BBM467 - Blog Post Project

Porcupine

2200356002, Orhan Aytekin

Understanding Explainable AI with SHAP

Explainable AI

There are several challenges to *Explainable AI* that make it difficult to understand how a machine learning model arrives at a particular decision.

- ***Complexity of modern machine learning models***:
  These models often have many layers and involve millions of parameters, making it difficult to understand how they process data and make decisions. Additionally, some machine learning techniques, such as deep learning and neural networks, are designed to mimic the way the human brain works, which makes them even more difficult to interpret.

- ***Black Boxes***:
  Another challenge is that machine learning models can be "black boxes" that do not provide any insight into their decision-making process. These models are trained to maximize performance on a specific task, but do not necessarily consider the interpretability of their decisions.

- ***Effecting factors to a machine model***:
  Decisions made by machine learning models can be influenced by many factors, including the data used to train the model, the choice of model architecture, and the optimization algorithms used. This makes it difficult to understand how a particular decision was made and to identify any biases that might be present in the model.

Overall, the challenges of Explainable AI make it difficult to understand how machine learning models arrive at particular decisions, which can be a problem when these models are used in high-stakes situations or to make important decisions that affect people’s lives.

SHAP's Role

SHAP (SHapley Additive exPlanations) is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions (see papers for details and citations).

The Shapley value for a feature is calculated as the average marginal contribution of that feature across all possible combinations of features. The marginal contribution of a feature is the change in the prediction caused by that feature, given the presence of all other features.

SHAP Explainer

Before jumping into coding, we need to choose data and pick an Explainer and ML model.

I will use the *adult* dataset that comes within the *shap* library.

The Adult dataset is from the Census Bureau and the task is to predict whether a given adult makes more than $50,000 a year based on attributes such as education, hours of work per week, etc..

first 10 elements of the dataset

	Age	Workclass	Education-Num	Marital Status	Occupation	Relationship	Race	Sex	Capital Gain	Capital Loss	Hours per week	Country
0	39.0	7	13.0	4	1	0	4	1	2174.0	0.0	40.0	39
1	50.0	6	13.0	2	4	4	4	1	0.0	0.0	13.0	39
2	38.0	4	9.0	0	6	0	4	1	0.0	0.0	40.0	39
3	53.0	4	7.0	2	6	4	2	1	0.0	0.0	40.0	39
4	28.0	4	13.0	2	10	5	2	0	0.0	0.0	40.0	5
5	37.0	4	14.0	2	4	5	4	0	0.0	0.0	40.0	39
6	49.0	4	5.0	3	8	0	2	0	0.0	0.0	16.0	23
7	52.0	6	9.0	2	4	4	4	1	0.0	0.0	45.0	39
8	31.0	4	14.0	4	10	0	4	0	14084.0	0.0	50.0	39
9	42.0	4	13.0	2	4	4	4	1	5178.0	0.0	40.0	39

For the ML model I picked the *XGBClassifier* from *xgboost* library.

And for the Explainer, shap provides couple of them:

- ***KernelExplainer***: This is also better for complex model, but it makes assumptions about the data, such as local linearity and feature independence, which may not hold in all cases. But it is faster than shap.DeepExplainer.

- ***LinearExplainer***: Approximates the model's output using a linear model. This makes it faster and more accurate than the shap.DeepExplainer and shap.KernelExplainer explainers for some models, but it is less suitable for use with complex models such as deep neural networks.

- ***DeepExplainer***: Uses SHAP Kernel to approximate the model's output. This is suitable for complex models such as deep neural networks, but it can be slower and less accurate.

- ***TreeExplainer***: Approximates the model's output using a decision tree. This makes it suitable for use with tree-based models such as random forests and gradient boosting models, but it is less suitable for use with complex models.

Since I will be using XGBClassifier which is a decision tree based model, the most suitable explainer is TreeExplainer.

Model Creation

Before starting we have to install the *shap* and *xgboost* library:

```
pip install shap
pip install xgboost
```

Load the Data

```
# Load the Adult dataset
X, y = shap.datasets.adult()
# Since target values were True and False, I converted them to 0 and 1's
y = y.astype(int)
```

Create and train the model

```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create a model
model = xgboost.XGBClassifier(objective='binary:logistic', n_estimators=10)
# Train the model on the training data
model.fit(X_train, y_train)
```

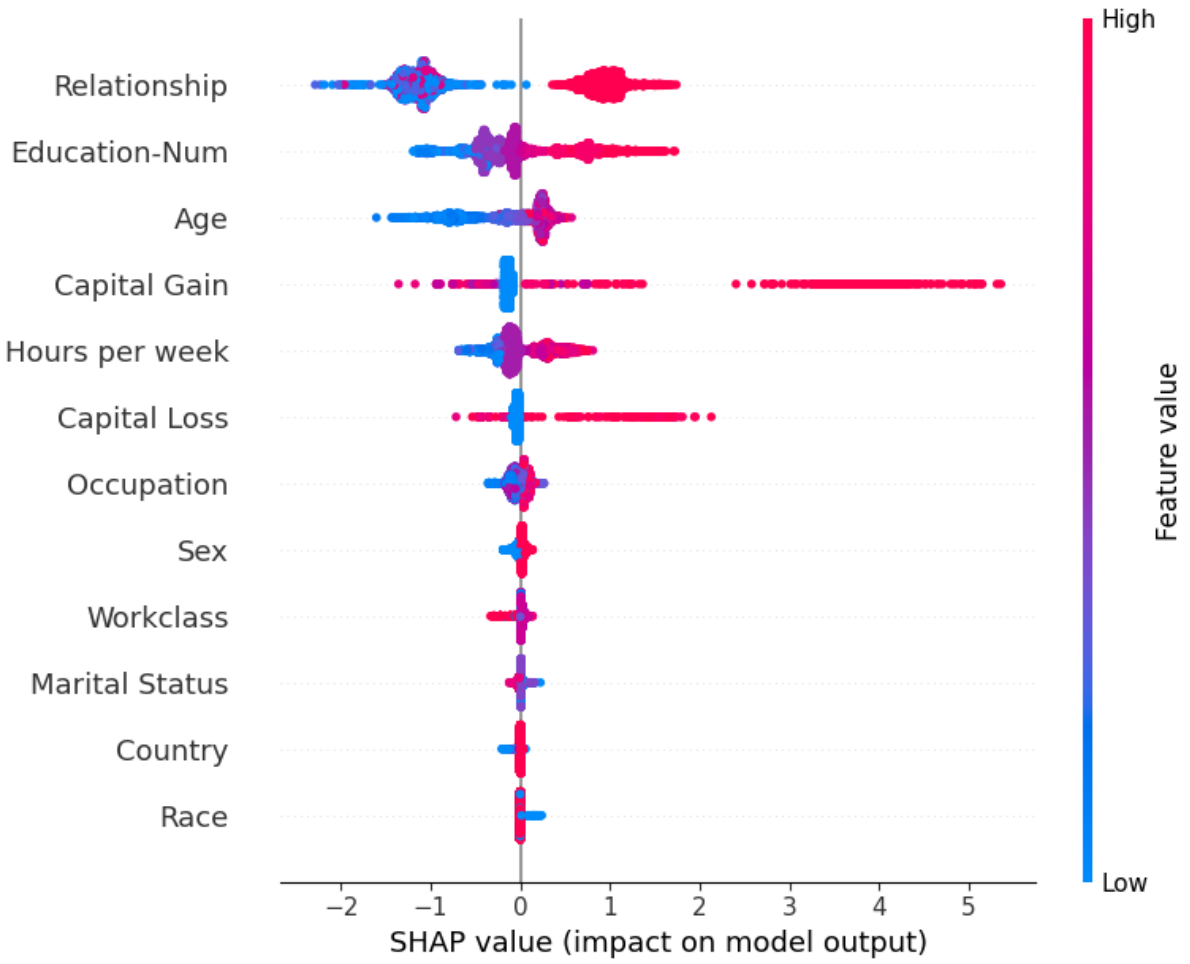
Create an TreeExplainer object and assign shap values

```
# Explain the model's predictions using SHAP values
explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)
```

EVALUATIONS

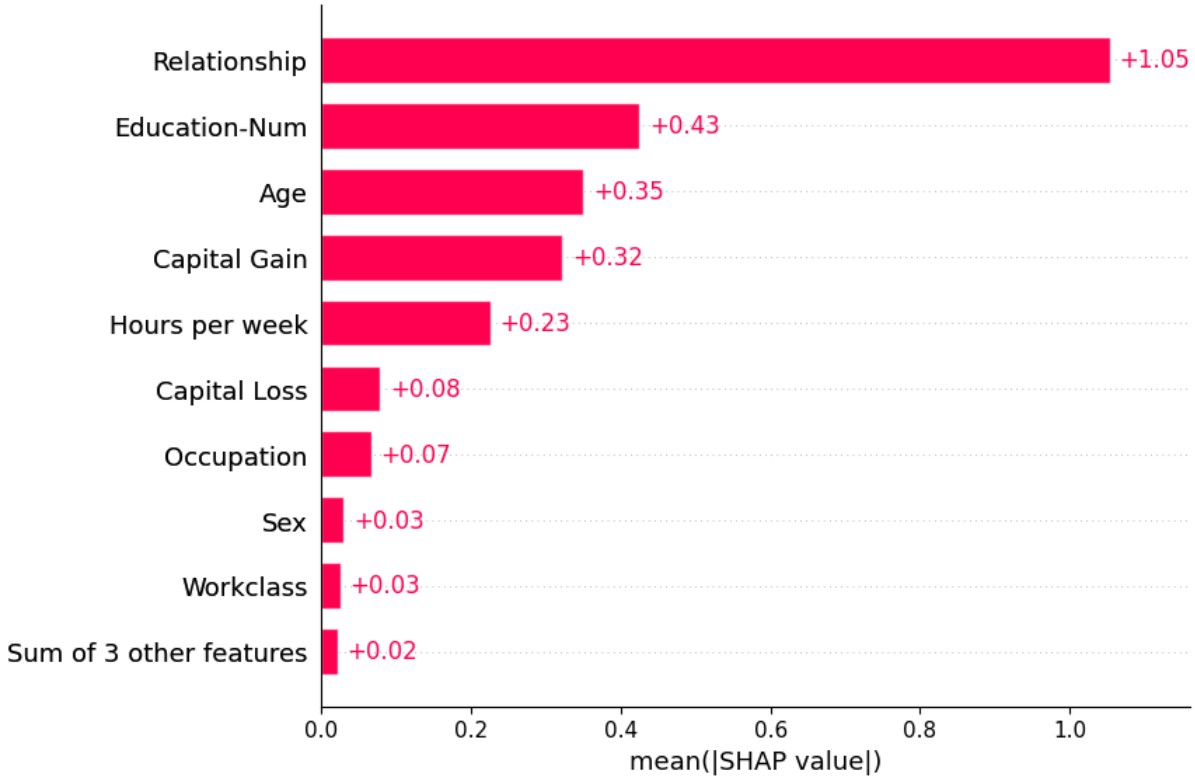
Visualizing Global Features

```
shap.summary_plot(shap_values, X_test)
```



Summary plot let's us see how each feature contributes to the model's predictions.
We can interpret from here that, *Relationship* status has the biggest effect while predicting the annual income and *Race* is the least effective.

```
shap.plots.bar(explainer(X_test))
```



This plot shows the mean SHAP value for each feature.

Visualizing Local Features

To make a local visaulization, we can choose a specific index, then using the *force_plot* we can see that which features effected the decisions and how much they effected while predicting the output.

```
# initalize JavaScript for visualizatiom
shap.initjs()
# Select a specific index
start_index = 5
end_index = 6
# Reconstruct the explainer for the selected index
exp = shap.TreeExplainer(model)
s_v = exp.shap_values(X_test[start_index:end_index])
shap.plots.force(explainer.expected_value, shap_values[5],
X_test[start_index:end_index])
```

We can see from here that, how much each attribute effected the prediction.
For example *Relationship* = 4 is being an own-child, we can see that it had a positive effect while determining the annual income. And this individual completed 14 years of education which also has positive effect.
Since *Capital Gain* and *Capital Loss* are 0, these effected the result negatively.

Dependence plot is used for checking a feature's dependence to another featue.
From this plot we can interpret that if the Age is less, there is more chance that the person is unmarried (*Relationship* = 5) and with more age there is more chance that the person will be husband (*Relationship* = 2) or a wife (*Relationship* = 3).

Explainable AI in Practice

Explainable AI is important because it helps to ensure that artificial intelligence (AI) systems are trustworthy, accountable, and fair. By understanding and interpreting the decisions and predictions made by AI systems, we can identify and address potential biases or errors in the data or algorithms used by the AI system, and we can also ensure that the AI system meets legal and regulatory requirements, such as those related to data privacy and fair treatment of individuals.

In addition, Explainable AI can help to build trust and engagement with users by providing transparent and understandable explanations of how an AI system works and why it is making certain decisions or predictions. This is especially important in domains where it is important to understand the factors that influence decisions or predictions made by an AI system, such as in healthcare or finance.

References

1. https://machinelearningmastery.com/imbalanced-classification-with-the-adult-income-dataset/#:~:text=The%20Adult%20dataset%20is%20from,work%20per%20week%2C%20etc.
2.https://github.com/slundberg/shap/issues/1460
3.https://youtu.be/-taOhqkiuIo
4.https://shap.readthedocs.io/en/latest/
5.https://shap.readthedocs.io/en/latest/generated/shap.plots.partial_dependence.html
6.https://www.birlasoft.com/articles/demystifying-explainable-artificial-intelligence
7.https://towardsdatascience.com/explainable-ai-in-practice-6d82b77bf1a7
8.https://youtu.be/VB9uV-x0gtg
9.https://www.researchgate.net/publication/358579179_The_Shapley_Value_in_Machine_Learning