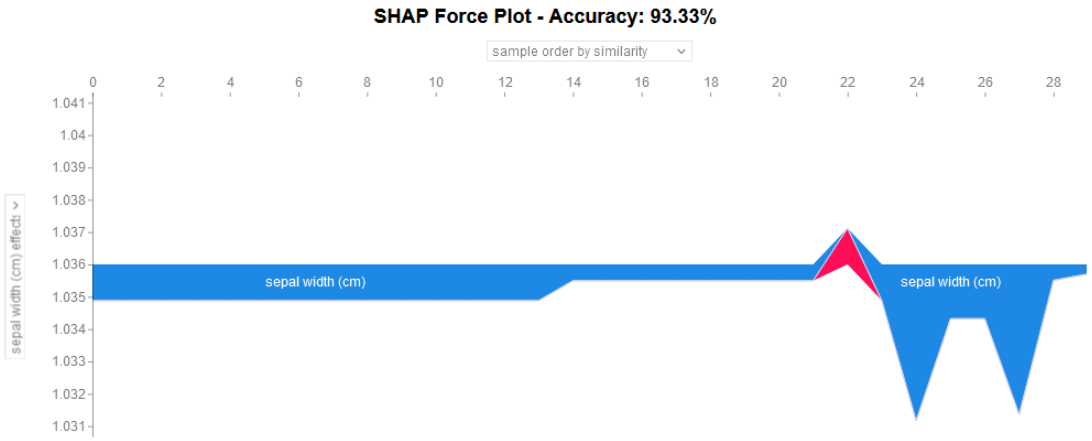


Figure 5.11: *SHAP force plot 4 for the Iris dataset*

Figure 5.12 presents the SHAP force plot for the Iris dataset, showing the combined contributions of all features to a specific prediction. The plot reveals that the decision is primarily driven by petal length, with positive values that impact the model's decision, indicating its dominant role in the classification. Petal width also contributes to the decision, though to a lesser extent, providing additional adjustments to the prediction outcome. In contrast, sepal length and sepal width have minimal impact, with their SHAP values remaining close to zero, indicating that they play a negligible role in this particular instance. This visualization underscores how the model relies heavily on petal measurements, especially petal length, to differentiate between species, while sepal features contribute only marginally.

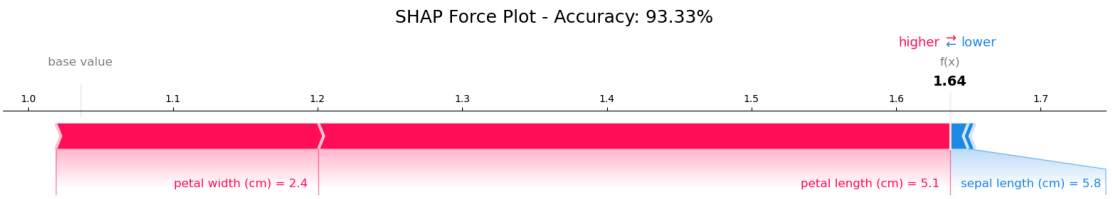


Figure 5.12: *SHAP force plot 5 for the Iris dataset*

Figure 5.13 displays the SHAP waterfall plot for a specific instance in the Iris dataset, showing the contribution of each feature to the prediction. Petal width has the most substantial impact with a SHAP value of -0.19, indicating that its lower value pushes the prediction away from certain species. Petal length contributes positively with a SHAP value of +0.16, reinforcing the model's prediction towards another species. In contrast, sepal length has a near-zero contribution of 0.01, and sepal width has no impact with a SHAP value of 0. This plot highlights how the prediction is primarily influenced by petal measurements, with petal width and petal length shaping the outcome, while the sepal features play an insignificant role in this particular decision.

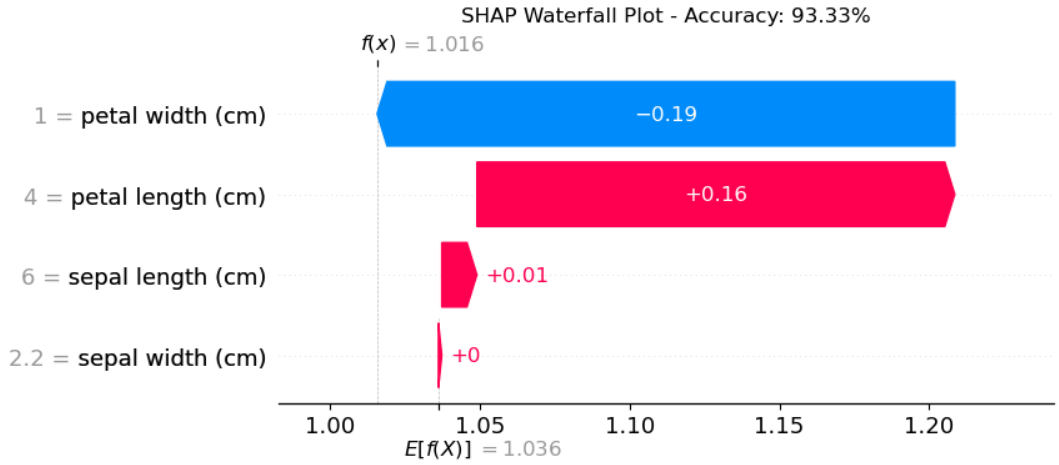
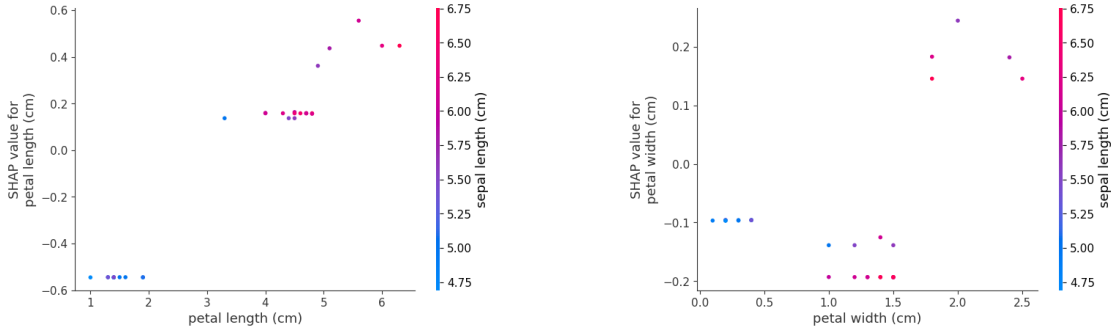


Figure 5.13: *SHAP waterfall for the Iris dataset*

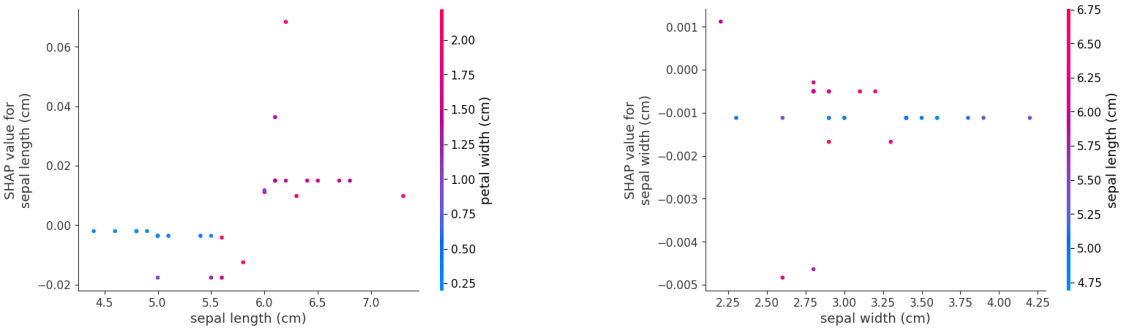
Figure 5.14 contains two subfigures: Subfigure 5.14a shows the SHAP dependence plot for petal length, while Subfigure 5.14b displays the plot for petal width. In Subfigure 5.14a, the SHAP values range from -0.6 to 0.6, indicating a strong influence of this feature on the model's predictions. The spread of points across the plot demonstrates how variations in petal length significantly impact the likelihood of predicting different species, with both positive and negative contributions depending on the length. Subfigure 5.14b has a narrower SHAP range of -0.2 to 0.2, suggesting a more moderate but still important influence on predictions. The points scattered throughout this plot reflect the variation in petal width's contribution to the model's decision-making, complementing the dominant role of petal length.



(a) *petal length* (b) *petal width*

Figure 5.14: *Iris dataset SHAP dependence petal*

Figure 5.15 includes two subfigures: Subfigure 5.15a shows the SHAP dependence plot for sepal length, while Subfigure 5.15b presents the plot for sepal width. Subfigure 5.15a features SHAP values ranging from -0.02 to 0.06, indicating a relatively smaller impact on the model's predictions compared to the petal measurements. The points scattered across this plot show that changes in sepal length contribute slightly to the prediction, but the overall influence remains limited. Subfigure 5.15b has an even narrower SHAP range, from -0.005 to 0.001, suggesting that this feature has a minimal effect on the model's decisions. The points in this plot indicate that variations in sepal width barely alter the prediction outcome, reinforcing its status as the least influential feature among the four.



(a) *Sepal length* (b) *Sepal width*

Figure 5.15: *Iris dataset SHAP dependence sepal*

5.1.3 Explainable Boosting Machines

Figure 5.16 displays the EBM plot of the mean absolute score for each feature in the Iris dataset, highlighting their relative contributions to the model's predictions. Petal width has the highest mean absolute score, exceeding 1.4, indicating that it plays a crucial role in influencing the model's decisions. Petal length follows closely with a score above 1.2, showing its significant impact as well. Sepal length and sepal width contribute to a lesser extent, with mean absolute scores of over 0.7 and 0.6, respectively. While petal measurements are the dominant factors driving the model's behavior, the sepal features still provide additional, albeit smaller, contributions. This plot clearly ranks the importance of each feature, with petal measurements being more predictive than sepal measurements.

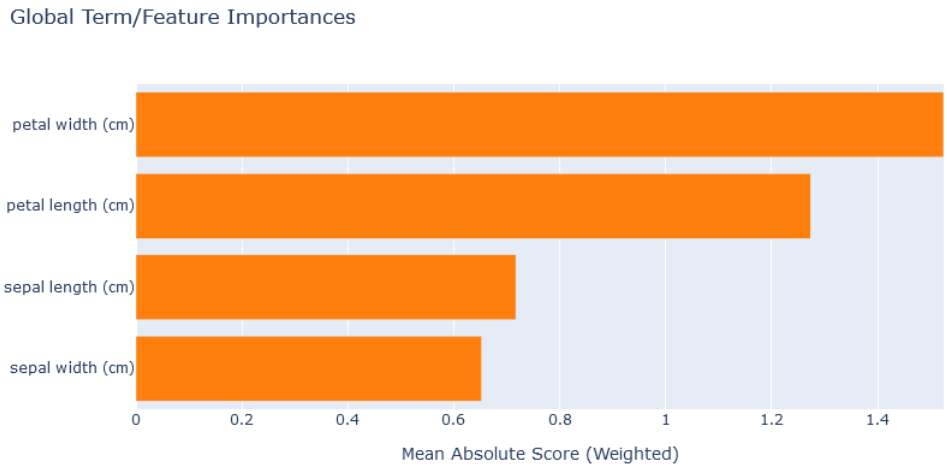


Figure 5.16: *EBM features for the Iris dataset*

Figure 5.17 displays the EBM plot for sepal length, showing the score and density distributions for each of the three classes in the Iris dataset. The plot reveals distinct patterns for each species in relation to sepal length. `Iris setosa` (label 0, shown in blue) follows a decreasing trend, resembling a -x graph, with scores ranging from 2 to -2, indicating that longer sepal lengths reduce the likelihood of classifying an instance as `Iris setosa`. In contrast, `Iris versicolor` (label 1, shown in orange) initially exhibits an increasing x-shaped pattern up to a sepal length of 5 cm, after which the score flattens, suggesting that larger sepal lengths have a consistent impact on `Iris versicolor` predictions beyond this point. `Iris virginica` (label 2, shown in green) maintains a relatively flat score throughout, indicating that variations in sepal length have little effect on its classification. The density plot below shows that most data points fall between 4.66 cm and 6.82 cm, highlighting the common range of sepal lengths in the dataset. This visualization provides insights into how sepal length influences the model's predictions differently for each species.

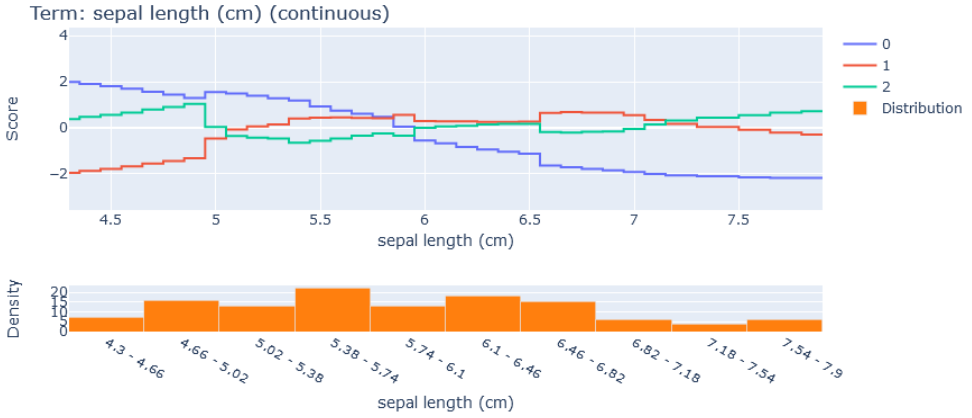


Figure 5.17: *EBM feature sepal length for the Iris dataset*

Figure 5.18 displays the EBM plot for sepal width, showing the score and density distributions for each of the three Iris species. `Iris setosa` (label 0, shown in blue) exhibits an increasing x-shaped pattern, indicating that as sepal width increases, the likelihood of classifying an instance as `Iris setosa` rises. `Iris versicolor` (label 1, shown in orange) maintains a flat score near zero until around 3 cm, after which it follows

a decreasing -x trend, with scores dropping from 0 to -2, suggesting that wider sepal widths reduce the probability of predicting `Iris versicolor`. `Iris virginica` (label 2, shown in green) shows a decreasing -x pattern throughout, starting with a score of -1 and increasing to 1, indicating that narrower sepal widths favor its classification. The density plot below reveals that most data points are concentrated around typical sepal widths for these species, highlighting the variation in sepal width across the dataset. This visualization illustrates how the model uses sepal width differently to influence the prediction for each species.

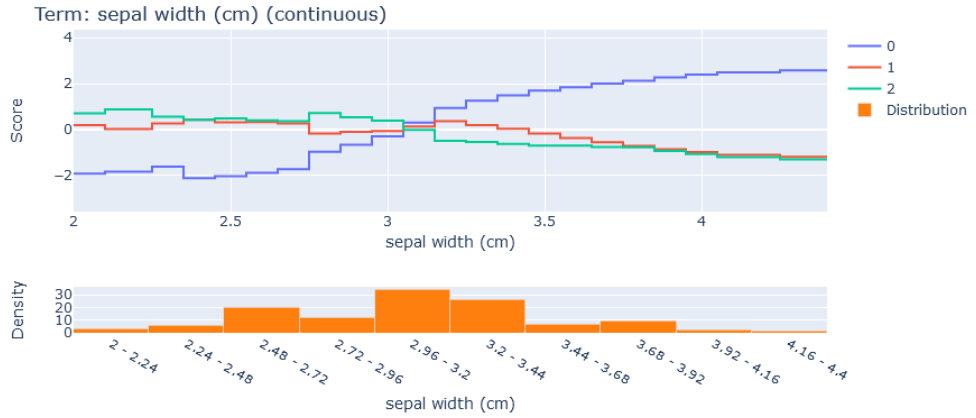


Figure 5.18: *EBM feature sepal width for the Iris dataset*

Figure 5.19 displays the EBM plot for petal length, showing the score and density distributions for each of the three Iris species. `Iris setosa` (label 0, shown in blue) follows a decreasing -x-shaped pattern, with scores starting at 3 and descending to -2 as petal length increases, indicating that shorter petal lengths favor the classification of `Iris setosa`. `Iris versicolor` (label 1, shown in orange) exhibits a bell-shaped curve, beginning near -1, peaking around a score of 2, and dropping back down to -2, suggesting that mid-range petal lengths increase the likelihood of predicting `Iris versicolor`. `Iris virginica` (label 2, shown in green) shows a sigma-shaped function, with scores transitioning from -2 to 2, highlighting that longer petal lengths strongly support the classification of `Iris virginica`. The density plot below indicates the common ranges of petal length for each species, providing insight into how variations in petal length distinctly influence the model's predictions.

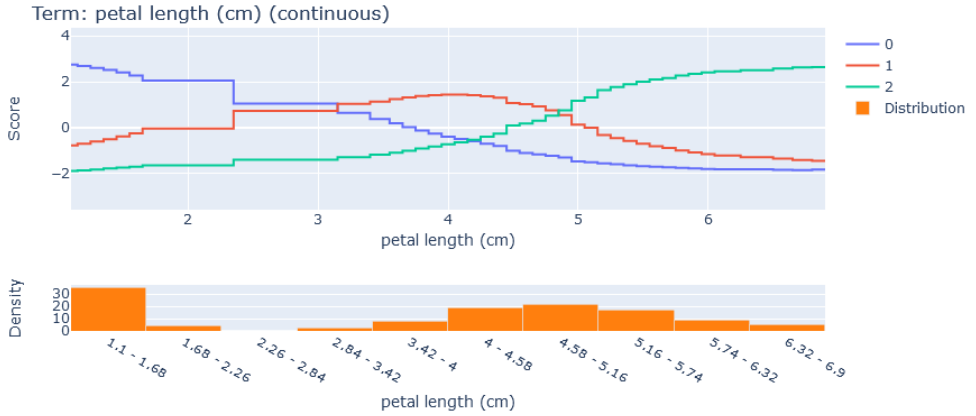


Figure 5.19: *EBM feature petal length for the Iris dataset*

Figure 5.20 displays the EBM plot for petal width, showing the score and density distributions for each of the three Iris species. `Iris setosa` (label 0, shown in blue) exhibits a decreasing -x-shaped curve, with scores beginning at 3 and descending to -2 as petal width increases, suggesting that narrower petal widths favor the classification of `Iris setosa`. `Iris versicolor` (label 1, shown in orange) follows a bell-shaped curve, starting at a score of 0, peaking at 2, and dropping to -2, indicating that mid-range petal widths increase the likelihood of predicting `Iris versicolor`. `Iris virginica` (label 2, shown in green) displays an increasing x-shaped function, with scores starting at -2 and rising to 3, showing that larger petal widths support the classification of `Iris virginica`. The density plot below highlights the typical distribution of petal width values across the dataset, underscoring how each species responds differently to variations in this feature.

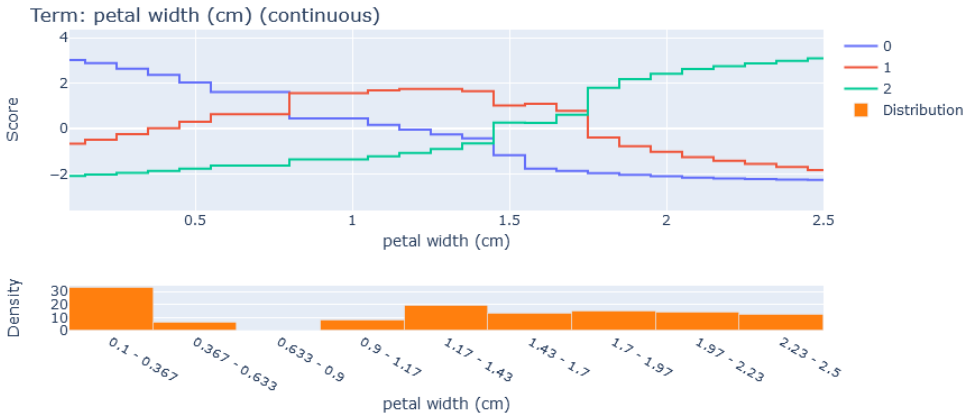


Figure 5.20: *EBM feature petal width for the Iris dataset*

Figure 5.21 displays an EBM horizontal bar plot for an instance where the actual and predicted class are both `Iris setosa` (label 0, shown in blue). In this plot, each feature contributes positively toward `Iris setosa`, reinforcing the model's confidence in this prediction. Conversely, `Iris virginica` (label 2, shown in green) has a consistently negative impact across all features, reducing its likelihood. `Iris versicolor` (label 1, shown in orange) has a minimal effect overall, neither strongly supporting nor opposing the prediction. This distribution highlights a clear alignment of features with the correct class, resulting in an accurate classification as `Iris setosa`.

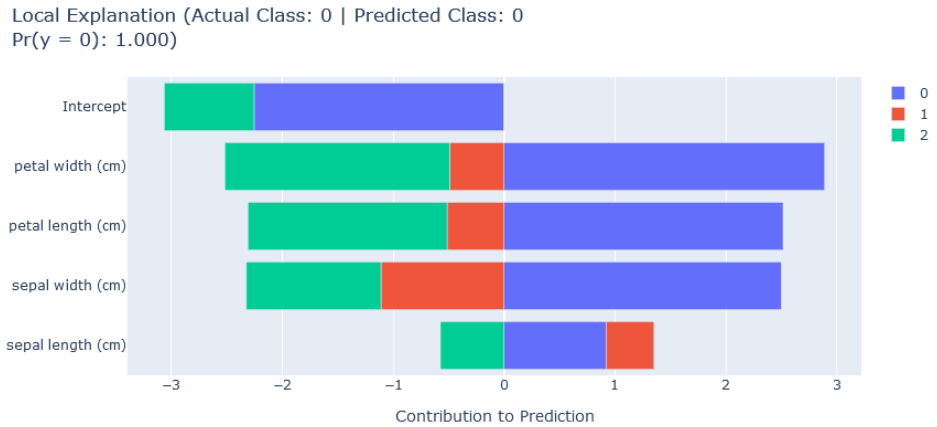


Figure 5.21: *EBM correct classification for iris setosa*

Figure 5.22 illustrates an EBM horizontal bar plot for an instance where the actual and predicted class are both `Iris versicolor` (label 1, shown in orange). In this plot, petal width and petal length strongly support the prediction, contributing significantly positive values toward `Iris versicolor`. Sepal width shows a large negative contribution for `Iris setosa` (label 0, shown in blue), reducing its likelihood. Additionally, `Iris virginica` (label 2, shown in green) has small negative contributions in the petal features and a minor positive effect in sepal width. This distribution of feature influences reinforces the model's confidence in classifying this instance as `Iris versicolor`.

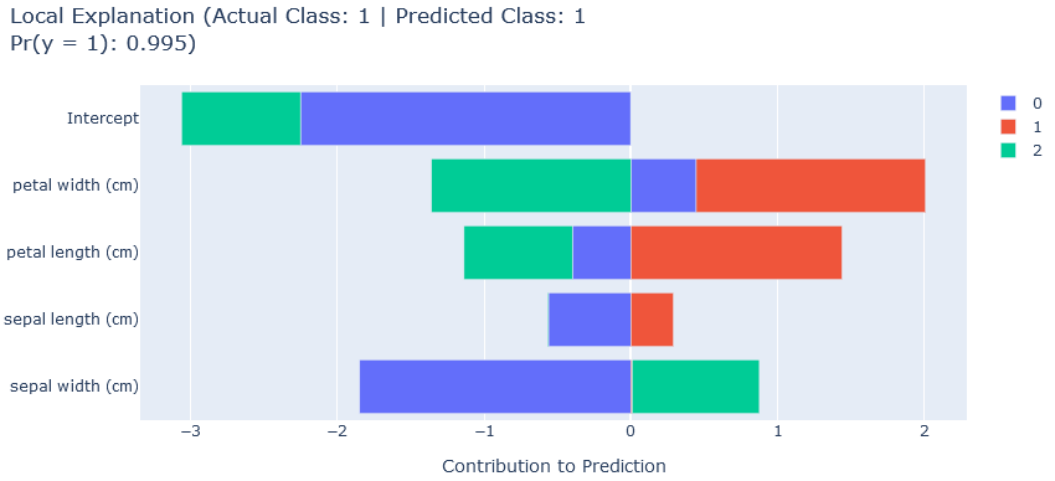


Figure 5.22: *EBM correct classification for iris versicolor*

Figure 5.23 shows an EBM horizontal bar plot for an instance where the actual and predicted class are both `Iris virginica` (label 2, shown in green). In this plot, petal width has a strong positive contribution toward `Iris virginica`, with additional, smaller positive impacts from petal length and sepal width. In contrast, `Iris setosa` (label 0, shown in blue) has negative contributions from both petal features and sepal width, reducing its likelihood as the predicted class. Sepal length has almost no impact across the classes, indicating limited influence on the prediction. This configuration of feature impacts aligns with the correct classification as `Iris virginica`.

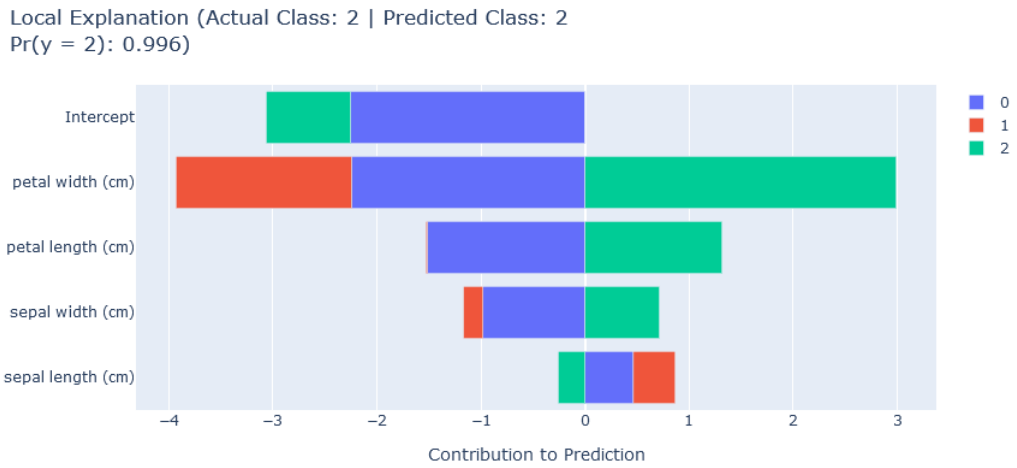
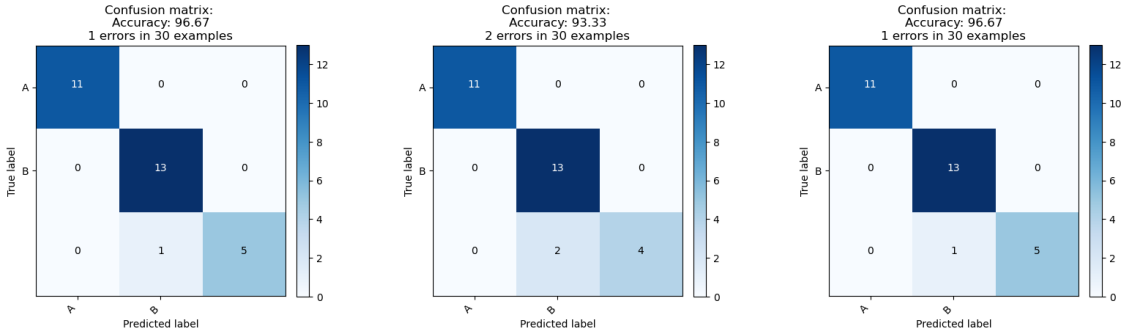


Figure 5.23: *EBM correct classification for iris virginica*

5.1.4 Discussion

In this final analysis of the Iris dataset, Figure 5.24 presents three confusion matrices, each corresponding to a different model applied in this study: a DT, SHAP, and EBM. Although the models display similar performance metrics, the focus here is not on their accuracy but on the interpretability and insights each model provides regarding the features that drive the predictions.



(a) *DT* (b) *SHAP* (c) *EBM*

Figure 5.24: *Confusion Matrix for the Iris dataset*

The DT model offered a straightforward visualization of feature importance, with `petal width` consistently emerging as the initial splitting node, underscoring its significant role in distinguishing between the Iris species. However, this differs slightly from the SHAP results, where visualizations such as the bar plot, beeswarm plot, and force plots highlighted `petal length` as the most influential feature in the model's decisions, with `petal width` identified as the second most important. The high SHAP values for both features across multiple instances reaffirmed their critical roles in determining species classification, especially between species with similar characteristics.

EBM further reinforced these observations through its feature weight analysis, which confirmed `petal width` as the dominant feature, followed closely by `petal length`. This feature ranking provided an additive, transparent view of how these petal dimensions contribute to predictions, aligning well with the DT and SHAP findings. EBM visualizations allowed for a clear understanding of each feature's effect on the model's classification, emphasizing that petal measurements, particularly width and length, are the primary drivers of the model's decisions.

Overall, all three interpretability tools consistently identified `petal length` and `petal width` as the most influential features in the Iris dataset. This alignment across models underscores the robustness of the interpretability methods used, demonstrating their effectiveness in highlighting the core features responsible for species differentiation. By focusing on interpretability, this analysis provides a comprehensive understanding of the features driving classification in the Iris dataset, with petal measurements, especially `petal length` for SHAP and `petal width` for DT and EBM, emerging as key contributors across all models.

5.2 Adult dataset

In this chapter, we present the results obtained from applying SHAP and EBM methods to the Adult dataset. The results include performance metrics such as accuracy, complemented by SHAP visualizations to interpret feature importance and interactions. SHAP summary plots and dependence plots are used to explain the influence of features such as age, education level, and occupation on the model's predictions regarding income classification. EBM's inherently interpretable modeling provides additional insights into how these features contribute to predicting whether an individual's income exceeds a certain threshold, making the model's decision-making process transparent and easier to understand.

5.2.1 Decision Trees

Figure 5.25 presents the DT generated for the Adult dataset, showcasing the model's structure for predicting income categories. The first node in the tree is `relationship`, which serves as a key initial split, reflecting its strong influence in distinguishing income levels. This choice suggests that relationship status, such as whether someone is married, single, or has dependents, provides an effective starting point for income classification. On the left branch, the next node is `capital gain`, likely capturing high-income individuals with substantial investments or assets. On the right, `education num` indicates that educational attainment is a defining factor for those not separated by capital gain, contributing further to refining the model's predictions. This structure highlights the importance of social and economic variables in determining income, with relationship status as a foundational feature in the model.

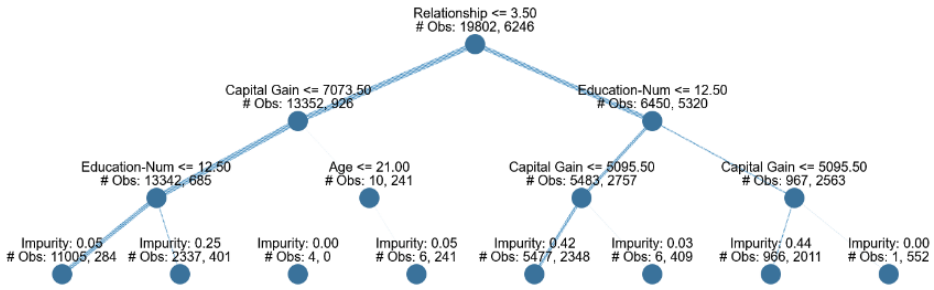


Figure 5.25: *Decision Tree for the Adult dataset*

Figure 5.26 illustrates the DT path leading to a correct classification with the label `False`. The highlighted path in orange starts with `relationship` as the initial split, moving through `capital gain` based on the instance's characteristics. This sequence shows how these key features contributed to the accurate prediction, with each decision guiding the model towards the correct outcome.

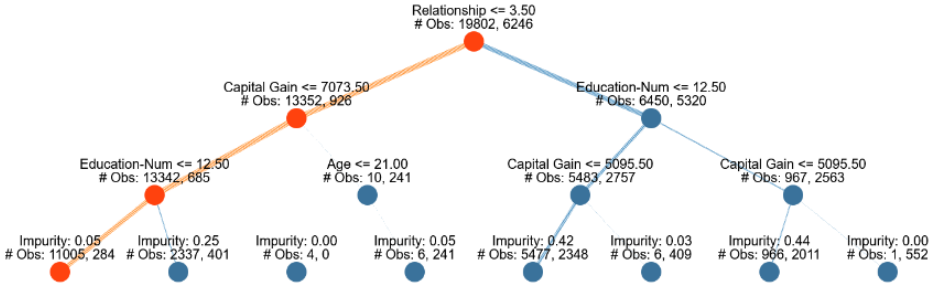


Figure 5.26: *Decision Tree for the Adult dataset label false*

Figure 5.27 shows the DT path for a correct classification with the label `True`. The path, highlighted in orange, begins at the `relationship` node and progresses through `education num`. Each feature along this path played a role in guiding the model to its correct prediction, illustrating the weighted influence of these attributes.

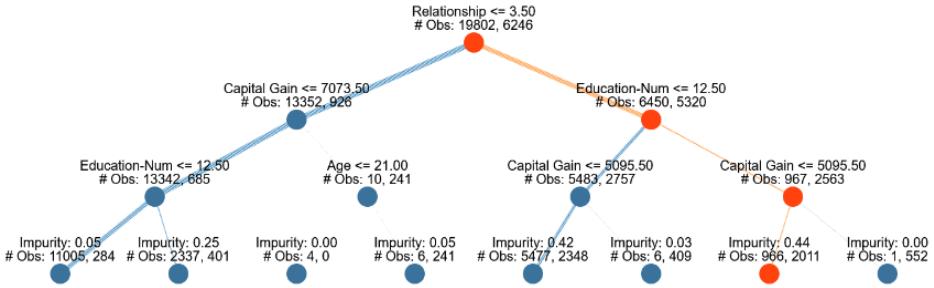


Figure 5.27: *Decision Tree for the Adult dataset label true*

5.2.2 SHapley

Figure 5.28 presents the SHAP bar plot for the Adult dataset, displaying the average contribution of each feature to the model's predictions regarding income classification. The feature `relationship` has the highest SHAP value at +0.1, indicating its significant influence on the model's decision-making process. This high value suggests that relationship status, which includes categories such as married, single, or with dependents, plays a primary role in distinguishing income levels. Following relationship, `education-num` contributes positively with a SHAP value of +0.05, emphasizing the impact of years of education on income, as higher educational attainment often correlates with higher-paying job opportunities. `Capital gain` ranks third with a SHAP value of +0.03, which reflects the effect of financial assets or investments on predicting higher income brackets. Additional features, such as `age` and `hours per week`, have smaller positive SHAP values at +0.02 and +0.01, respectively. These contributions indicate that while age and working hours per week have less predictive power than relationship or education, they still provide meaningful context, as older individuals with more work hours may have higher incomes. `Capital loss` also has a small positive SHAP value at +0.01, likely accounting for instances where financial loss might affect the model's classification, although it plays a less prominent role than capital gain. Interestingly, features like `occupation`, `marital status`, and `sex`, as well as a combined set of three other features, contribute a net SHAP value of zero. This lack of influence suggests that, within the model's framework, these features do not

add substantial value in determining income classification, either due to less variation in income among these groups or due to their interaction with more impactful features. The SHAP bar plot thus provides a clear ranking of feature importance, highlighting the primary contributions of relationship, education, and capital gain, while indicating the minimal influence of several demographic and occupational attributes.

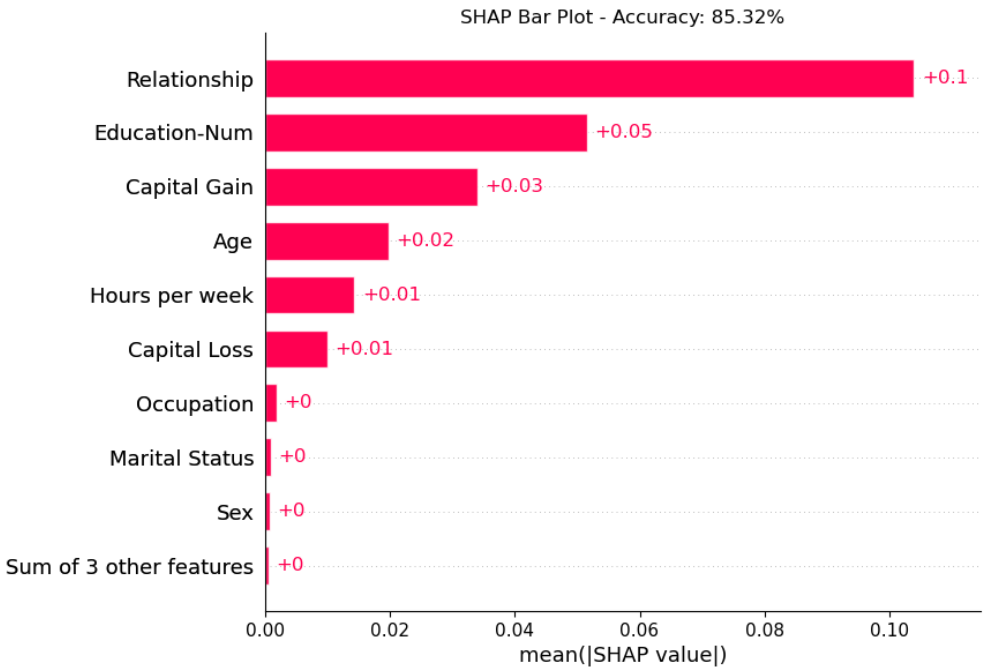


Figure 5.28: *SHAP barplot for the Adult dataset*

Figure 5.29 presents the SHAP beeswarm plot for the Adult dataset, showing the distribution of SHAP values across instances for each feature, ordered by their average importance. The most influential feature, `relationship`, exhibits a concentrated spread of SHAP values, indicating consistent predictive power across instances and highlighting the strong role of relationship status in income classification. `Education-num` follows with a stable impact, reflecting the reliable influence of years of education on the model's predictions, where higher education levels correlate positively with income.

Distinctly, `capital gain` and `capital loss` show substantial spreads in their SHAP values, indicating considerable variability in their influence across instances. This broad distribution for capital gain underscores its significant impact on income prediction for certain individuals, likely those with higher financial investments or assets. Similarly, capital loss displays a large spread, showing that in cases with substantial financial losses, this feature can strongly influence the prediction, either enhancing or diminishing the likelihood of a high-income classification.

The remaining features, such as `age` and `hours per week`, exhibit smaller spreads, contributing moderately to the model's predictions. These indicate that while age and work hours add value to the classification process, their impact is more stable and less variable than capital-related features. Featur

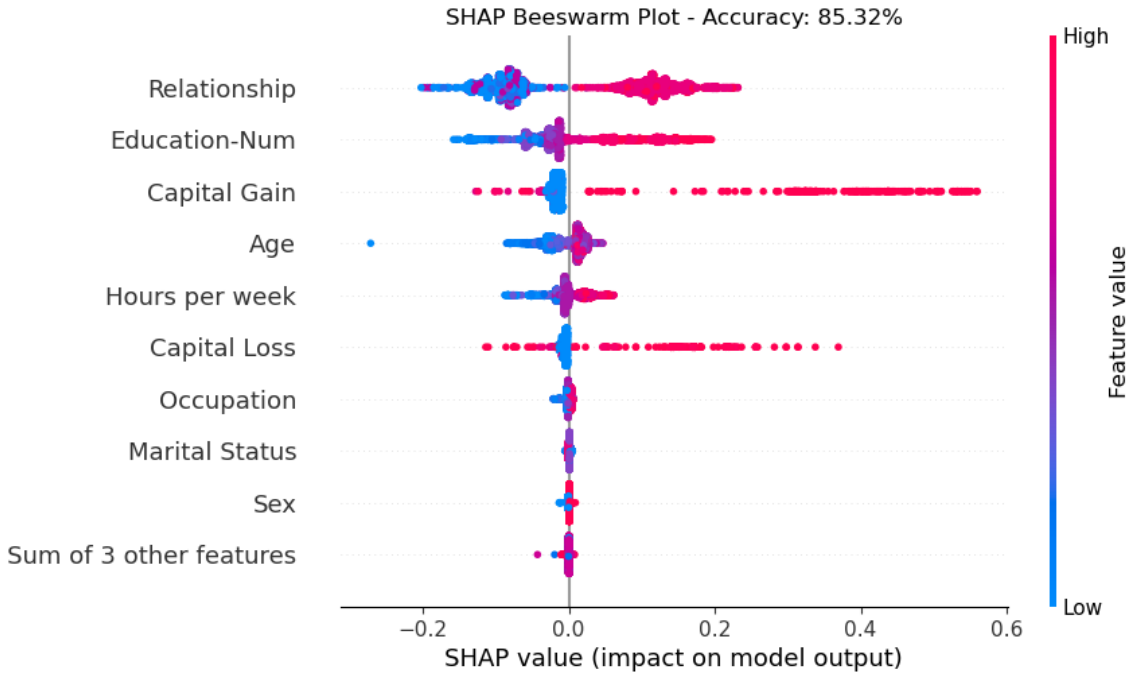


Figure 5.29: *SHAP beeswarm for the Adult dataset*

Figure 5.30 shows the SHAP force plot for `relationship`, highlighting its strong, consistent impact on income predictions across instances.

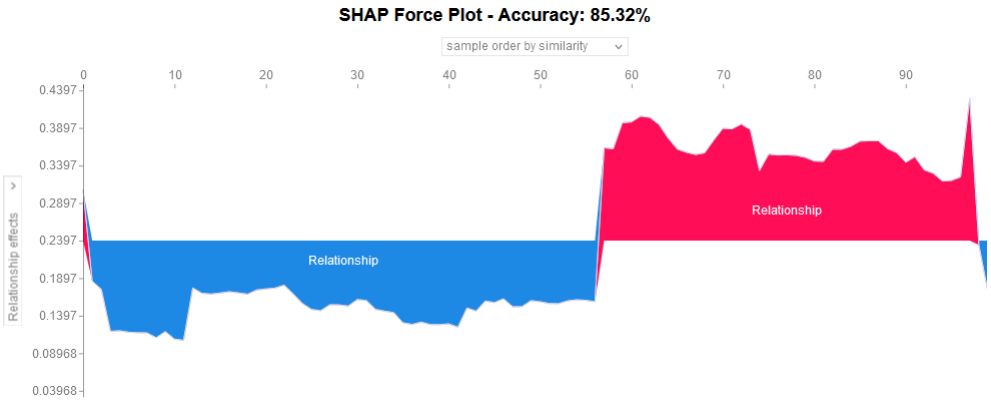


Figure 5.30: *SHAP force plot 1 for the Adult dataset*

Figure 5.31 presents the SHAP force plot for `education-num`, reflecting a stable influence where higher values favor higher-income predictions.

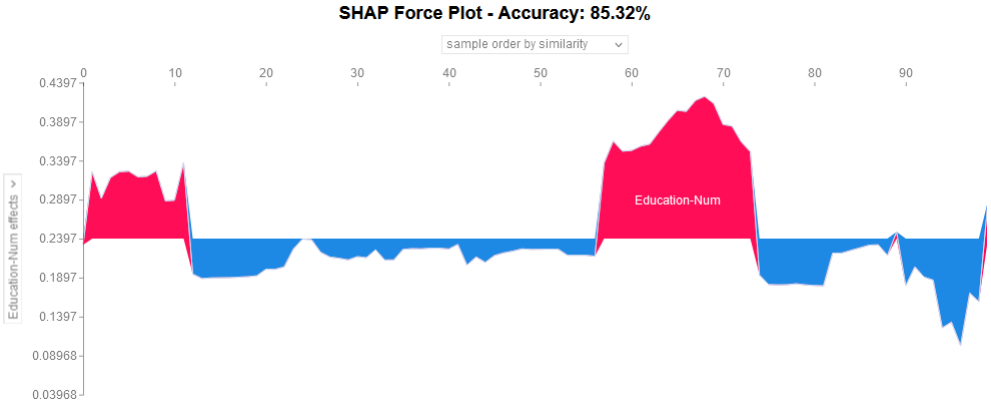


Figure 5.31: *SHAP force plot 2 for the Adult dataset*

Figure 5.32 illustrates the SHAP force plot for `capital gain`, revealing significant variation, with positive values boosting the likelihood of high-income predictions.

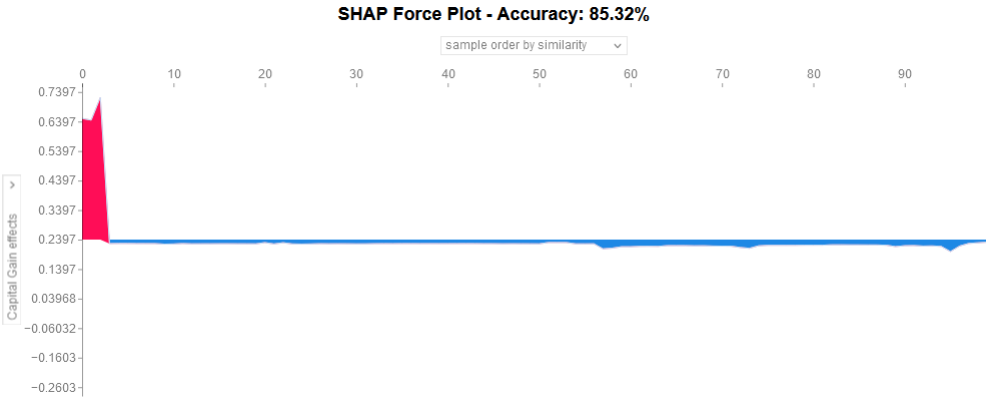


Figure 5.32: *SHAP force plot 3 for the Adult dataset*

Figure 5.33 displays the SHAP force plot for `age`, showing a moderate, steady effect, where older ages slightly increase income predictions.

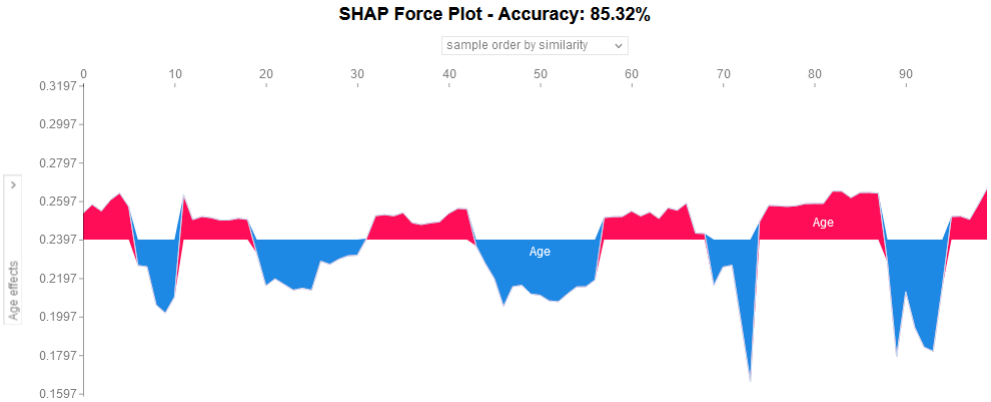


Figure 5.33: *SHAP force plot 4 for the Adult dataset*

Figure 5.34 shows the SHAP force plot for `hours per week`, where higher work hours slightly favor higher-income classifications.

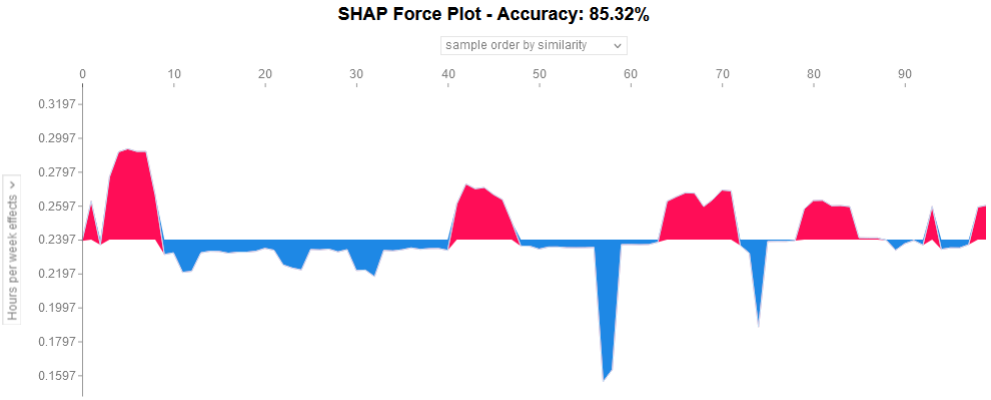


Figure 5.34: *SHAP force plot 5 for the Adult dataset*

Figure 5.35 presents the SHAP force plot for `capital loss`, showing wide variability, with high values negatively impacting income predictions.

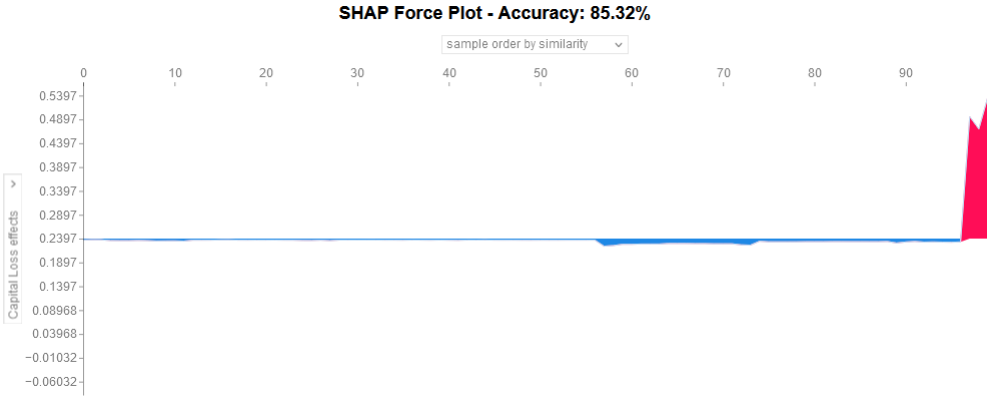


Figure 5.36: *SHAP force plot 6 for the Adult dataset*

Figure 5.36 displays the SHAP force plot for `occupation`, with minimal influence as contributions are close to zero across instances.

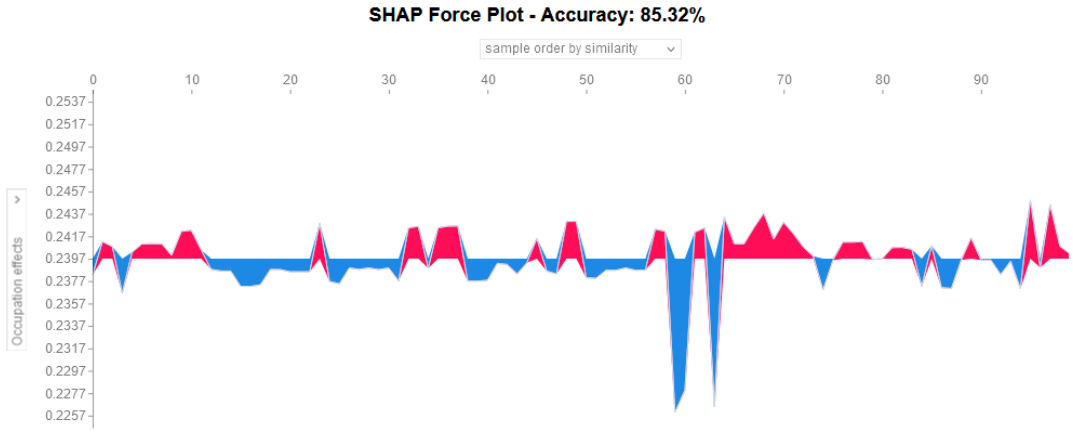


Figure 5.36: *SHAP force plot 7 for the Adult dataset*

Figure 5.37 presents the SHAP force plot for `sex`, showing minimal contribution to income predictions, with little impact across instances.

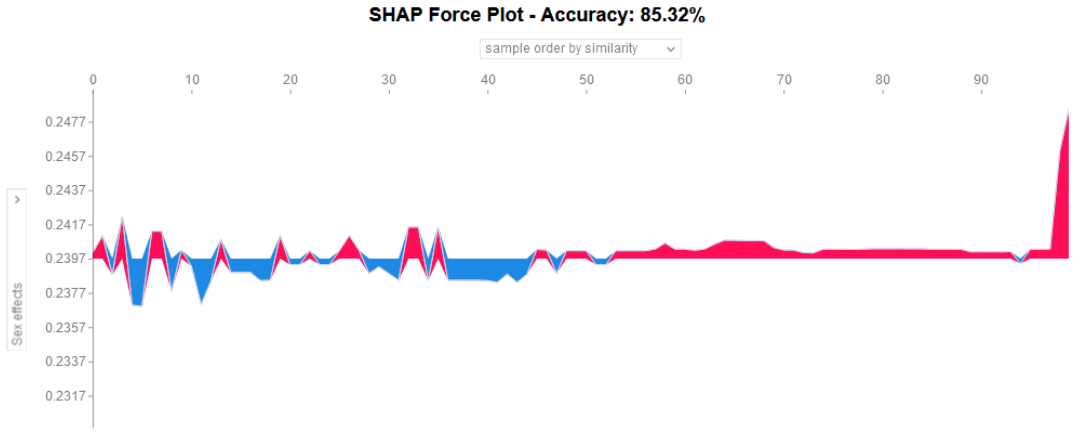


Figure 5.37: *SHAP force plot 8 for the Adult dataset*

Figure 5.38 shows the SHAP force plot for `marital status`, indicating a minor effect on income predictions, with near-zero values.

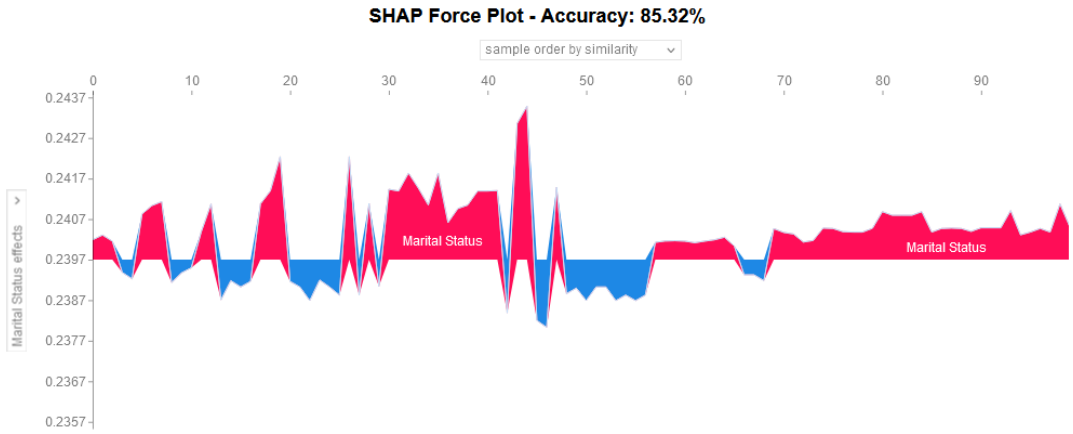


Figure 5.38: *SHAP force plot 9 for the Adult dataset*

Figure 5.39 presents the SHAP force plot for `workclass`, where contributions remain near zero, indicating a limited impact on the model's predictions.

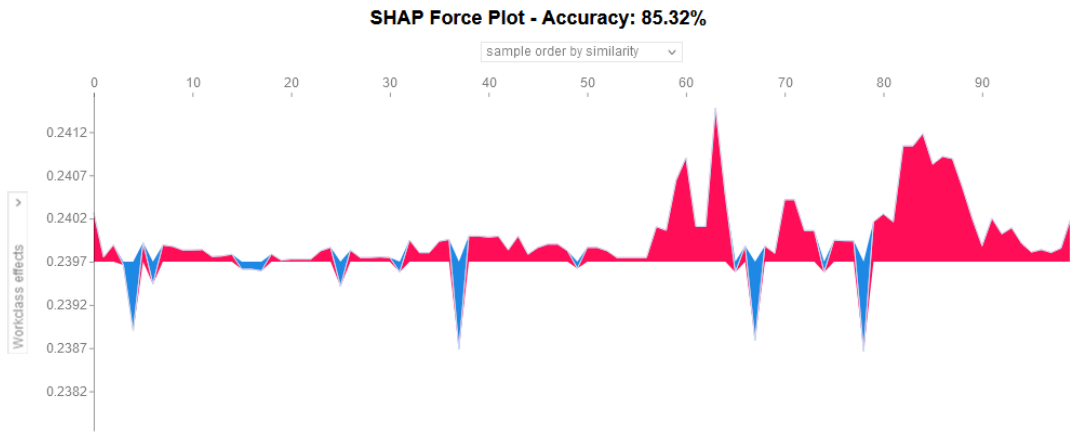


Figure 5.39: *SHAP force plot 10 for the Adult dataset*

Figure 5.40 illustrates the SHAP force plot for `race`, showing a minor effect on income classifications, with values centered around zero.

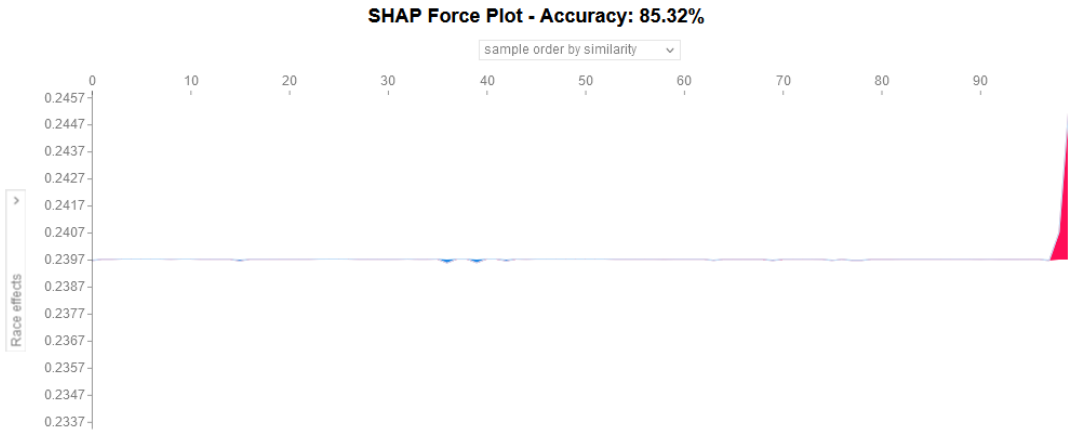


Figure 5.40: *SHAP force plot 11 for the Adult dataset*

Figure 5.41 displays the SHAP force plot for `native-country`, showing minimal influence on income predictions, with values close to zero across most instances.

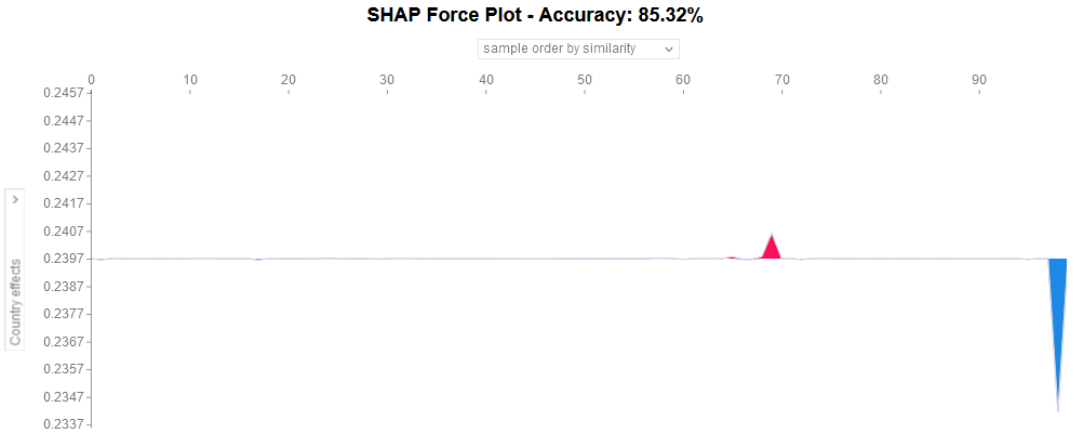


Figure 5.41: *SHAP force plot 12 for the Adult dataset*

Figure 5.42 presents the SHAP force plot displaying the combined contributions of all features for a specific prediction in the Adult dataset. The plot illustrates how each feature, either positively or negatively, influences the model's final decision. Notably, features such as `relationship`, `education-num`, and `capital gain` show the most substantial impact, with strong positive or negative shifts pushing the prediction towards a particular income classification. In contrast, features like `occupation`, `sex`, and `native-country` display minimal contributions, remaining near zero and having little effect on the model's outcome. This collective view of feature impacts provides insight into the key factors driving the prediction, highlighting the importance of socioeconomic features while deemphasizing demographic ones.

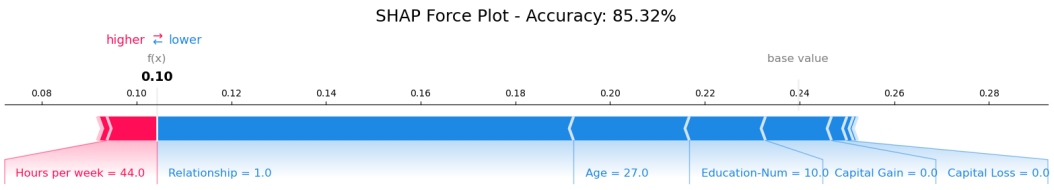


Figure 5.42: *SHAP force plot 13 for the Adult dataset*

Figure 5.43 shows the SHAP waterfall plot for a specific prediction in the Adult dataset, highlighting the primary feature contributions. `Relationship` has the largest impact with a SHAP value of -0.13, pulling the prediction downward, while `education-num` contributes positively with a SHAP value of +0.05, slightly supporting a higher income classification. `Age` has a smaller negative effect at -0.03, while the remaining features have values close to zero, indicating minimal influence on the prediction. This plot emphasizes the dominant roles of relationship, education, and age in shaping the model's decision.

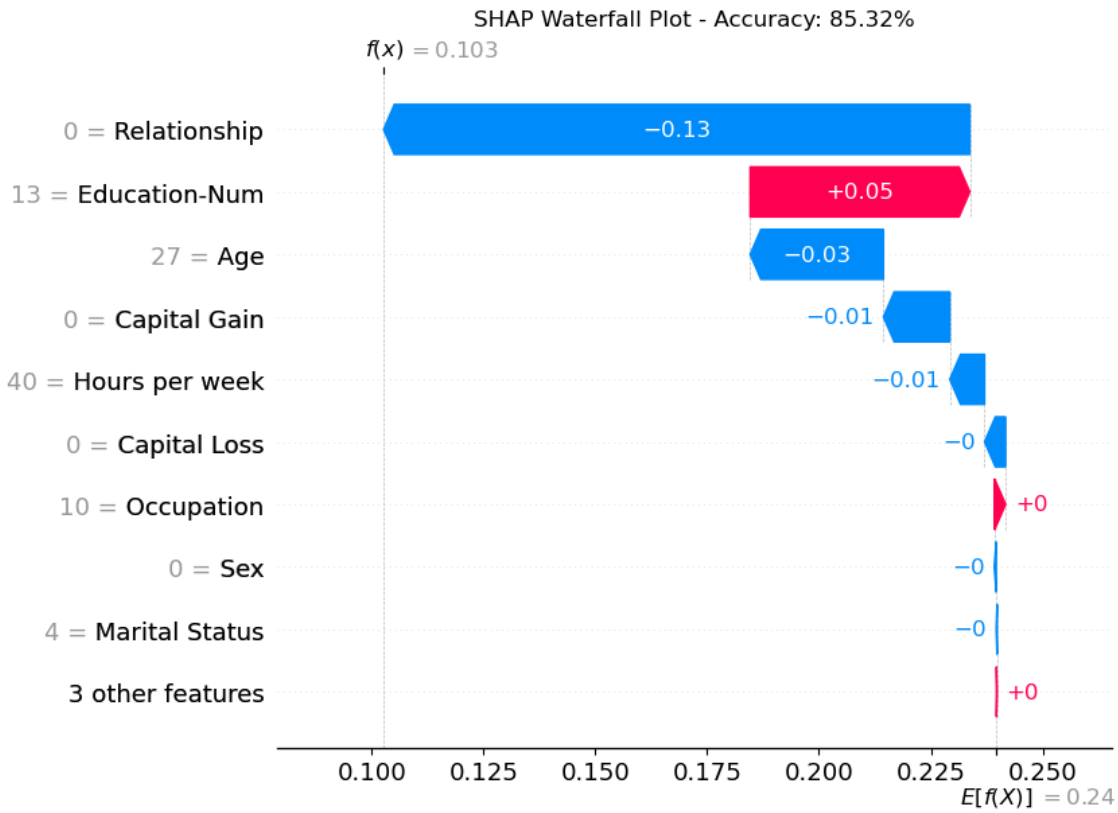
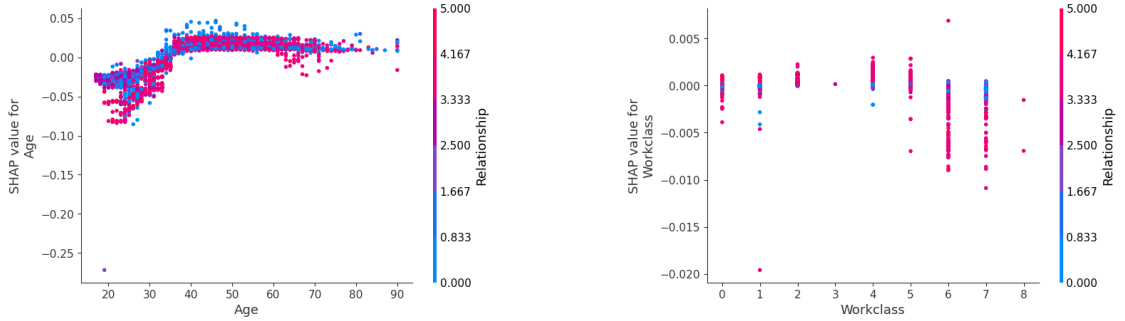


Figure 5.43: *SHAP waterfall for the Adult dataset*

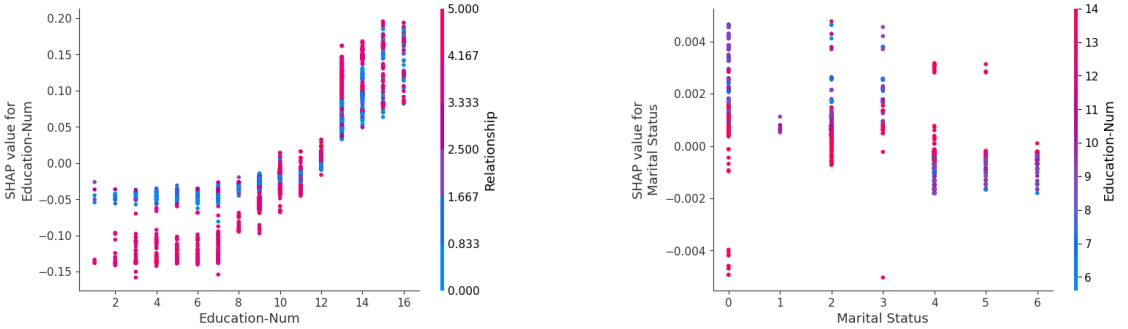
Figure 5.44 presents the SHAP dependence plots for `age` and `workclass`. The `age` plot shows moderate dispersion, indicating its steady influence on income predictions across various age ranges, with older ages generally increasing the probability of a higher income classification. This spread suggests that age has a consistent, though not dominant, effect on the model's decisions. In comparison, `workclass` has a much narrower spread around zero, reflecting its limited impact on predictions, as it contributes minimally to variations in income classification across instances.



(a) *Feature Age* (b) *Feature Workclass*

Figure 5.44: *SHAP dependence plot 1 for the Adult dataset*

Figure 5.45 displays the dependence plots for `education-num` and `marital status`. The `education-num` plot has noticeable dispersion, highlighting the role of educational attainment in income predictions, with higher values generally pushing predictions toward higher income categories. This spread underscores education's meaningful impact, as more years of education correlate with increased income likelihood. In contrast, `marital status` shows a tighter spread near zero, suggesting a smaller, more neutral role in the model's classification process, contributing only minimally to predictive changes.

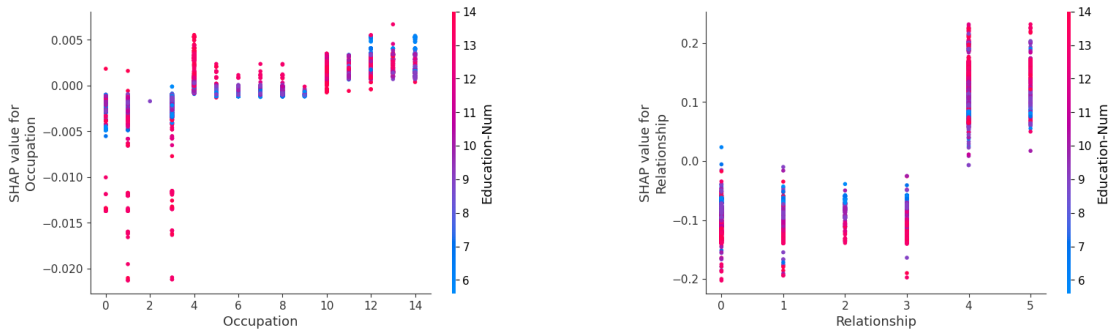


(a) *Feature Education-Num* (b) *Feature Martial Status*

Figure 5.45: *SHAP dependence plot 2 for the Adult dataset*

Figure 5.46 shows the dependence plots for `occupation` and `relationship`. `Occupation` displays a relatively low spread, indicating its minimal effect on income predictions and reflecting only minor SHAP contributions across different occupational categories. In contrast, `relationship` exhibits a wider range of SHAP values, underscoring its strong influence in distinguishing income categories based on relationship status. This variation indicates that certain relationship statuses, like being married or single, have a more direct

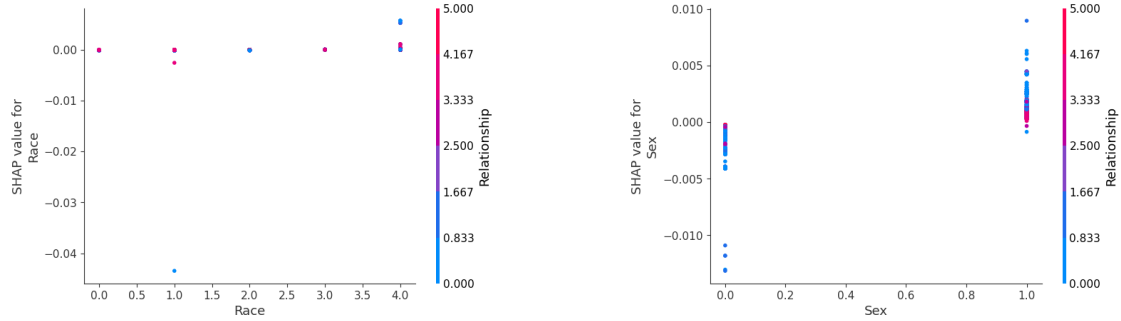
impact on income likelihood within the model.



(a) *Feature Occupation* (b) *Feature Relationship*

Figure 5.46: *SHAP dependence plot 3 for the Adult dataset*

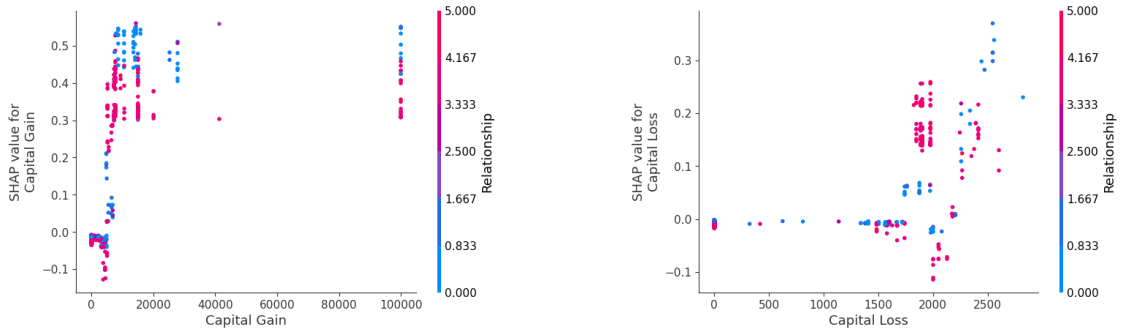
Figure 5.47 presents the dependence plots for `race` and `sex`. Both features display limited dispersion around zero, suggesting that `race` and `sex` contribute minimally to the model's income predictions. The near-zero range of SHAP values indicates a minor role in the classification process, with little effect on the model's output. This tight spread suggests that any slight variations in income predictions due to these features are infrequent and likely overshadowed by more influential socioeconomic factors.



(a) *Feature Race* (b) *Feature Sex*

Figure 5.47: *SHAP dependence plot 4 for the Adult dataset*

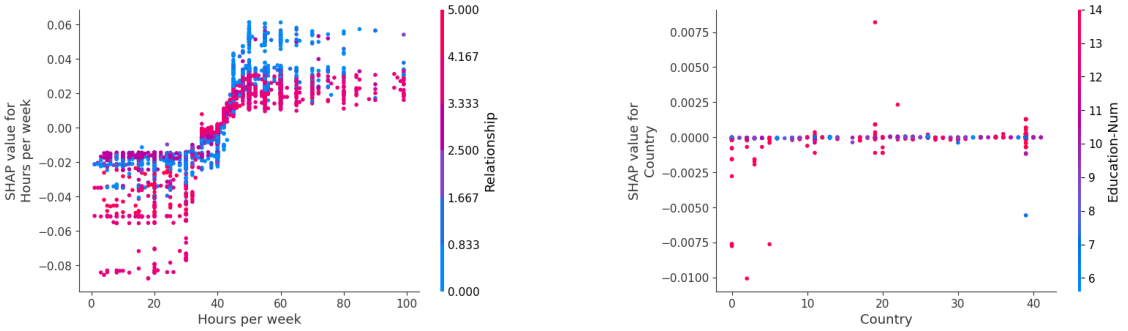
Figure 5.48 shows the SHAP dependence plots for `capital gain` and `capital loss`. Both features exhibit considerable spread, with `capital gain` showing a particularly wide range of SHAP values, highlighting its significant and variable impact on income predictions. This variability suggests that individuals with substantial capital gains greatly affect the likelihood of a high-income classification. Meanwhile, `capital loss` also contributes variably, albeit to a lesser extent, reflecting that while losses can impact income classification, their influence is less pronounced than capital gains.



(a) *Feature Capital Gain* (b) *Feature Capital Loss*

Figure 5.48: *SHAP dependence plot 5 for the Adult dataset*

Figure 5.49 displays the dependence plots for `hours per week` and `native-country`. `Hours per week` shows moderate dispersion, with SHAP values indicating that working more hours per week supports higher income classifications. This steady spread reflects that working hours moderately impact predictions, although not as strongly as features like relationship or capital gain. `Native-country` has minimal spread, indicating its negligible influence on the model's decisions, as country of origin does not significantly alter the model's predictions in this dataset.



(a) *Feature Hours per week* (b) *Feature Country*

Figure 5.49: *SHAP dependence plot 6 for the Adult dataset*

5.2.3 Explainable Boosting Machines

Figure 5.50 presents the EBM bar plot for the Adult dataset, ranking each feature by its contribution to the model's predictions regarding income classification. `Age` stands out as the most influential feature, with a score close to 0.9, indicating its significant impact on income prediction. `Capital gain` follows, just below 0.7, highlighting its importance as a predictor of higher income levels. `Relationship`, `marital status`, and `education-num` all contribute similarly, with values just below 0.6, underscoring the role of social and educational factors in distinguishing income classes. Further down, `occupation` and `sex` each have values under 0.4, reflecting moderate influence on predictions. `Hours per week` contributes with a score of 0.3, suggesting a lower but still relevant impact on the model's output. `Capital loss` shows limited influence, with a value below 0.2. Finally, `workclass`, `race`, and `native-country` display minimal contributions, each with scores around 0.1 or less, indicating their relatively minor effect on income classification. This

bar plot highlights how the model relies most heavily on age and financial indicators, while demographic details like race and country have minimal predictive power.

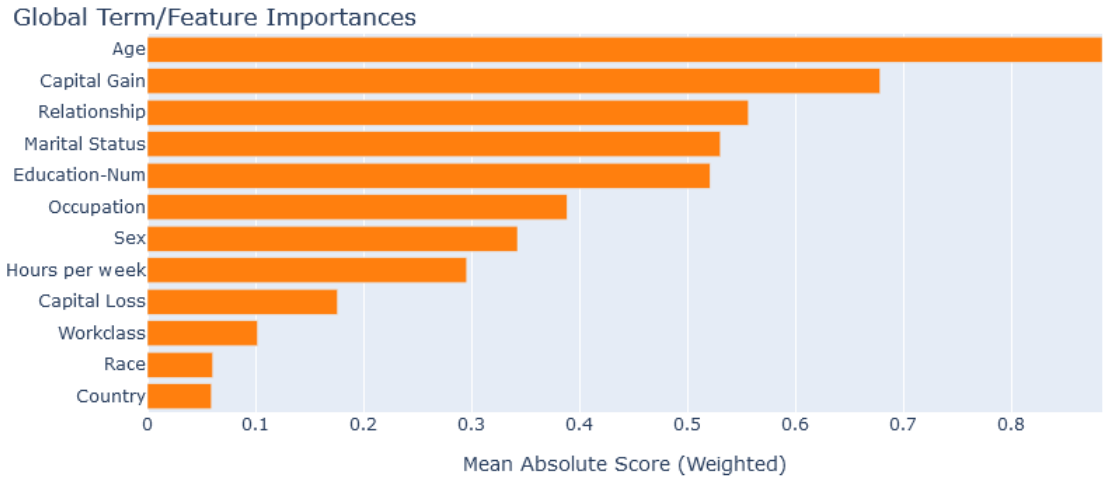


Figure 5.50: *EBM features for the Adult dataset*

Figure 5.51 shows the EBM score and density plots for `age`. The score plot reveals a strong positive correlation between age and income prediction, with older ages associated with higher scores, indicating a greater likelihood of higher income classifications. The density plot below highlights that most data points cluster around middle age, but there is a broad distribution, allowing the model to capture a range of age-related patterns.

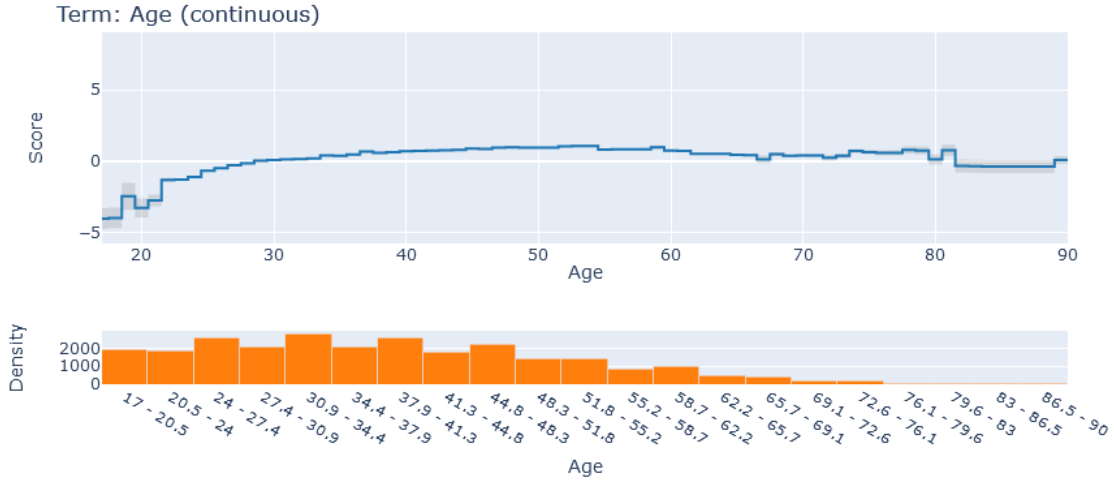


Figure 5.51: *EBM feature analysis for* `Age feature` *in the Adult Dataset*

Figure 5.52 displays the score and density plots for `capital gain`. The score plot demonstrates a sharp increase for instances with high capital gains, reflecting a substantial influence on the prediction of higher income. The density plot indicates that while most data points have low or zero capital gain, the few high values significantly impact the model's predictions.

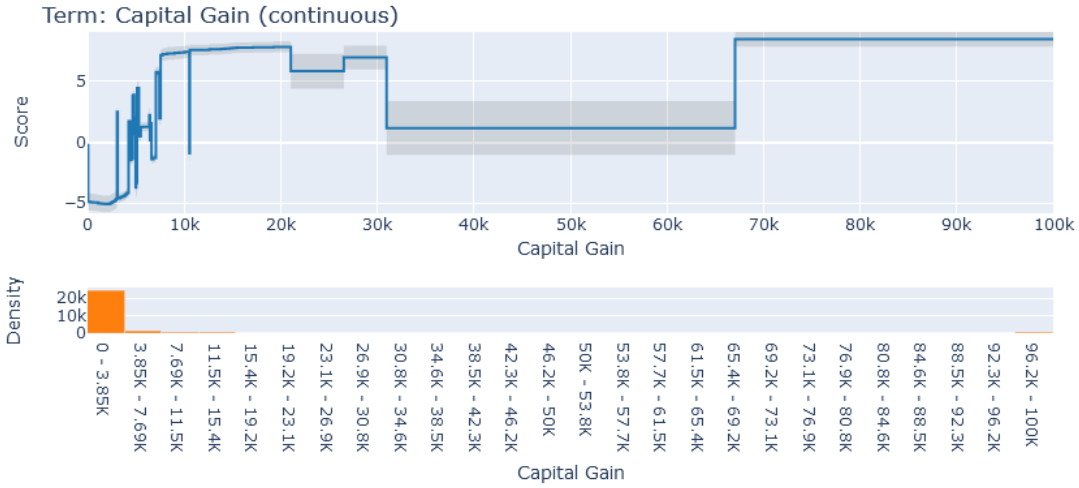


Figure 5.52: *EBM feature analysis for* ***`Capital Gain`*** *in the Adult Dataset*

Figure 5.53 presents the EBMscore and density plots for `relationship`. The score plot shows distinct values for different relationship statuses, with certain categories, such as `married`, positively impacting income predictions. The density plot highlights a concentrated distribution around specific relationship types, illustrating how these groupings affect income classification.

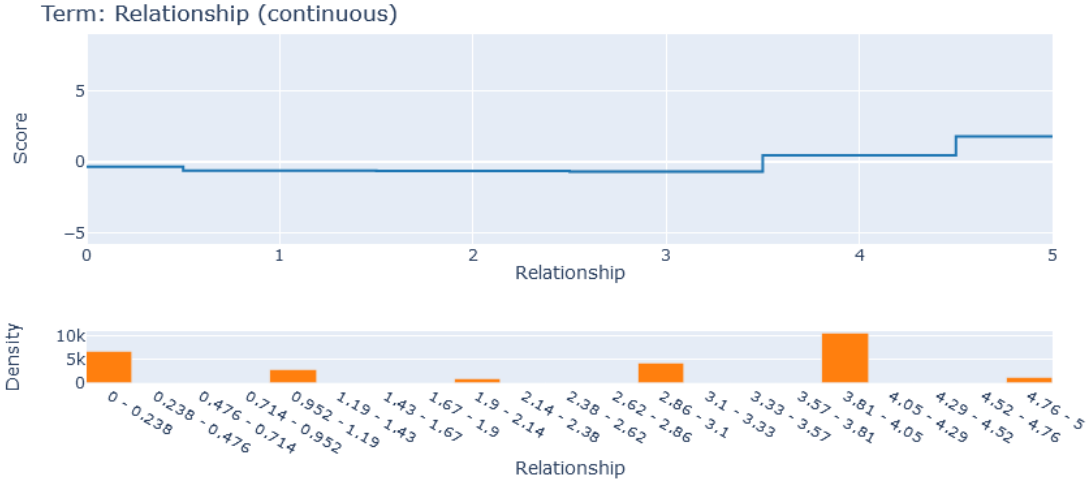


Figure 5.53: *EBM feature analysis for* ***`Relationship`*** *in the Adult Dataset*

Figure 5.54 shows the score and density plots for `marital status`. The score plot reflects how certain marital statuses contribute positively to higher income predictions, while others show little effect. The density plot reveals a concentration around common statuses, helping the model to differentiate income patterns associated with marital status.

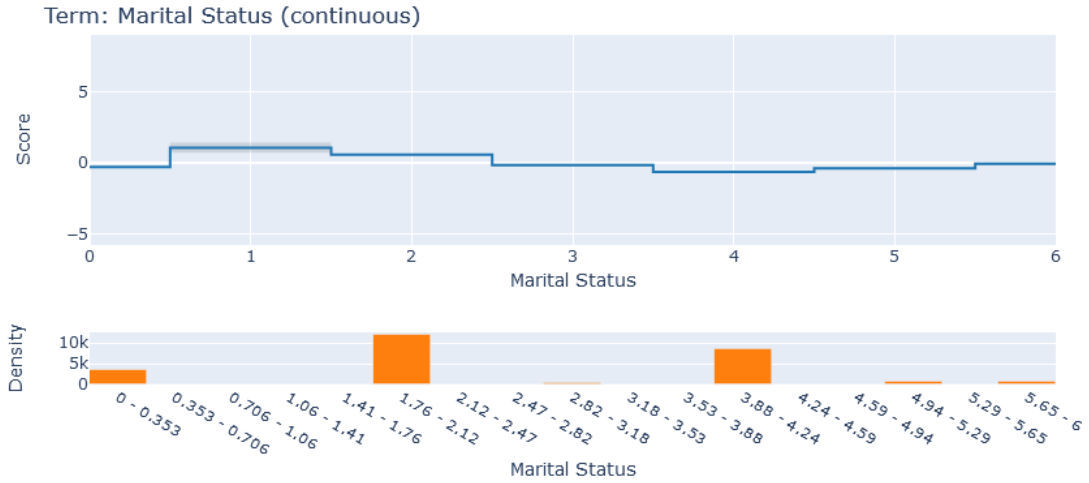


Figure 5.54: *EBM feature analysis for* ***`Martial Status`*** *in the Adult Dataset*

Figure 5.55 displays the score and density plots for `education-num`. The score plot indicates that higher education levels generally correspond with positive scores, boosting the likelihood of higher income predictions. The density plot shows a diverse distribution across education levels, providing the model with a range of patterns to interpret.

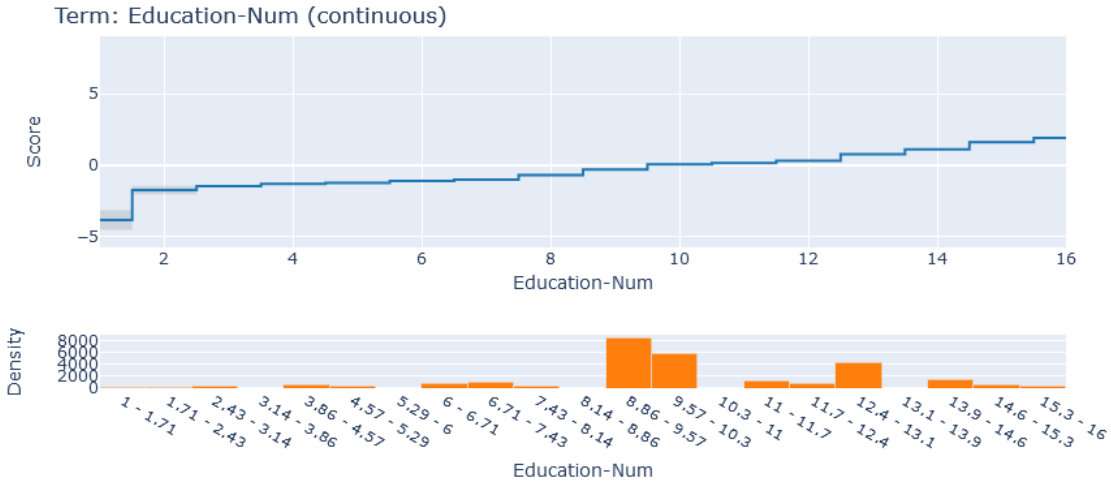


Figure 5.55: *EBM feature analysis for* ***`Education-Num`*** *in the Adult Dataset*

Figure 5.56 presents the score and density plots for `occupation`. The score plot highlights moderate variability across different occupations, with certain roles contributing positively to higher income classifications. The density plot reflects a broad distribution, illustrating how the model interprets different occupations in the income prediction.

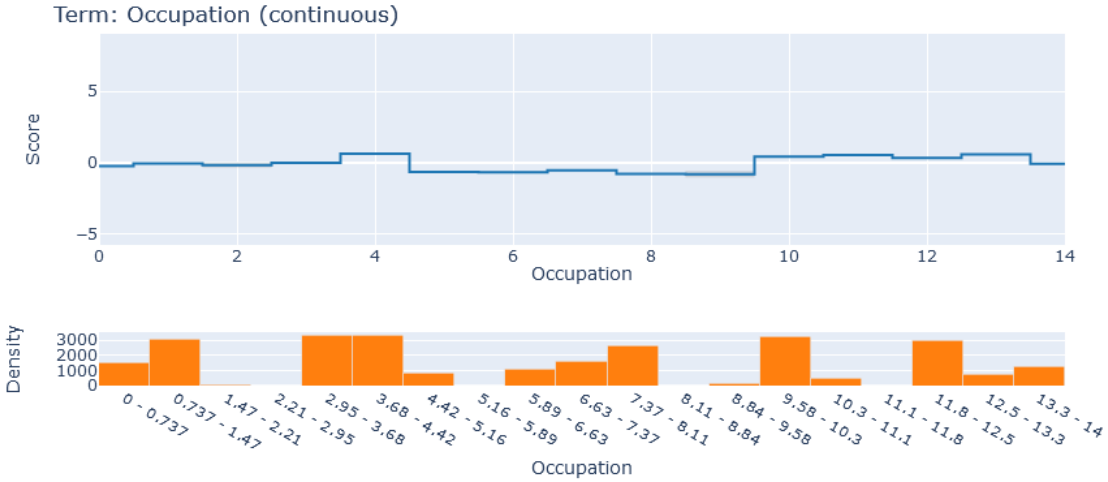


Figure 5.56: *EBM feature analysis for* `Occupation` *in the Adult Dataset*

Figure 5.57 shows the score and density plots for `sex`. The score plot displays minor differences, with slight variations between categories, indicating a low impact on income predictions. The density plot shows an even distribution, confirming that this feature contributes minimally to the model's decisions.

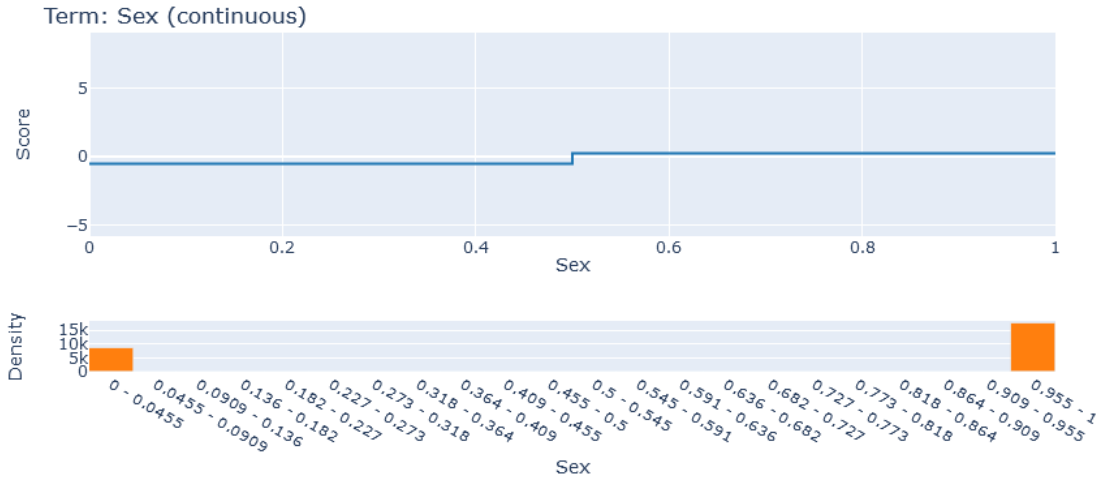


Figure 5.57: *EBM feature analysis for* `Sex` *in the Adult Dataset*

Figure 5.58 presents the score and density plots for `hours per week`. The score plot demonstrates that higher weekly hours slightly increase income predictions. The density plot shows that most data points cluster around standard working hours, indicating this feature's moderate role in classification.

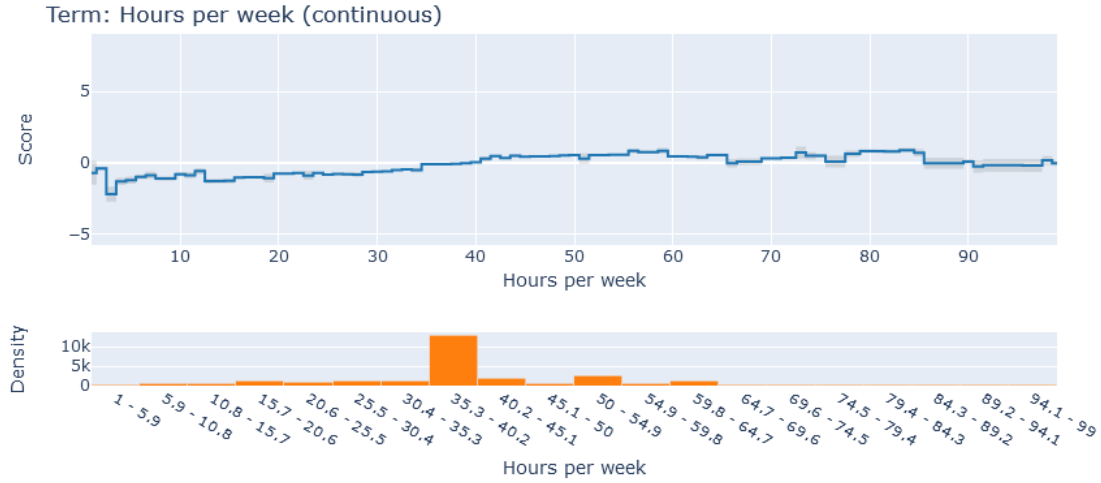


Figure 5.58: *EBM feature analysis for* ***`Hours per week`*** *in the Adult Dataset*

Figure 5.59 displays the score and density plots for `capital loss`. The score plot shows a mild positive effect for instances with capital loss, albeit less influential than capital gain. The density plot reveals that most data points have minimal capital loss, but higher values can still impact the model's predictions.

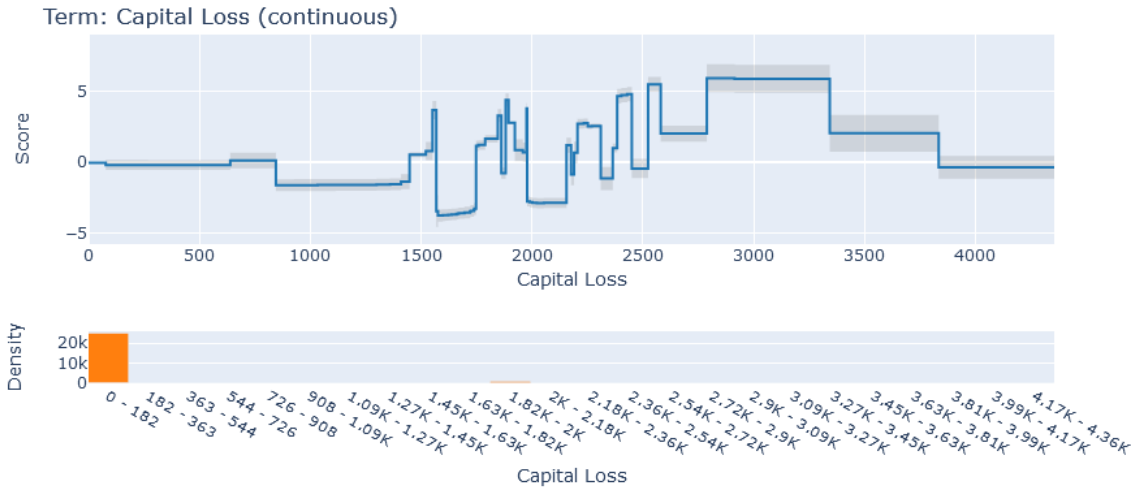


Figure 5.59: *EBM feature analysis for* ***`Capital loss`*** *in the Adult Dataset*

Figure 5.60 presents the score and density plots for `workclass`. The score plot indicates low influence across different work classes, with minimal variations in income predictions. The density plot reflects that this feature has a balanced distribution but a limited role in the model's classification process.

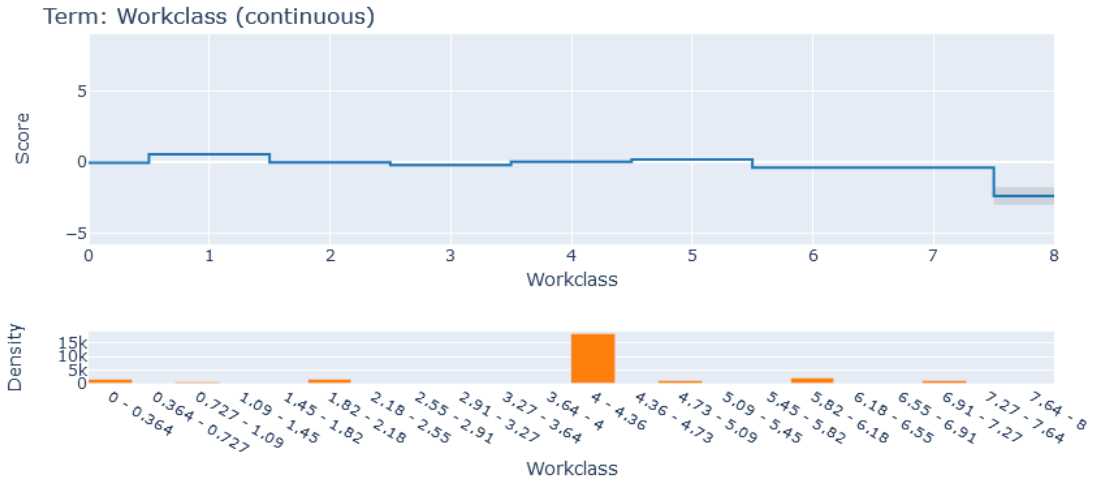


Figure 5.60: *EBM feature analysis for* `Worclass` *in the Adult Dataset*

Figure 5.61 shows the score and density plots for `race`. The score plot indicates minimal effect on income predictions, with values close to zero for all categories. The density plot shows an even distribution, confirming race as a minor factor in the model's decision-making process.

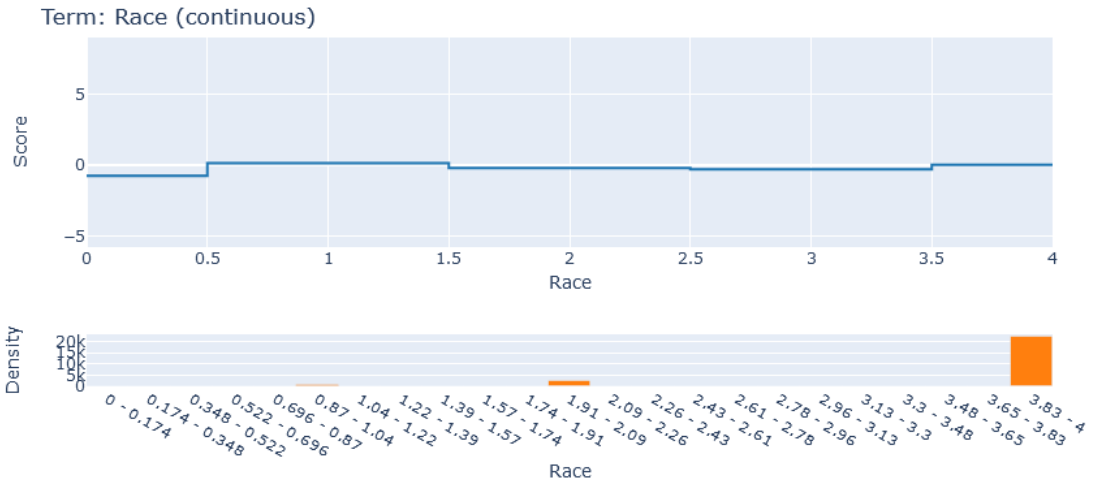


Figure 5.61: *EBM feature analysis for* `Race` *in the Adult Dataset*

Figure 5.62 displays the score and density plots for `native-country`. The score plot reveals negligible impact on income classification, with all values centered around zero. The density plot shows a scattered distribution, underscoring that native-country contributes minimally to the model's predictions.

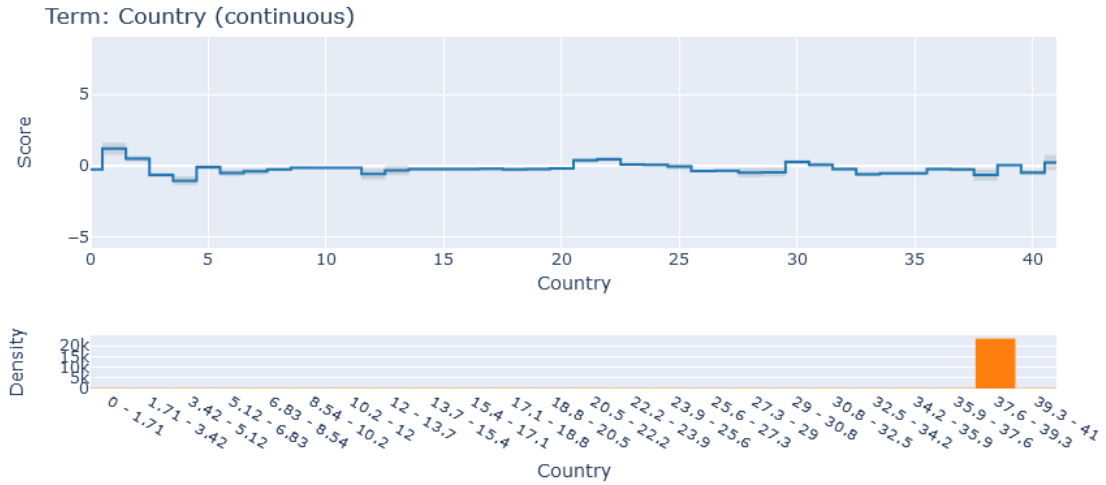


Figure 5.62: *EBM feature analysis for* `Country` *in the Adult Dataset*

Figure 5.63 displays a horizontal bar plot illustrating the contributions of key features to the prediction of a `false` class, which was correctly classified as `false`. The features `relationship` and `sex` exhibit the largest negative contributions, each with values just below -1, indicating that both features strongly support the `false` classification. Conversely, `hours per week` contributes positively with a value of +0.5, adding moderate support toward the prediction outcome. Other features, including `age`, `marital status`, and `capital gain`, contribute minor negative values around -0.3. Remaining features show minimal influence, collectively indicating that relationship and sex were the primary factors leading to the correct prediction.

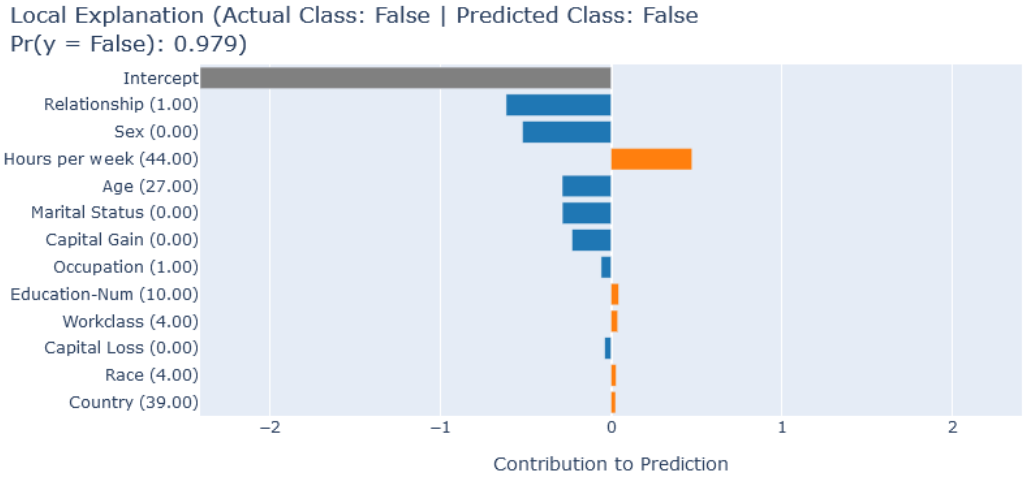


Figure 5.63: *EBM features for the Adult dataset example classified as false*

Figure 5.64 presents a horizontal bar plot of feature contributions for an instance in the `true` class that was accurately predicted as `true`. `Capital loss` has the most substantial

impact with a positive contribution nearing 5, strongly reinforcing the `true` prediction. Meanwhile, `education-num` has a negative contribution of -1, suggesting it slightly countered the prediction. `Age` contributes positively with a value of 1, lending additional support toward the final outcome. Contributions from other features are minimal, indicating that capital loss, education, and age played decisive roles in securing the accurate classification.

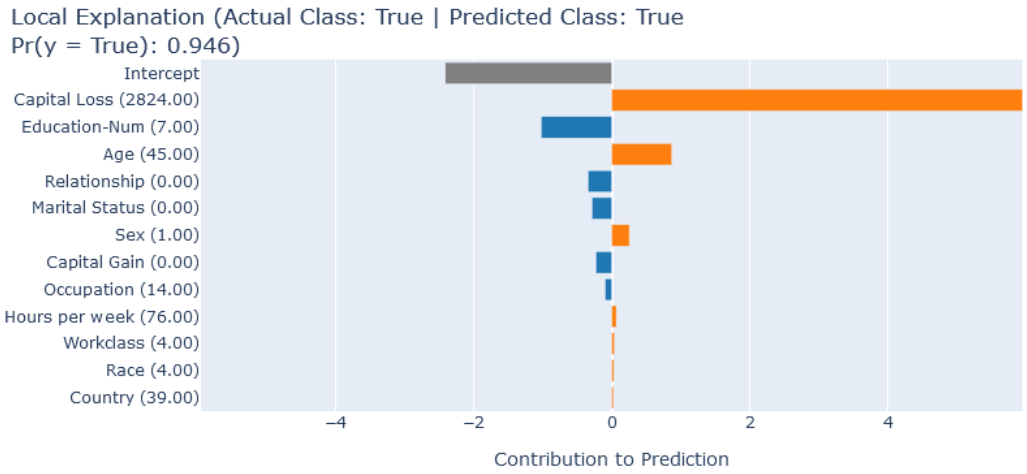
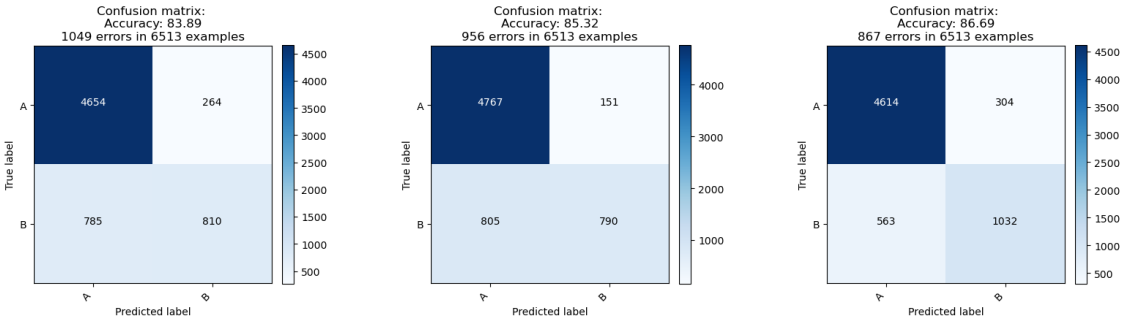


Figure 5.64: *EBM features for the Adult dataset example classified as true*

5.2.4 Discussion

In this final analysis of the Adult dataset, Figure 5.65 presents three confusion matrices, each corresponding to a different model used in the study: a DT, SHAP, and EBM. While the performance metrics of the models show minor variations, the primary objective here is not to compare scores but to examine the interpretability and insights provided by each approach in understanding the model's decision-making process.



(a) *DT* (b) *SHAP* (c) *EBM*

Figure 5.65: *Confusion Matrix for the Adult dataset*

The DT model offered a clear visualization of feature importance within the Adult dataset. The `relationship` feature consistently appeared as the most critical splitting node, indicating its strong predictive influence. This finding aligns closely with the insights from SHAP, where multiple SHAP visualizations, including the bar plot, beeswarm plot, and force plots, consistently highlighted `relationship` as the dominant factor in shaping

the model's predictions. This feature held a consistently high SHAP value, confirming its importance across various instances and interactions within the dataset.

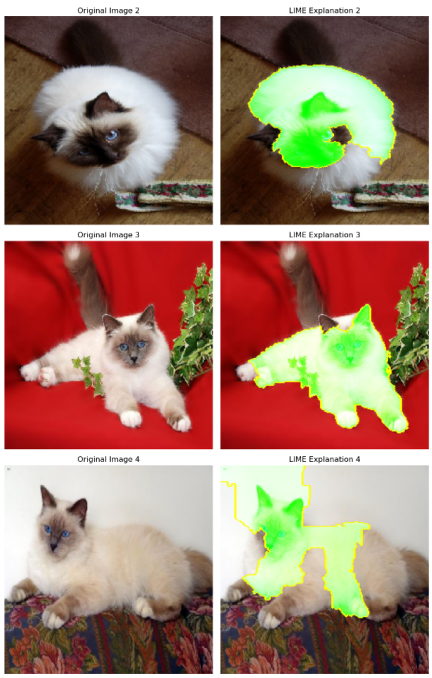
EBM reinforced findings with its feature weight analysis, highlighting `relationship` as the most influential feature, closely followed by `capital gain` and `education-num`. This additive approach in EBM provided a clear view of each feature's impact, simplifying the interpretation of how `relationship` and other key factors contributed to the model's predictions. While `relationship` emerged as the dominant feature, EBM also revealed the secondary roles of `capital gain` and `education-num` in influencing outcomes.

In summary, all three interpretability methods, DT, SHAP, and EBM, consistently highlighted `relationship` as the most influential feature in shaping predictions within the Adult dataset. This convergence of findings across interpretability methods reinforces the robustness of the results, demonstrating the reliability of `relationship` as a primary factor in income prediction. By focusing on interpretability, this study highlights how each model's insights work together to reveal feature contributions in the Adult dataset, with `relationship` standing out as the most influential feature across all approaches.

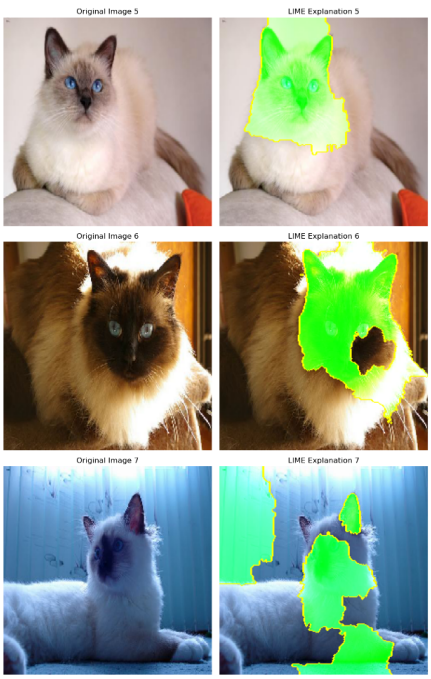
6 Cats and Dogs Dataset

6.1 LIME Explanations

The LIME analysis, as shown in Figure 6.1, provided insightful visualizations by overlaying the original image with highlighted regions that influenced the model's prediction. In the LIME overlay, specific areas of the image are marked to indicate the parts most relevant to the model's decision-making process. This comparison allows for an intuitive understanding of which features the model prioritized, revealing patterns or objects that strongly contributed to the classification outcome. Such visualization enhances interpretability, making it easier to validate the model's reasoning by directly observing the influential sections on the image.



(a) *LIME Explanations example 1*

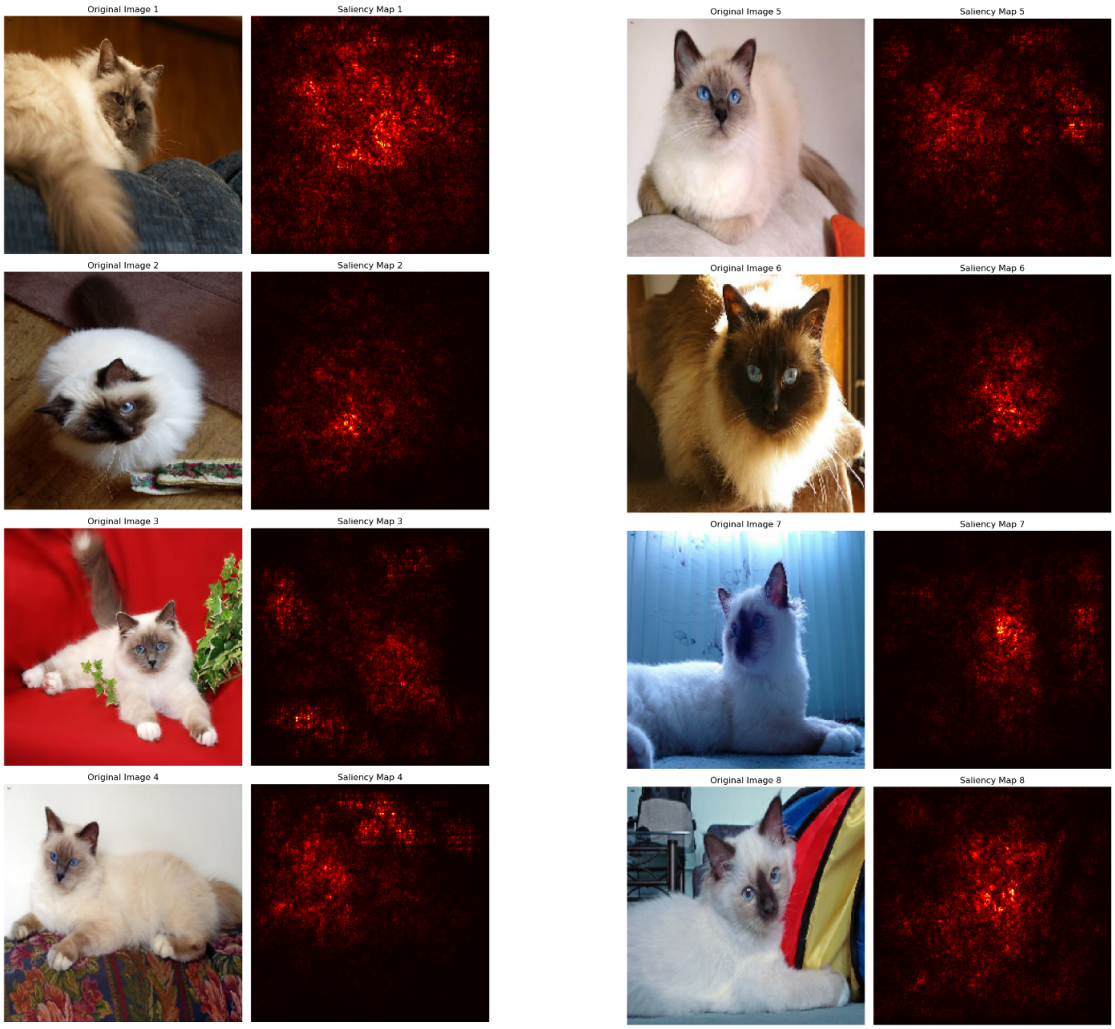


(b) *LIME Explanations example 2*

Figure 6.1: *Highlighted Feature Regions in Cats and Dogs Dataset Using LIME*

6.2 Saliency Maps

The saliency map analysis, as shown in Figure 6.2, provided a visualization that highlights the regions of the original image most influential to the model's prediction. Using a red color scheme, areas of high importance are marked, allowing for an understanding of which sections of the image the model focused on during classification. This color-coded saliency map enables a clearer interpretation of the model's decision-making process by visually emphasizing the pixels or regions that contributed most strongly to the output, thereby enhancing model transparency and aiding in result validation.



(a) *Saliency maps example 1* (b) *Saliency maps example 2*

Figure 6.2: *Highlighted Feature Regions via Saliency Maps in the Cats and Dogs Dataset*

6.3 Boosted Saliency Maps

This analysis includes four visualizations—original image, LIME overlay, standard saliency map, and boosted saliency map—each contributing unique insights into model interpretability. Figure 6.3 showcases the boosted saliency map, which significantly enhances the clarity of critical regions compared to the standard saliency map. The boosted saliency map uses intensified contrast and brightness adjustments, making it much easier for the human eye to identify the areas most influential to the model's decision. While the standard saliency map provides an initial view of important regions, the boosted version offers a more accessible and intuitive representation, improving the interpretability of the model's focus points and aiding in validation for human users.

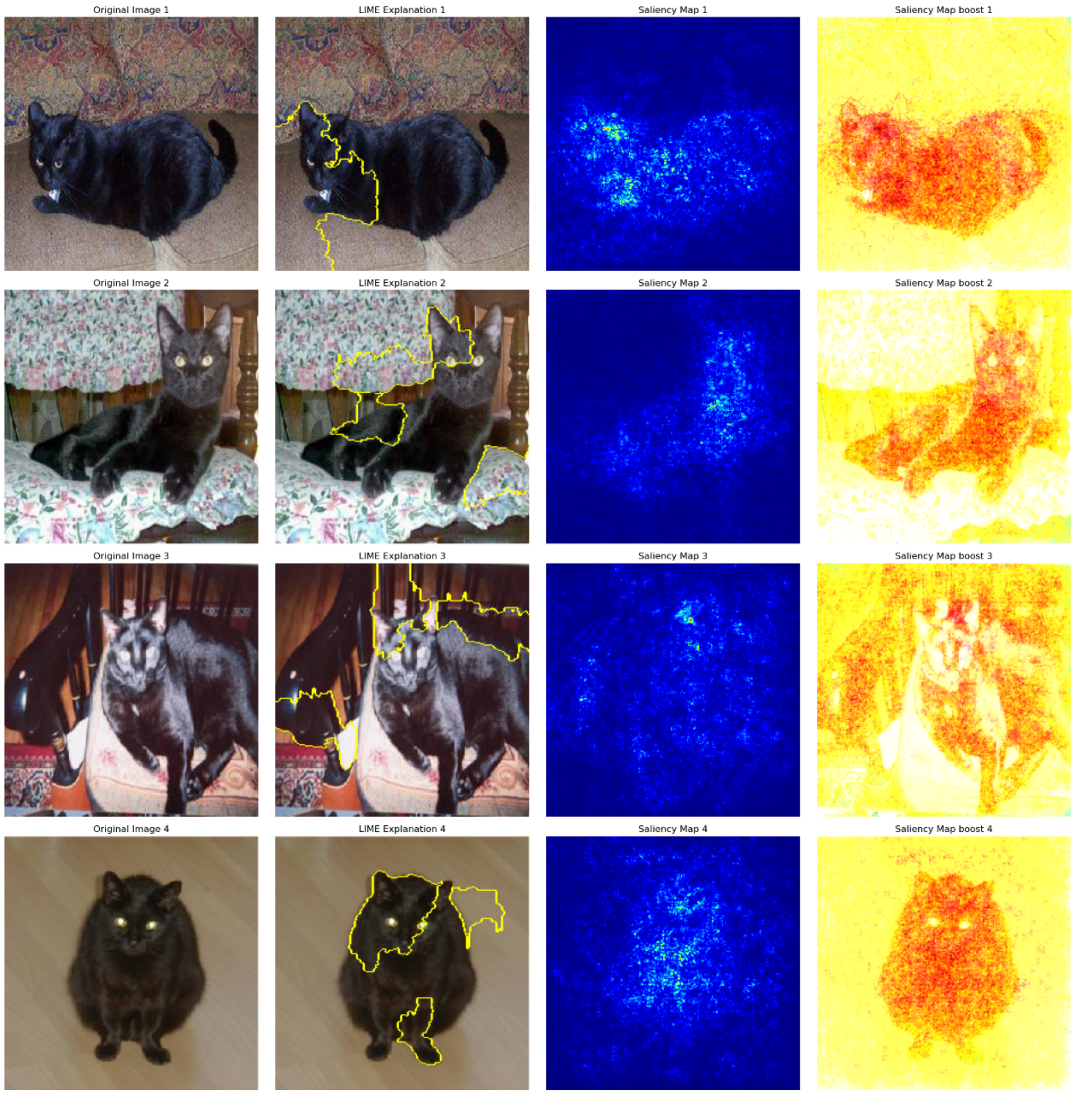


Figure 6.3: *LIME, saliency maps and saliency maps boost example 1*

The second set of visualizations, as shown in Figures 6.4, includes the original image, LIME overlay, standard saliency map, and boosted saliency map for various cat images. In this case, the boosted saliency map again proves to be a substantial improvement over the standard saliency map, offering clearer visibility of the regions most relevant to the model's classification. The enhanced contrast in the boosted version makes it easier for the human eye to discern key features in each cat image, helping to intuitively understand where the model focused. While the standard saliency map offers some insights, the boosted saliency map provides a far more accessible and detailed interpretation of the model's attention, facilitating a stronger validation of the classification results.

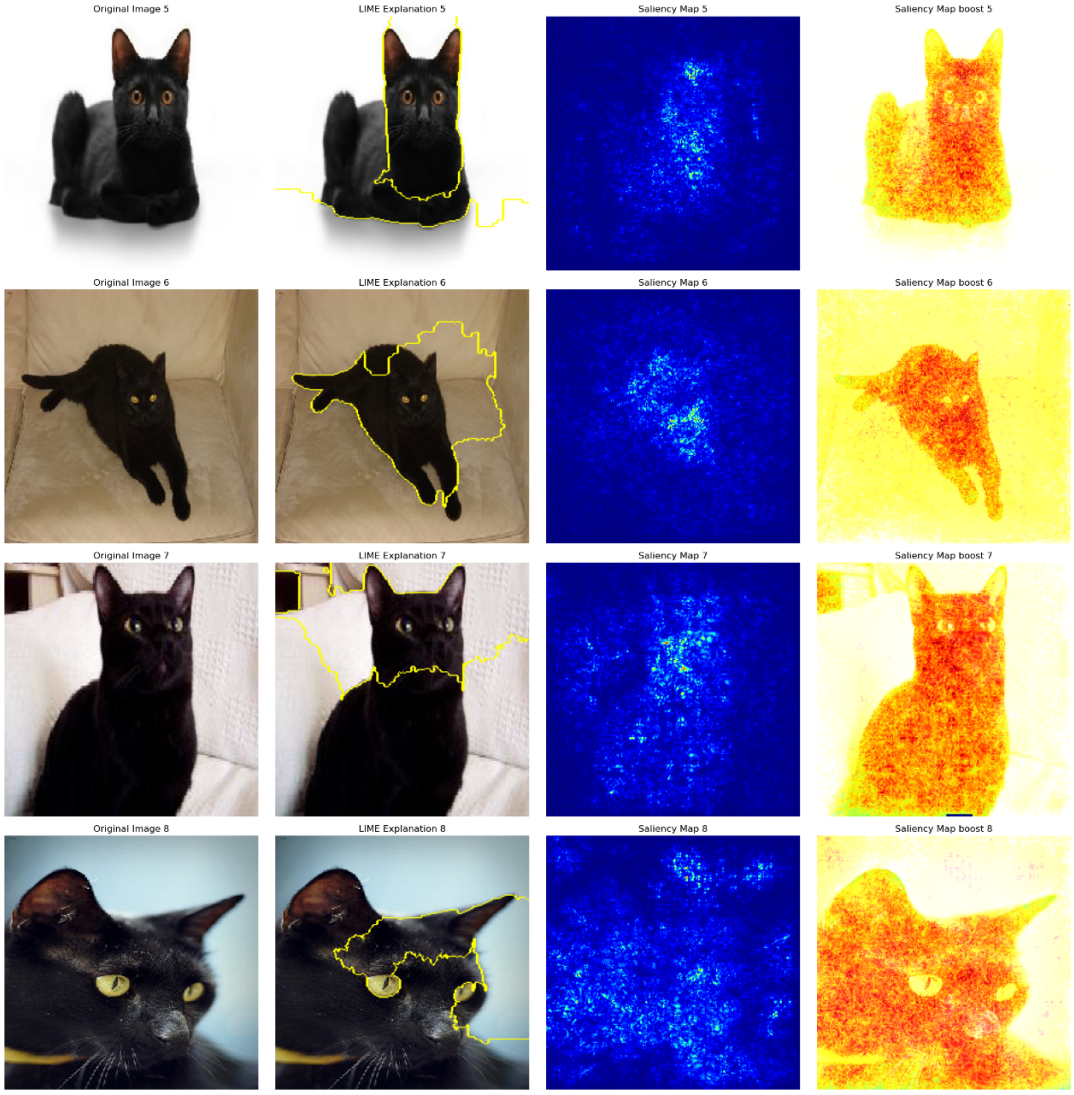


Figure 6.4: *LIME, saliency maps and saliency maps boost example 2*

6.4 Discussion

In the analysis of the Cats and Dogs dataset, Figure 6.5 presents the CM for the multiclass classification task, where each breed represents a unique class. The model achieved an impressive accuracy of 87%, reflected in the well-defined structure of the CM. The matrix shows minimal misclassifications, indicating the model's strong ability to correctly identify and differentiate between the various breeds. This high level of accuracy suggests that the model effectively captures distinguishing visual features, such as fur patterns, body shape, and facial structure, which are characteristic of each breed. The matrix's low rate of error further reinforces the reliability of the model's predictions, demonstrating that the selected interpretability methods can accurately convey the unique features that drive breed classification.

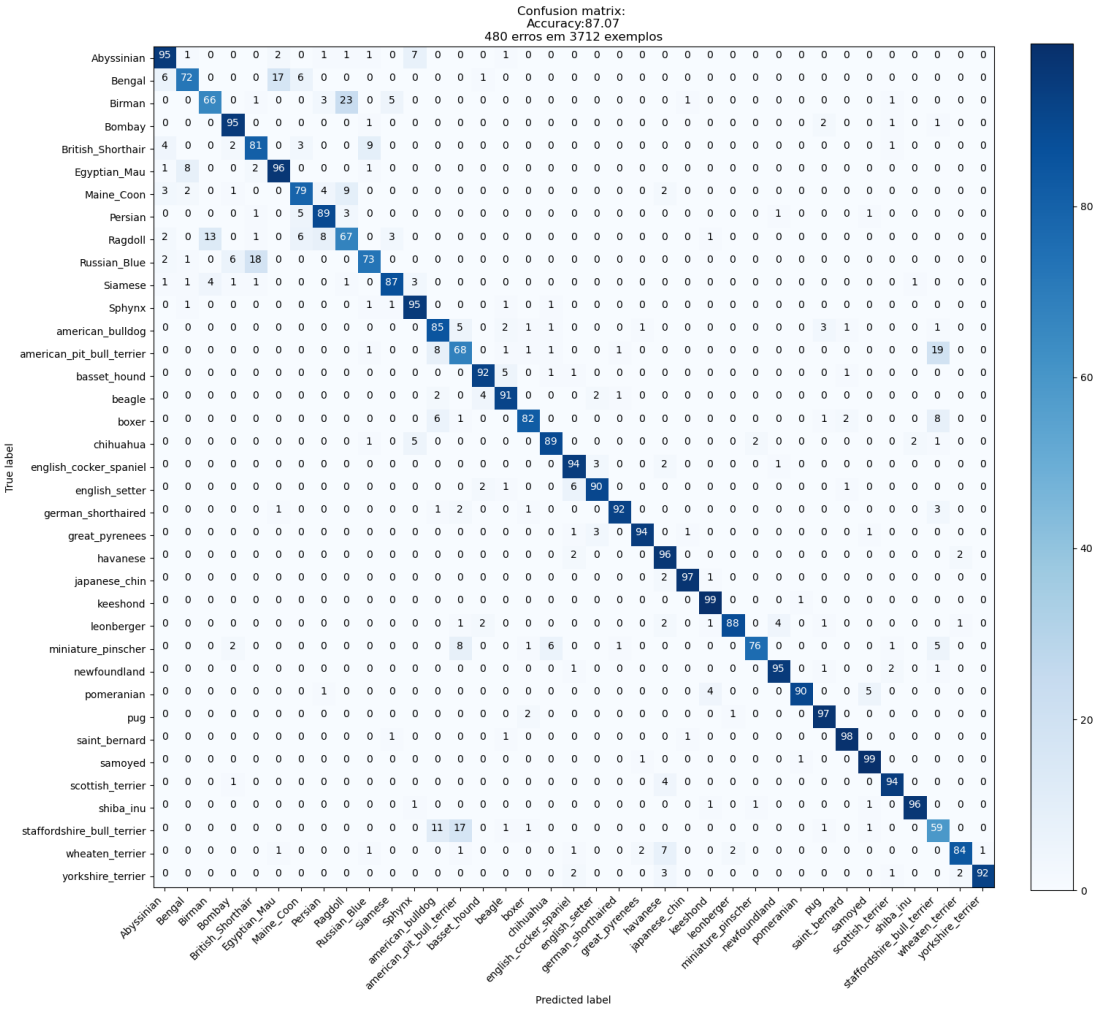


Figure 6.5: *Cats and dogs multiclass Confusion Matrix*

In the analysis of the Cats and Dogs dataset, LIME was applied to the trained model to visually highlight the most influential regions in each image, aiming to enhance interpretability by identifying the parts of the image that contributed most to each breed classification. Figure 6.6 shows these critical areas outlined in yellow, effectively pinpointing the model's focus during classification. This highlighted outline provides insight into

which visual elements—such as unique fur patterns, ear shapes, or facial features—play a decisive role in distinguishing between the various breeds. The yellow line serves as an intuitive visual aid, enabling a deeper understanding of the decision-making process in the model's classification by directing attention to specific, identifiable regions.

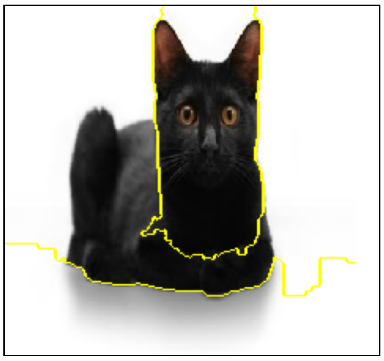


Figure 6.6: *Cats and dogs dataset LIME example*

Beyond LIME, saliency maps were also generated to capture the areas of each image that the model deemed most important for classification. Initially, the transparency of these saliency maps was set at a low level, which, while still informative, did not clearly delineate the critical areas due to limited contrast. As a result, the initial maps offered a subtler view of the features impacting classification, potentially missing finer details in complex images. To overcome this limitation, a map boost technique was introduced to amplify the saliency values, resulting in boosted saliency maps with enhanced clarity. These boosted maps, as seen in Figure 6.7, increased the visibility of high-impact regions, such as distinct fur textures or facial shapes, enabling a more precise interpretation of the features driving the model's predictions.

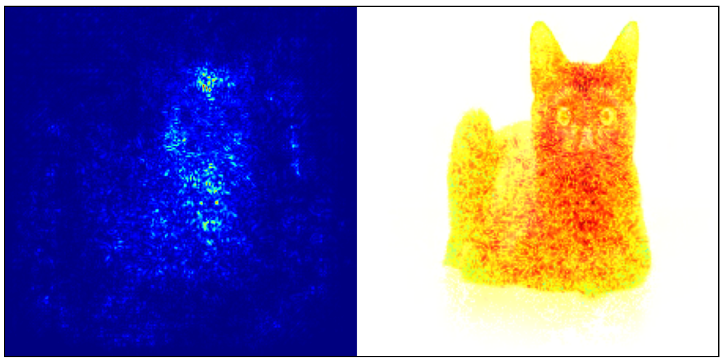


Figure 6.7: *Cats and dogs dataset saliency maps example*

Overall, these interpretability techniques support a transparent view of the model's decision-making process, which is critical in high-dimensional image data. The boosted saliency maps, in particular, offer a refined understanding that aligns closely with domain knowledge, showing how features unique to each breed guide the model's classifications. This layered approach to interpretability confirms the model's reliability in recognizing distinct breed characteristics, reinforcing trust in its predictions while offering practical insights into the visual factors that contribute most significantly to model accuracy.

7 Conclusions

In summary, this study provided valuable insights into the use of interpretability and explainability tools in machine learning. Through the application of various interpretability techniques, such as DT, SHAP, LIME, EBM, and saliency maps, it was possible to identify and understand the influential features within predictive models. Additionally, boosted saliency maps, applied in image classification tasks, further enhanced interpretability by providing clearer and more intuitive visualizations of the regions influencing model predictions.

These tools allowed for a detailed analysis across different types of datasets, uncovering the specific features driving classification decisions and highlighting the potential for a nuanced understanding of model classification behavior. The significance of interpretability in machine learning cannot be overstated. As models grow in complexity, their inner workings often become opaque, making it challenging for stakeholders to trust and validate their decisions. This study emphasized the importance of transparency, particularly in high-stakes applications where understanding a model's decision-making process is critical. By utilizing interpretability tools, this work demonstrates the potential to bridge the gap between model accuracy and transparency.

A comparative approach was taken to assess the interpretability tools, each contributing unique perspectives on feature importance. For tabular data, DT and EBM provided both global and local explanations, with DT offering straightforward, intuitive rules and EBM presenting additive feature contributions in a transparent manner. SHAP, while also providing global and local explanations, distinguished itself through its mathematically grounded framework, consistency across models, and ability to capture and visualize feature interactions. For image classification tasks, LIME and saliency maps were used exclusively for local explanations, focusing on one image at a time to identify the critical regions influencing model predictions. Among these, boosted saliency maps stood out by offering clearer and more human-intuitive visualizations compared to LIME, which was less effective in identifying key regions in images. This diversity of approaches allowed for a comprehensive analysis, with each tool validating and complementing the others' findings, strengthening the robustness of the conclusions.

Several challenges were encountered during the study, particularly in training the ResNet model for the multiclass classification of the Cats and Dogs dataset. Due to the high computational demands of ResNet, processing times were significant, requiring extended training sessions to achieve optimal performance. To manage these time constraints, it was necessary to save the trained model's weights. This approach allowed for resuming training or performing evaluations without needing to retrain the model from scratch, effectively balancing resource usage while ensuring accuracy. Additionally, the interpretability analysis for this dataset posed its own challenges, as it required applying both LIME and boosted saliency maps. While LIME provided useful localized explanations for individual instances, the boosted saliency maps excelled in highlighting critical regions with greater clarity, offering an effective means of visualizing model decisions. Saving weights proved essential not only for addressing computational demands but also for ensuring consistency during the iterative process of applying and validating interpretability techniques for large, complex datasets like Cats and Dogs.

The findings underscore the importance of interpretability in model development and encourage the adoption of explainable methods in ML practices.

7.1 Future Work

Future work could build on this study by incorporating a broader range of datasets and data modalities, including text and audio, to broaden the applicability of interpretability tools across different types of data. Expanding to complex domains such as medical datasets would test the scalability and robustness of these interpretability methods in high-dimensional, domain-specific data environments. Additionally, exploring advanced interpretability techniques, such as concept-based explanations and counterfactual analysis, would provide richer insights into model behavior and enhance understanding beyond traditional methods.

Real-time interpretability and on-the-fly explanations present another promising area of research, especially for applications that require immediate, accessible feedback. Developing techniques that offer near-instant model interpretations would be valuable in real-time settings, improving usability in high-stakes domains where timely insights are critical.

Evaluating the ethical and social implications of interpretability methods is essential as machine learning models become more integrated into decision-making processes. Future research could investigate how interpretability can promote fairness, accountability, and transparency, addressing potential biases and ensuring that models are responsibly deployed.

Moreover, further research could focus on experimenting with alternative algorithms and refining interpretability methods to overcome some of the challenges noted in this study. Additional work in real-time interpretability, faster and more accessible explanation methods, and assessments of interpretability tools across diverse domains would contribute significantly to the field, making machine learning more transparent and reliable across various applications.

References

[1] E. Alpaydin. *Introduction to machine learning*. 4th ed. The MIT Press, 2020. ISBN: 9780-262043793 (cit. on p. 1).

[2] *Backpropagation*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Backpropagation (cit. on p. 7).

[3] T. Caruana, H. Kim, and M. C. Loureiro. *EBM: Explainable Boosting Machine*. Microsoft Research. 2019. URL: https://github.com/interpretml/interpret (cit. on pp. 14, 25).

[4] *Convolutional Neural Network*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Convolutional_neural_network (cit. on pp. 9, 25).

[5] *Decision Tree*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Decision_tree (cit. on p. 25).

[6] *Deep Neural Networks*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks (cit. on p. 8).

[7] *Explainable Artificial Intelligence*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Explainable_artificial_intelligence (cit. on pp. 5, 10).

[8] *Feedforward Neural Network*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Feedforward_neural_network (cit. on p. 7).

[9] R. A. Fisher. *The Iris Dataset: A Benchmark in Pattern Recognition*. University of Cambridge. 1936. URL: https://archive.ics.uci.edu/ml/datasets/iris (cit. on pp. 25, 27).

[10] *Gated Recurrent Unit*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Gated_recurrent_unit (cit. on p. 9).

[11] O. V. G. Group. *The Cats and Dogs Dataset for Image Classification*. University of Oxford. 2012. URL: https://www.robots.ox.ac.uk/~vgg/data/pets/ (cit. on p. 28).

[12] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, D. Pedreschi, and F. Giannotti. "A Survey of Methods for Explaining Black Box Models". In: *ACM Computing Surveys* (2019). URL: https://doi.org/10.1145/3236009 (cit. on p. 1).

[13] K. He, X. Zhang, S. Ren, and J. Sun. *Deep Residual Learning for Image Recognition*. Microsoft Research. 2016. URL: https://arxiv.org/abs/1512.03385 (cit. on p. 23).

[14] M. Kim, B. W. Kim, and J. Lee. *TCAV: Testing with Concept Activation Vectors*. Google Brain. 2018. URL: https://github.com/tensorflow/tcav (cit. on pp. 11, 17).

[15] *Knowledge Distillation*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Knowledge_distillation (cit. on pp. 11, 18).

[16] *Long Short-Term Memory*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Long_short-term_memory (cit. on p. 9).

[17] S. M. Lundberg and S.-I. Lee. *SHAP: SHapley Additive exPlanations for Model Interpretability*. University of Washington. 2017. URL: https://github.com/slundberg/shap (cit. on pp. 12, 25).

[18] C. Molnar. *Interpretable Machine Learning - Counterfactual Explanations*. 2023. URL: https://christophm.github.io/interpretable-ml-book/counterfactual.html (cit. on p. 20).

[19] C. Molnar. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2024. URL: https://christophm.github.io/interpretable-ml-book/ (cit. on pp. 16, 25).

[20] *Neural Network (machine learning)*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Neural_network_(machine_learning) (cit. on p. 6).

[21] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. *The Oxford-IIIT Pet Dataset*. Visual Geometry Group, University of Oxford. 2012. URL: https://www.robots.ox.ac.uk/~vgg/data/pets/ (cit. on p. 30).

[22] *Recurrent Neural Network*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Recurrent_neural_network (cit. on p. 9).

[23] U. M. L. Repository. *The Adult Dataset: Income Prediction from Census Data*. University of California, Irvine. 1996. URL: https://archive.ics.uci.edu/ml/datasets/adult (cit. on pp. 25, 27).

[24] M. Ribeiro, S. Singh, and C. Guestrin. *LIME: Local Interpretable Model-agnostic Explanations*. University of Washington. 2016. URL: https://github.com/marcotcr/lime (cit. on pp. 13, 25).

[25] T. D. Science. *Model Interpretability and Explainability*. 2021. URL: https://towardsdatascience.com/model-interpretability-and-explainability-27fe31cc0688 (cit. on p. 10).

[26] B. Software. *Machine Learning Interpretability vs. Explainability*. 2023. URL: https://www.bmc.com/blogs/machine-learning-interpretability-vs-explainability (cit. on p. 10).

[27] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th ed. 2020. ISBN: 9780134610993 (cit. on p. 1).

[28] *Transfer Learning*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Transfer_learning (cit. on p. 10).

[29] *XGBoost*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/XGBoost (cit. on p. 31).