**1.4 Tree-based method**

The following section will outline the design, analysis, and results from applying tree-based methods on the given classification dataset. Some of the approaches used are,

* Random Forest Classifier
* XGBoost Classifier
* Extra Tree Classifier

**1.4.1 Feature Scaling**:

The prominent idea of performing feature scaling is to scale the features to share same scale and reduce the outliers. **MinMaxScaler** is a normalization technique that helps to scale the input values between [0,1] whereas **Standardscaler** is a standardization technique used to center the feature columns at mean 0 and standard deviation of 1. Feature scaling is not an important attribute for tree models because they are scaling invariant and the following results prove the same.

**Without Standardization and feature selection:**

**Chart, bar chart

Description automatically generatedChart, waterfall chart

Description automatically generatedChart

Description automatically generated**

**MinMaxScaler without feature selection**:

Chart, waterfall chart

Description automatically generatedChart

Description automatically generated Chart

Description automatically generated

**Standardscaler without feature selection:**

**Chart

Description automatically generated Chart

Description automatically generatedChart

Description automatically generated**

**From the confusion matrix, all the three models worked better without performing feature scaling.**

**1.4.2 Feature Selection:**

Feature selection helps to reduce the dimenions without much loss in information. By performing feature selection across different models, it can be concluded that the attributes 1, 2 and 5 has comparatively less importance with target variable and hence it is removed to ease the process.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | XGBoost | Random Forest | Extra Tree Classifier |
| 1 | 0.05686183 | 0.0564171 | 0.06402296 |
| 2 | 0.05553203 | 0.05456134 | 0.06269189 |
| 3 | 0.16809618 | 0.1723684 | 0.15292186 |
| 4 | 0.11850529 | 0.10812387 | 0.11235779 |
| 5 | 0.05464358 | 0.05556143 | 0.06295744 |
| 6 | 0.1570057 | 0.16665671 | 0.14376379 |
| 7 | 0.13241676 | 0.13020339 | 0.1221229 |
| 8 | 0.25693861 | 0.25610775 | 0.27916138 |

Based on the results, feature selection helped to increase the efficiency in XGBoost, Random and Extra classifier.

**Without Standardization and With feature selection:**

**Chart, bar chart

Description automatically generatedChart

Description automatically generatedChart

Description automatically generated**

**Standardscaler with feature selection:**

**Chart

Description automatically generatedChart

Description automatically generatedChart, bar chart

Description automatically generated**

**MinMaxScaler with feature selection:**

Chart

Description automatically generated Chart, waterfall chart

Description automatically generatedChart

Description automatically generated

**From the results, it can be stated that by applying feature selection, performance is increased in all the three models.**

**PCA with MinMaxScaler:**

As we know, PCA helps in dimensionality reduction and scaling is important before performing PCA. Hence, we used MinMax Scaler and after performing PCA, the performance is comparatively low.

**Chart

Description automatically generated** **Chart, bar chart

Description automatically generated** Chart, bar chart

Description automatically generated

Hence, PCA is removed in further processing.

**1.4.3 Hypertuning:**

In order to reduce the problem of overfitting, the model can be calibrated by finding the right hyperparamaters. This process of generalization is called hypertuning and it led to significance increase in the accuracy of the model. The parameters that are tuned in different trees and their results are shown below.

**ExtraTree Classifier:**

Text

Description automatically generatedText

Description automatically generated

**XGBoost:**

A picture containing scatter chart

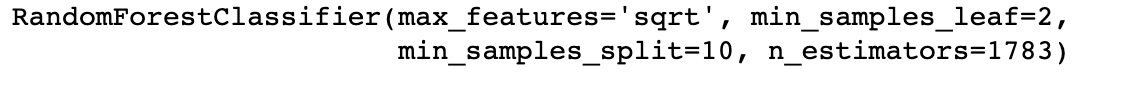
Description automatically generatedText

Description automatically generated

**Random forest:**

Text

Description automatically generated



Though GridSearchCV provides a accurate result, considering time constraints I chose to use RandomsearchCV which provided better results. By hypertuning the parameters, we can see an appropriate increase in the AUC score in both training and validation set.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **XGBoost** | **Random forest** | **Extra Tree Classifier** |
| Training AUC score | 0.9433362596673 | 0.99353692758825 | 0.98337225058859 |
| Kaggle score | 0.92005 | 0.92192 | 0.92257 |

**Conclusion:**

**Hence, it can be concluded that both Random and Extra Tree performed better with feature selection and hypertuning.**

References:

<https://towardsdatascience.com/machine-learning-step-by-step-6fbde95c455a>

<https://www.kaggle.com/code/eikedehling/extra-trees-tuning/script>

<https://www.kaggle.com/code/prashant111/a-guide-on-xgboost-hyperparameters-tuning/notebook>

<https://towardsdatascience.com/optimizing-hyperparameters-in-random-forest-classification-ec7741f9d3f6>

<https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>