Random Forest, Decision Tree

Why

1. They are non-linear ML models
2. Used for mainly three purposes, feature selection, regression, and classification
3. They are able to handle both categorical and numerical data
4. Able to handle high dimensional data with many features
5. Decision Trees are simple to understand and interpret
6. Random Forest, an ensemble method, combines the predictions of multiple decision trees to improve the overall performance and reduce overfitting.
7. Random Forest algorithm can handle missing values and outliers.
8. In a random forest, data instances are selected randomly from the data set, and this process is called bootstrapping.
9. It performs Bagging (Bootstrap aggregating) which
    a. minimizes the overfitting of data
    b. improves the model's accuracy
    c. deals with higher dimensional data efficiently
10. Random forest was used to select the features using Feature Importance
11. The classification was performed using the RandomForestClassifier and Decision Tree Classifier

How

1. RF and DT classifiers have several parameters
2. Parameters vary according to dataset
3. RandomizedSearchCV was utilized to determine the best parameters instead of GridSearchCV

Computational efficiency	For large parameter space RandomizedSearchCV is more computationally efficient than GridSearchCV
More efficient sampling	RandomizedSearchCV samples the parameter space randomly and GridSearchCV explores the entire parameter space. Thus, RandomizedSearchCV can be more efficient at finding a good set of parameters
More flexibility	Allows you to specify the number of iterations and the distribution for the random sampling. Allowing more flexibility
Exploring different parameter distributions	Allows different probability distributions for the parameters, which is useful for understanding how different parameter distributions affect the model performance

Best parameters for the Decision Tree Classifier and Random Forest Classifier:
DecisionTreeClassifier *(criterion= 'gini', max_depth= 5000, min_samples_leaf= 8)*
RandomForestClassifier *('bootstrap': False, 'criterion': 'entropy', 'max_depth': 3, 'max_features': 2, 'min_samples_leaf': 2, 'n_estimators': 10)*
We ran our tree-based models with the aforementioned parameters and with default parameters.

Parameters	Explanation
n_estimators	Number of decision trees in the forest
max_depth	How deep each tree can grow
min_samples_split	Number of samples required for a split to occur at internal nodes
min_samples_leaf	Number of samples required for a leaf node
max_features	Number of features to be taken into consideration when finding the best split
Gini	Measure of how often a randomly selected element would be mislabeled if it were randomly labeled according to the class labels distribution (Binary Classification)
Entropy	Measure of uncertainty or disorder in a set of data, it's calculated by summing the product of the probability of each class and the algorithm of the probability of that class. (Multi-class Classification)

- Criteria like n_estimators, max_depth, and max_features have a more significant impact than that of Gini/Entropy
- We ran the models without scaling because they do not necessitate feature scaling
- RF builds multiple decision trees and merging them to produce a final prediction
- Decision trees are created independently, using a random subset of the data and random subset of the features
- The algorithm is not based on distance measurements, it does not require scaling and can handle data with different scales and units
- DT algorithm splits the data based on the feature values, rather than the feature units

- However, in certain situations, it can be advantageous to scale the features before using them in a random forest
- If one feature has a much larger scale than the others, it can overpower the split criterion and cause overfitting
- Scaling the features can prevent this from happening and make the interpretation of feature importance more meaningful

Output

The decision tree algorithm gave us the following scores:

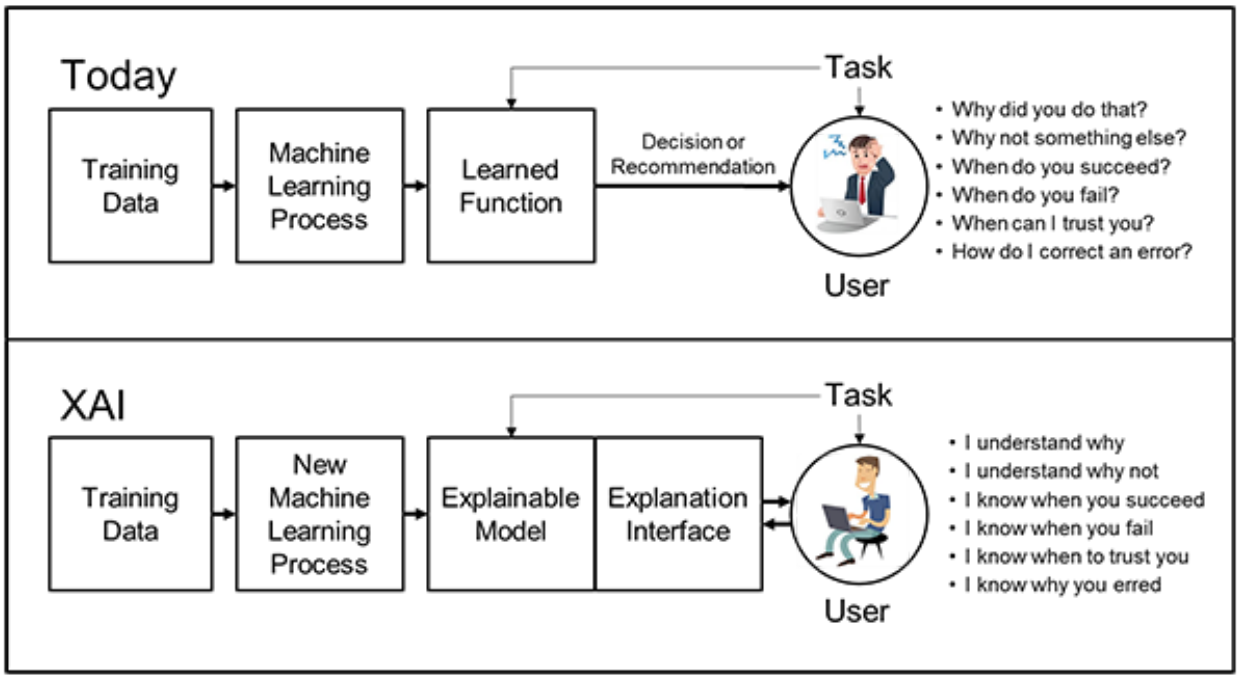
```
Accuracy of Decision Tree:  95.43
 f1_score of Decision Tree:  96.0
AUC Score of Decision Tree: 95.31
```

The random forest algorithm gave us the following scores:

```
Accuracy of rfc:  96.26
f1_score of rfc:  96.77
AUC Score of rfc: 95.92
```

XAI

- XAI, or explainable AI, is used to make AI systems more transparent and understandable to humans
- This is important in situations where the decision-making processes of AI systems have significant real-world consequences, such as in healthcare, finance, and self-driving cars.
- By providing explanations for the decisions made by AI systems, XAI can help build trust in these systems and increase their acceptability to users
- XAI can also help with debugging and improving the performance of AI systems



Lime & Shap

- Both LIME (Local Interpretable Model-Agnostic Explanations) and SHAP (SHapley Additive exPlanations) are methods used to interpret and understand how the individual features of input data contribute to a model's prediction
- LIME approximates the model locally to a specific prediction and then uses a simple interpretable model to explain it.
- It's particularly useful for models that are complex, opaque, and hard to interpret. SHAP uses a game theoretical method called Shapley values to explain the output of any model by assigning a unique contribution score to each feature and considering the interaction of all features.
- It calculates the expected value of feature importance over all possible coalitions of features.
- Both LIME and SHAP are used to increase the transparency and interpretability of a machine learning model which leads to better-informed decisions based on its predictions and builds trust in the model.

Limitations of LIME

- The approximated model generated by LIME may not be representative of the global behavior of the original model, which can lead to inaccuracies in the explanation. The choice of interpretable model used in LIME can affect the outcome, and also the results can be sensitive to the choice of parameters for this model.
- Computationally expensive, particularly when working with large datasets and complex models
- Doesn't take into account the interaction between features, which may lead to less accurate feature importance
- Less effective for image, text, and sequential data, where the global structure of the data is important
- Less effective for models that are already interpretable such as linear models or decision trees.

Limitations of SHAP

- SHAP can be computationally intensive, particularly when working with large datasets and complex models
- Relies on the assumption that feature interactions are additive, which may not always be the case
- SHAP values are not guaranteed to add up to the model's output, which can make it hard to interpret the overall importance of features
- SHAP values can be difficult to interpret for categorical variables, as they may not have a clear ordering
- SHAP does not provide an easy way to compare explanations across different instances or to compare different models
- Sensitive to the choice of reference dataset used to calculate the feature importances
- Might be less effective for models that are already interpretable such as linear models or decision trees

Why Lime is used instead of Shap

We chose to use LIME in our research because

- It is better suited for explaining the predictions of complex, non-linear models and large datasets
- SHAP is more appropriate for linear models and is computationally more demanding compared to LIME

This made LIME a more efficient choice for our research