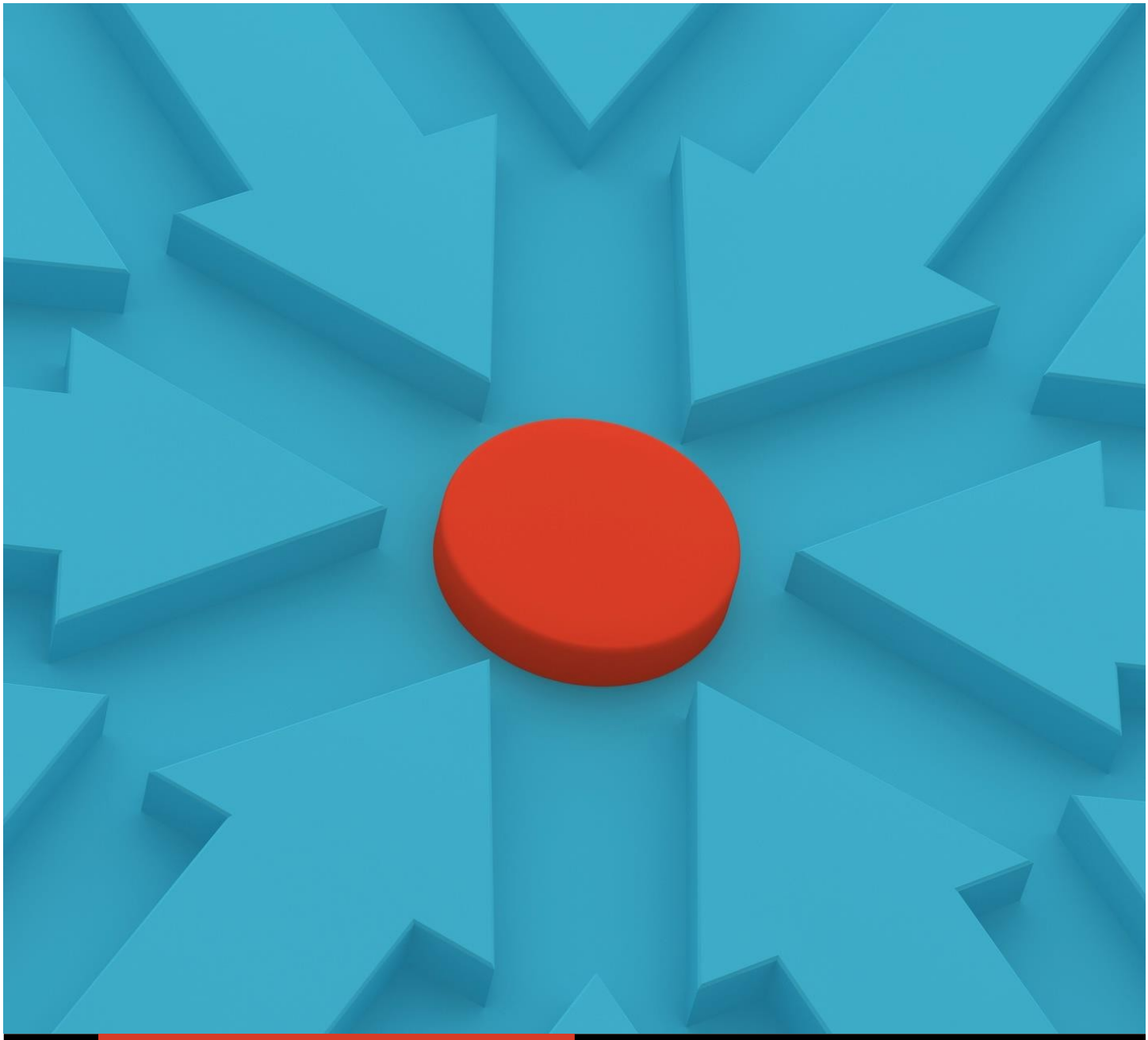


CAPSTONE BUSINESS REPORT

MRUDHULAA P V

PGP DSBA Online – March 22

26/02/2023

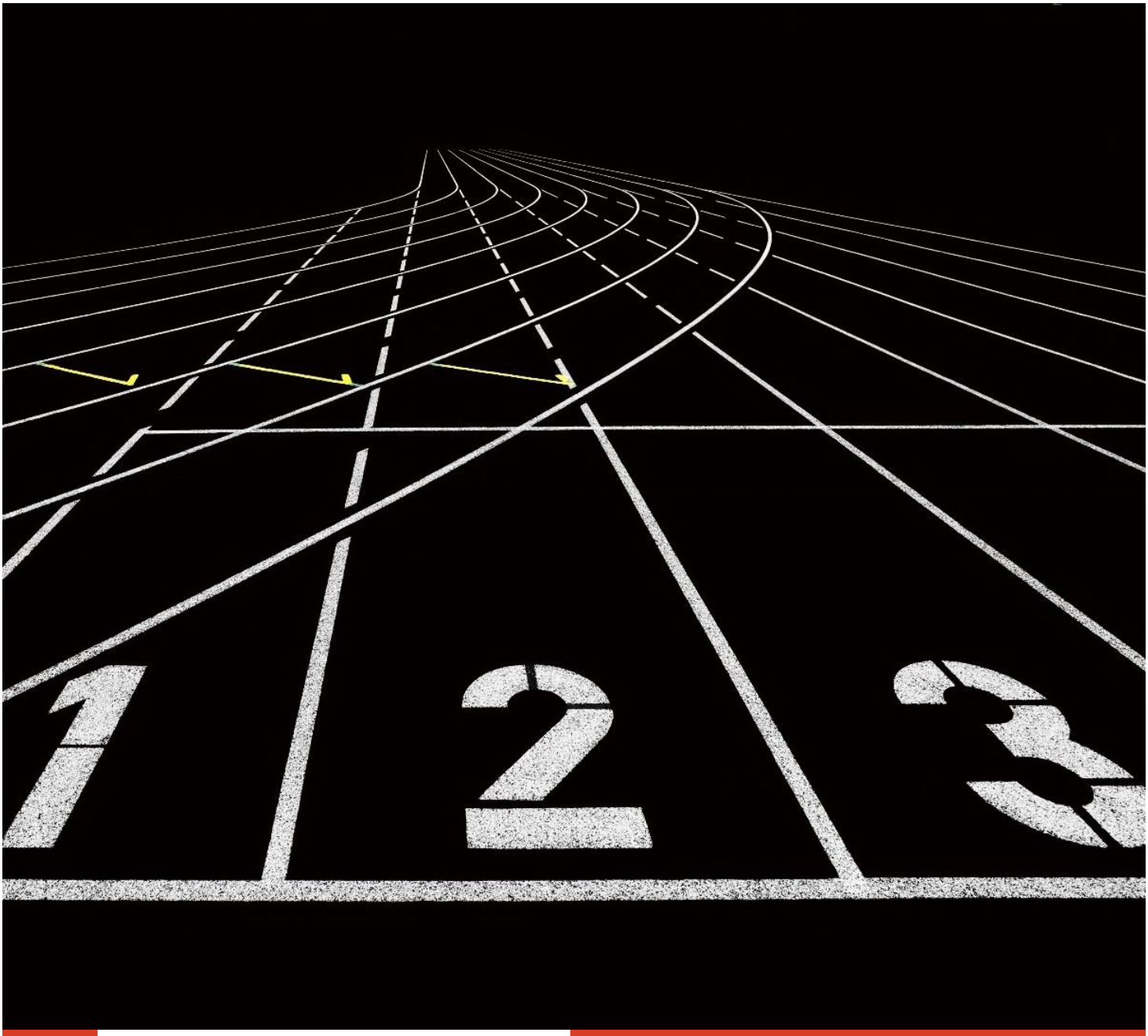


TABLE OF CONTENTS

List of figures & Tables

Model Building

Algorithms that are relevant to the problem at hand

- To predict whether a customer will churn in a business case, a binary classification problem needs to be solved using a supervised learning approach.
- This requires predicting a target variable called 'Churn,' which can have two possible outcomes.
- There are several algorithms available to solve classification problems like this, including linear methods such as Logistic Regression and Linear Discriminant Analysis, as well as non-linear methods like Decision Trees, KNN, and Artificial Neural Networks. Ensemble models like Random Forest, Adaboost, and Gradient Boost can also be used.
- However, the performance of these algorithms can vary depending on the characteristics of the data, as each algorithm makes assumptions about the data being fitted.

Comparison of Evaluation Metrics to Select the Optimal Model for Customer Churn Prediction

To find the best model, all the models' train and test data evaluation metrics need to be compared. When trying to predict customer churn, it's essential to take into account the cost related to false positives (predicting a customer will churn when they won't) and false negatives (predicting a customer won't churn when they will).

The best model should strike a balance between precision and recall. Precision measures the percentage of customers predicted to churn who do churn. A higher precision value implies that the model is accurately identifying true positives and reducing false positives. False positives are expensive for the DTH company since they may result in unnecessary retention efforts for customers who would not have churned otherwise.

Recall measures the proportion of actual churning customers who are correctly identified by the model. A high recall value suggests that the model is effectively identifying true positives and minimizing false negatives. False negatives are also costly for the DTH company as they may result in missed opportunities for retention efforts. Therefore, the best model should aim to maximize both precision and recall. The F1-score, which is the harmonic mean of precision and recall, is a useful metric to evaluate the trade-off between these two metrics and select the optimal model.

Splitting data into train and test data:

To tackle the imbalance present in the dataset, the target variable 'Churn' has a significant disparity in its categorical count, with 84% for '0' and 17% for '1'.

One possible solution is to use the SMOTE method, which generates synthetic data points to balance the distribution. However, it's crucial to apply SMOTE only to the training dataset, not the test dataset.

The initial dataset was divided into a 70:30 ratio for the training and testing datasets, but this ratio can be modified if required.

Shape of train and test dataset:

```
X_train (7700, 17)
X_test (3301, 17)
y_train (7700,)
y_test (3301,)
```

Fig.1 Shape of training and testing dataset

Building Logistic regression model using SMOTE:

We can utilise the SMOTE approach to balance the data, and once we develop a model using the balanced data, we can assess if the accuracy of the training and testing datasets has improved significantly.
We can observe that the performance is not that substantial in terms of accuracy after developing the model on a balanced dataset and assessing accuracy.

Metrics score:

```
Accuracy on training set :  0.7866233766233767
Accuracy on test set :  0.7730990608906392
Recall on training set :  0.8183925811437404
Recall on test set :  0.8064516129032258
Precision on training set :  0.4292663153627888
Precision on test set :  0.41246562786434465
F1 on training set :  0.5631480989098644
F1 on test set :  0.545785324439054
```

Fig.2 Metrics score of Logistic regression using SMOTE

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.96	0.78	0.86	6406
1.0	0.43	0.82	0.56	1294
accuracy			0.79	7700
macro avg	0.69	0.80	0.71	7700
weighted avg	0.87	0.79	0.81	7700

Fig.3 Classification report of training data

Confusion matrix of training set:

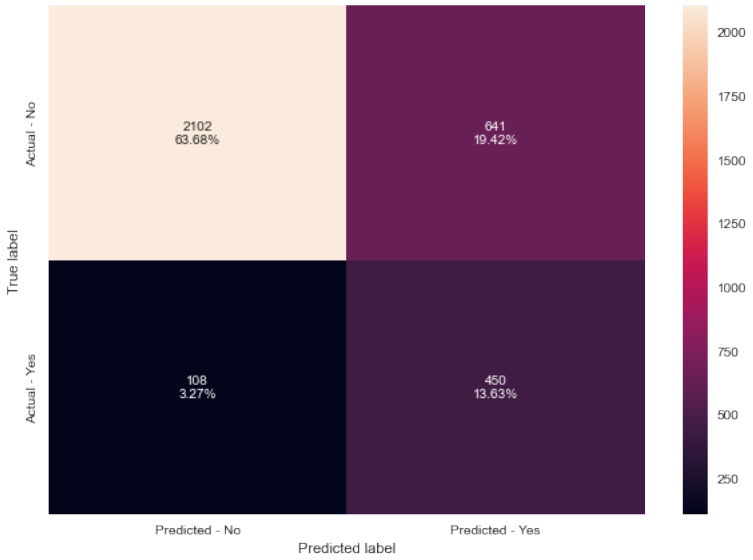


Fig.4 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.95	0.77	0.85	2743
1.0	0.41	0.81	0.55	558
accuracy			0.77	3301
macro avg	0.68	0.79	0.70	3301
weighted avg	0.86	0.77	0.80	3301

Fig.5 Classification report of testing data

Confusion matrix of testing data:

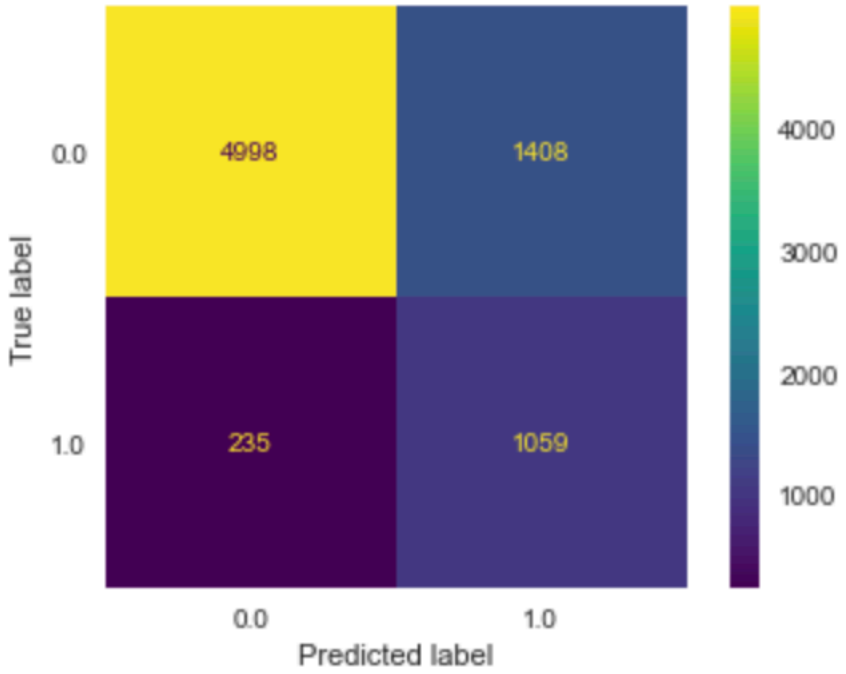


Fig.6 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 0.871
```

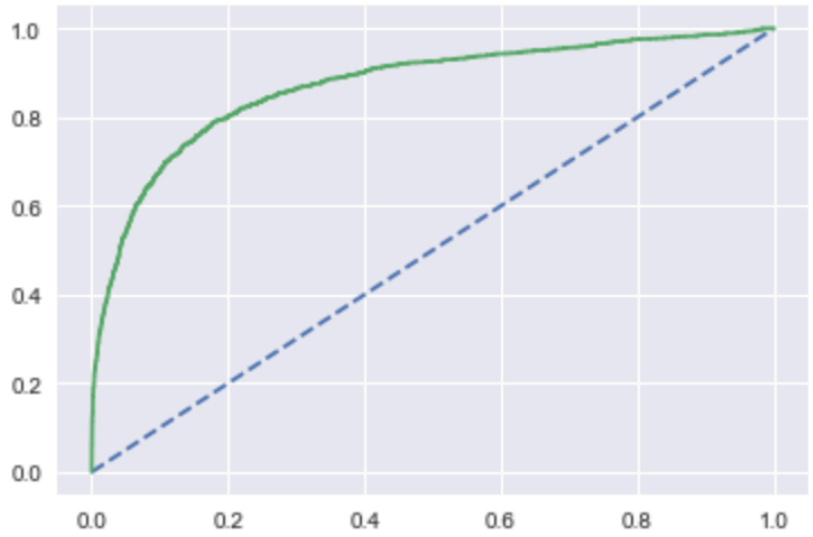


Fig.7 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.856
```

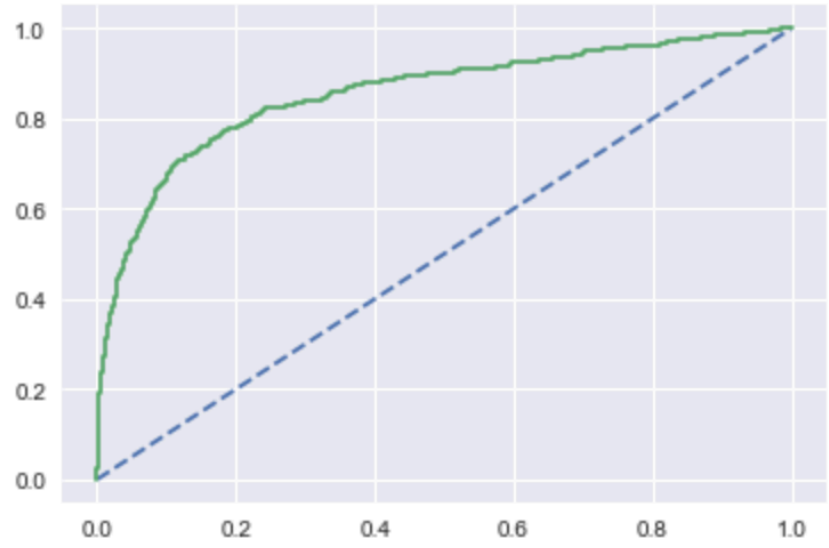


Fig.8 AUC and ROC curve of testing data

INFERENCE:

- The accuracy score of this model on the testing dataset is lower than some of the other models evaluated.
- However, the recall score is high, indicating that the model is able to identify most of the positive cases.
- The precision score is relatively low, suggesting that the model produces a moderate number of false positives.

- The F1 score is also relatively low, indicating a trade-off between precision and recall. It is important to note that SMOTE was used to balance the classes, which may have contributed to the lower precision score.
- Overall, this model performs worse than some of the other models evaluated, with lower precision and F1 score on the testing dataset.

Building Logistic regression model

After dividing the data into training and testing sets, we fitted a logistic regression model to the training dataset and used the same model to make predictions for both sets of data.

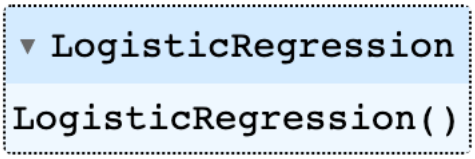


Fig.9 Logistic Regression

Metrics score:

```
Accuracy on training set :  0.7866233766233767
Accuracy on test set :  0.7730990608906392
Recall on training set :  0.8183925811437404
Recall on test set :  0.8064516129032258
Precision on training set :  0.4292663153627888
Precision on test set :  0.41246562786434465
F1 on training set :  0.5631480989098644
F1 on test set :  0.545785324439054
```

Fig.10 Metrics score of Logistic regression

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.89	0.97	0.93	6406
1.0	0.75	0.43	0.55	1294
accuracy			0.88	7700
macro avg	0.82	0.70	0.74	7700
weighted avg	0.87	0.88	0.87	7700

Fig.11 Classification report of training data

Confusion matrix of training set:

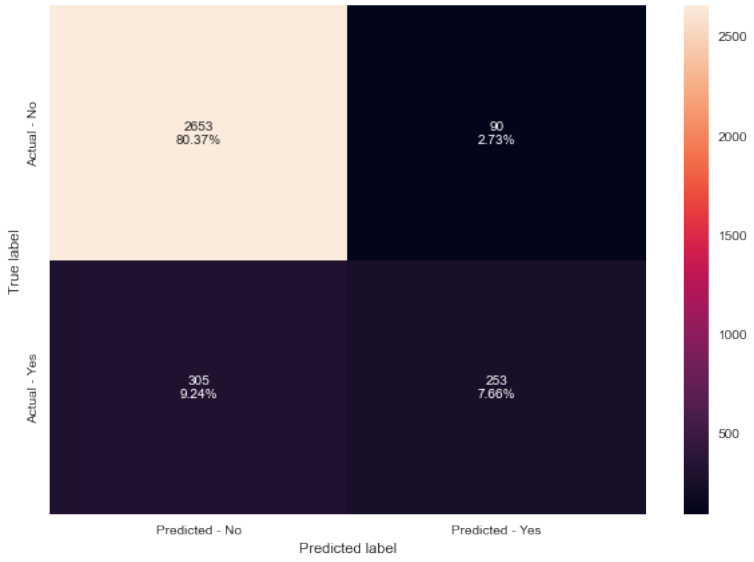


Fig.12 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.90	0.97	0.93	2743
1.0	0.74	0.45	0.56	558
accuracy			0.88	3301
macro avg	0.82	0.71	0.75	3301
weighted avg	0.87	0.88	0.87	3301

Fig.13 Classification report of testing data

Confusion matrix of testing data:

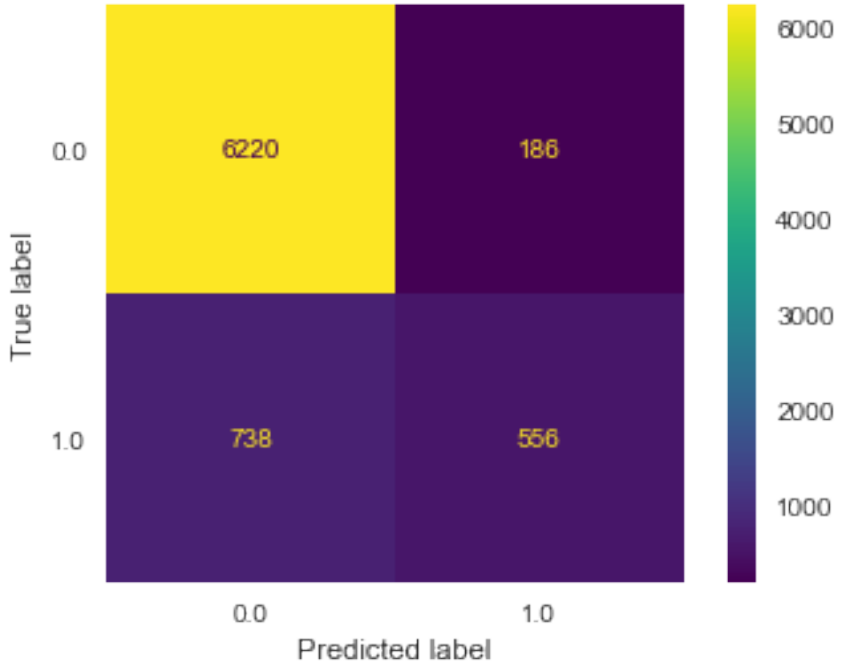


Fig.14 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.871

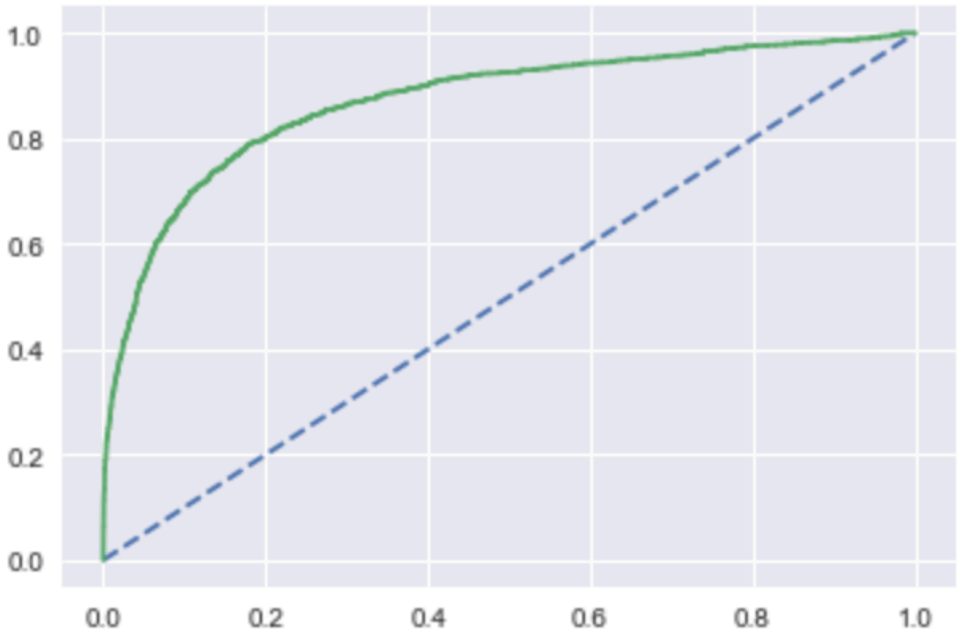


Fig.15 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.856

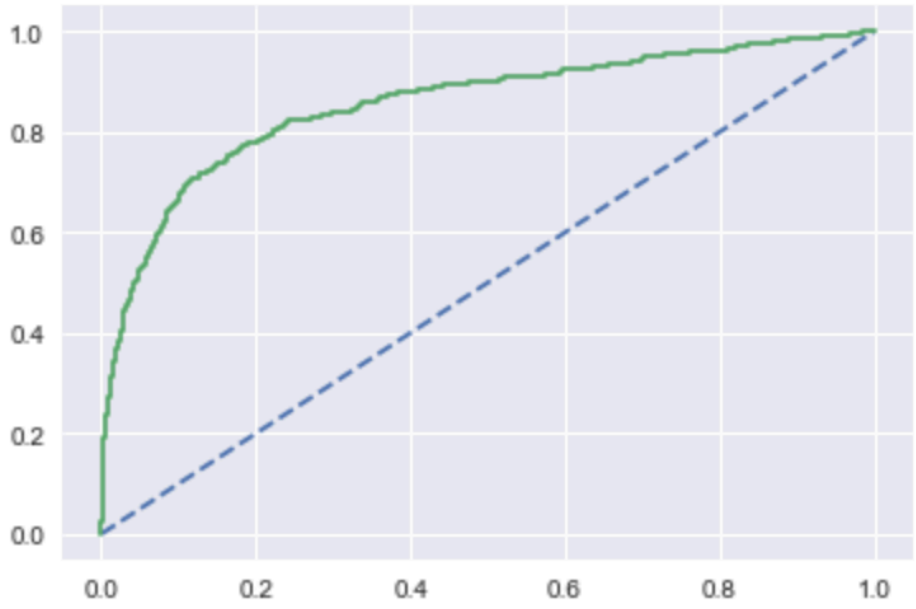


Fig.16 AUC and ROC curve of testing data

INFERENCE:

- Compared to the other models evaluated, the Logistic Regression with default parameters model has lower accuracy, recall, precision, and F1 score on the testing dataset.

- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than all of the other models evaluated, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building Logistic Regression using GridSearchCV

```
▼ GridSearchCV
GridSearchCV(cv=3, estimator=LogisticRegression(max_iter=100000,
n_jobs=2),
             n_jobs=-1,
             param_grid={'penalty': ['l1', 'l2', 'none'],
                         'solver': ['lbfgs', 'liblinear'],
                         'tol': [0.0001, 1e-06]},
             scoring='f1')
    ▼ estimator: LogisticRegression
    LogisticRegression(max_iter=100000, n_jobs=2)
        ▼ LogisticRegression
        LogisticRegression(max_iter=100000, n_jobs=2)
```

Fig.17 Hyper parameters tuning for logistic regression using GridSearchCV

```
▼ LogisticRegression
LogisticRegression(max_iter=100000, n_jobs=2, penalty='none')
```

Fig.18 Best parameters for logistic regression obtained from GridSearchCV

Metrics score

```
Accuracy on training set :  0.8805194805194805
Accuracy on test set :  0.8806422296273856
Recall on training set :  0.445904173106646 04
Recall on test set :  0.464157706093189 96
Precision on training set :  0.7397435897435898
Precision on test set :  0.731638418079096
F1 on training set :  0.5564127290260367
F1 on test set :  0.5679824561403509
```

Fig.19 Metrics score of Logistic regression

Classification report of training set:

```
Classification Report of Training Data
              precision    recall  f1-score   support

         0.0       0.90      0.97      0.93      6406
         1.0       0.74      0.45      0.56      1294

    accuracy                           0.88      7700
   macro avg       0.82      0.71      0.74      7700
weighted avg       0.87      0.88      0.87      7700
```

Fig.20 Classification report of training data

Confusion matrix of training set:

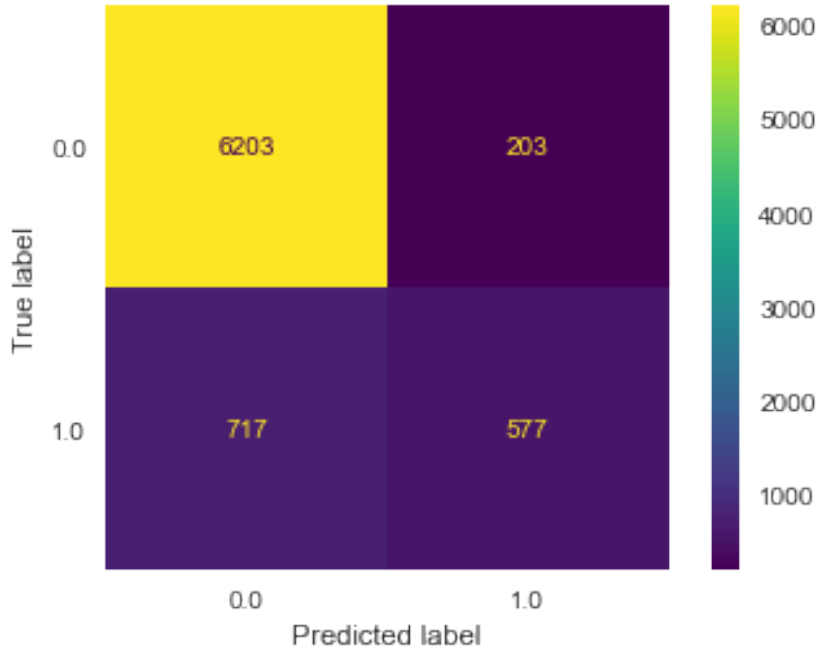


Fig.21 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.90	0.97	0.93	2743
1.0	0.73	0.46	0.57	558
accuracy			0.88	3301
macro avg	0.82	0.71	0.75	3301
weighted avg	0.87	0.88	0.87	3301

Fig.22 Classification report of testing data

Confusion matrix of testing data:

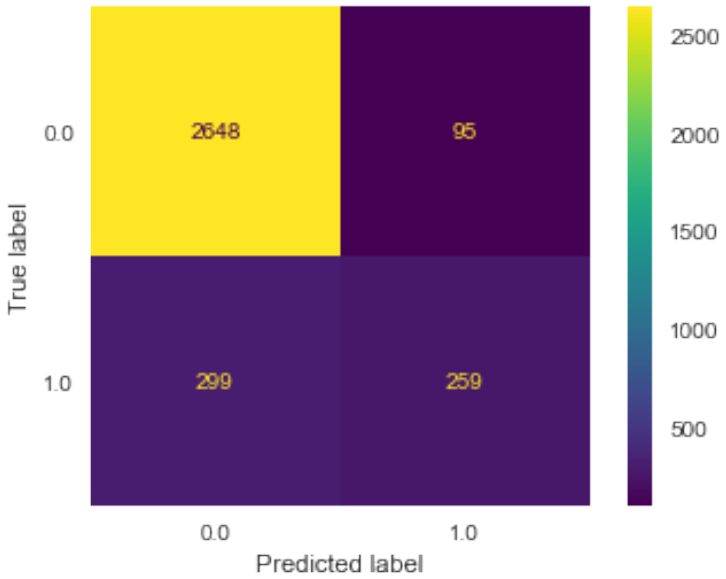


Fig.23 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.872

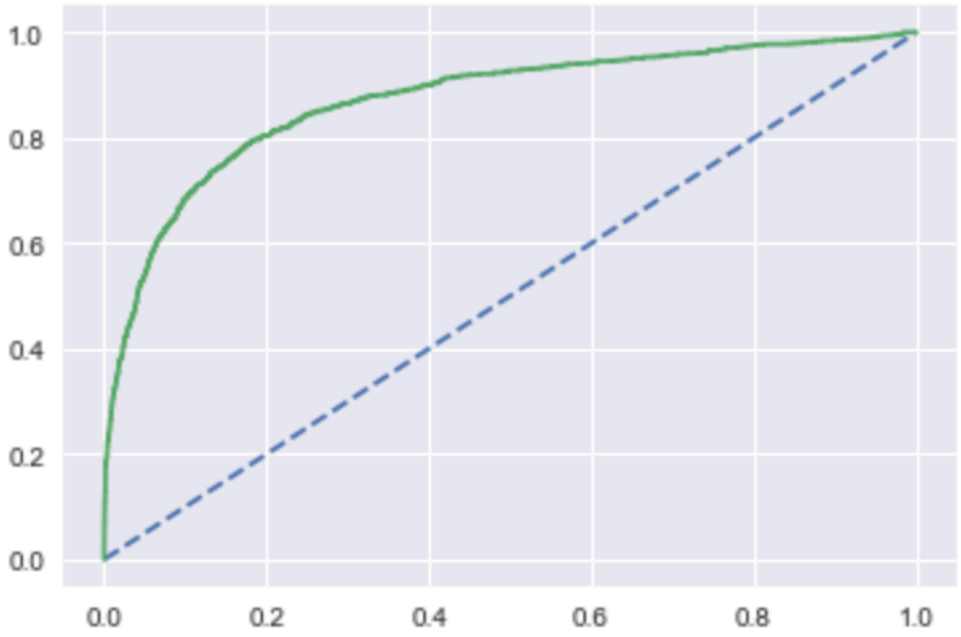


Fig.24 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.856

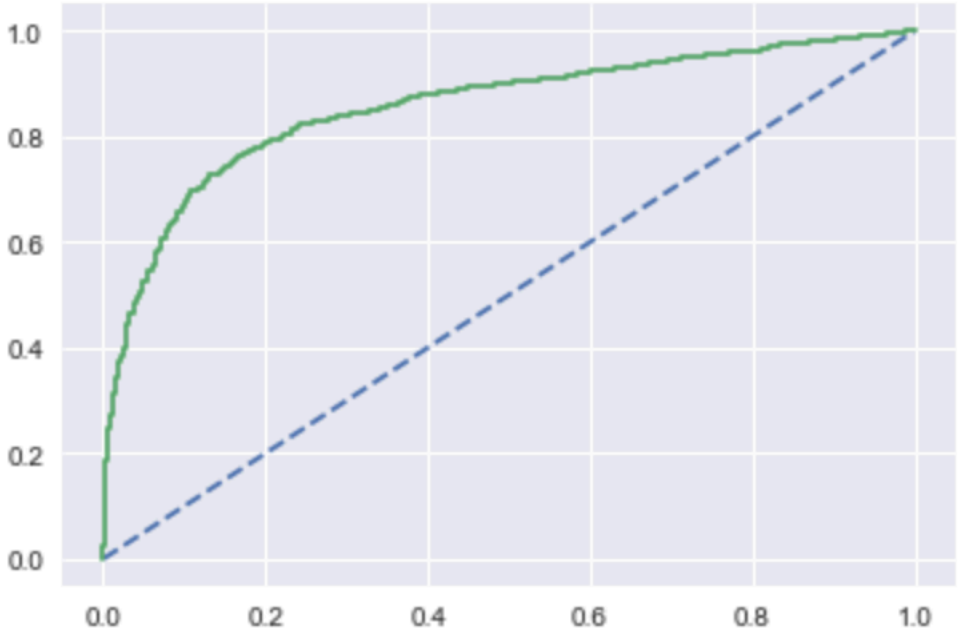


Fig.25 AUC and ROC curve of testing data

INFERENCE:

- Compared to the other models evaluated, the Logistic Regression with tuned parameters model has lower accuracy, recall, precision, and F1 score on the testing dataset.
- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than all of the other models evaluated, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building LDA model

There is also the machine learning classifier known as linear discriminant analysis (LDA). When there are linearly separable classes in the data, it performs well. Even though it makes the gaussian distribution of the underlying data an assumption, it can nevertheless perform well.

Modeling was done using Sklearn's Linear Discriminant Analysis tool.

We are developing a Linear Discriminant Analysis (LDA) model from the aforementioned training and testing datasets to see whether it can beat the logistic regression model and determine which version is optimal for future predictions.

Metrics score:

```
Accuracy on training set :  0.8766233766233766
Accuracy on test set :  0.8791275371099667
Recall on training set :  0.4080370942812983
Recall on test set :  0.4390681003584229
Precision on training set :  0.7415730337078652
Precision on test set :  0.7401812688821753
F1 on training set :  0.5264207377866401
F1 on test set :  0.5511811023622047
```

Fig.26 Metrics score of LDA

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.89	0.97	0.93	6406
1.0	0.74	0.41	0.53	1294
accuracy			0.88	7700
macro avg	0.82	0.69	0.73	7700
weighted avg	0.87	0.88	0.86	7700

Fig.27 Classification report of training data

Confusion matrix of training set:

Fig.28 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.89	0.97	0.93	2743
1.0	0.74	0.44	0.55	558
accuracy			0.88	3301
macro avg	0.82	0.70	0.74	3301
weighted avg	0.87	0.88	0.87	3301

Fig.29 Classification report of testing data

Confusion matrix of testing data:

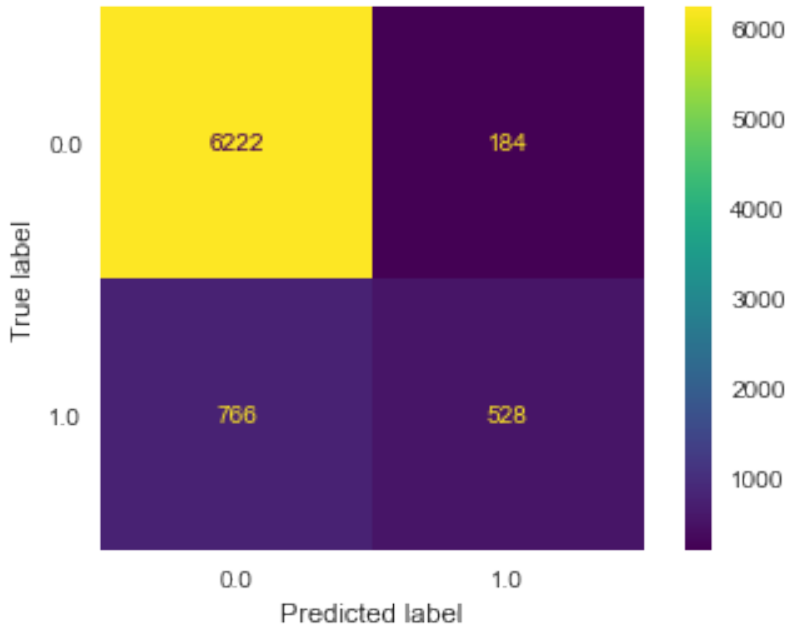


Fig.30 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.862

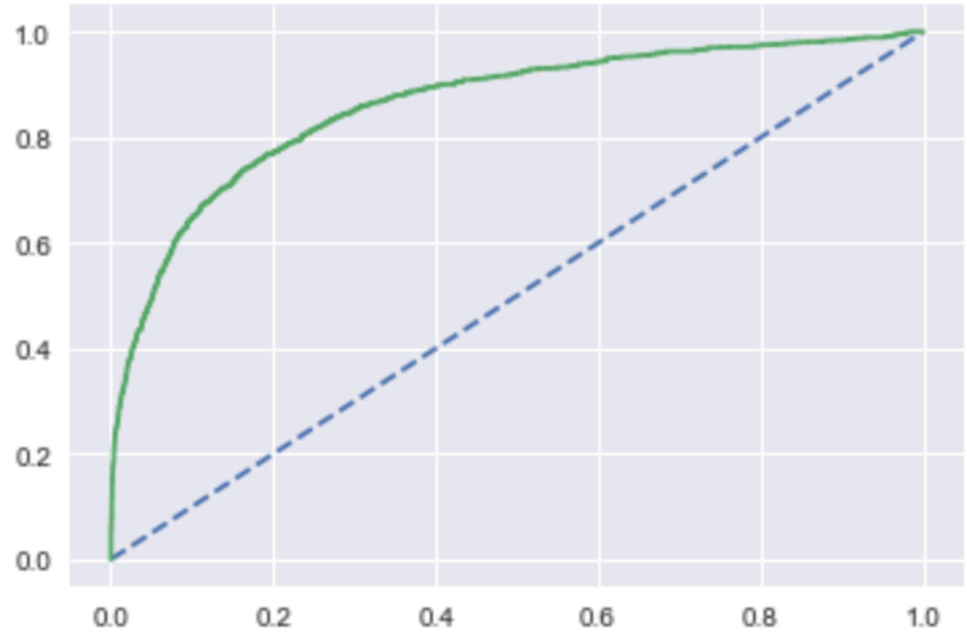


Fig.31 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.849
```

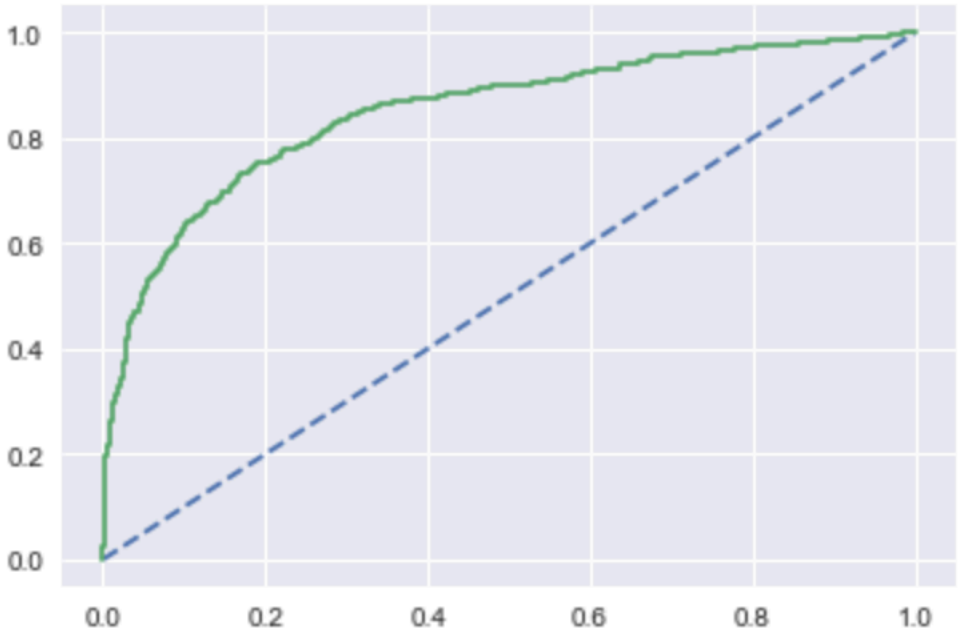


Fig.32 AUC and ROC curve of testing data

INFERENCE:

- Compared to the other models evaluated, the LDA model with default parameters has lower accuracy, recall, precision, and F1 score on the testing dataset.
- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is similar to the recall score, suggesting that the model produces a moderate number of false positives.
- The F1 score for the testing dataset is relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than many of the other models evaluated, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building ADA Boost model

A meta-estimator called an AdaBoost classifier starts by fitting a classifier to the initial dataset. It then fits additional copies of the classifier to the same dataset, but with the weights of instances that were incorrectly classified being changed so that later classifiers would concentrate more on challenging cases.

```
▼ AdaBoostClassifier
AdaBoostClassifier(random_state=1)
```

Fig.33 ADA Boost Classifier

Metrics score:

```
Accuracy on training set :  0.8958441558441559
Accuracy on test set :  0.8936685852771887
Recall on training set :  0.5795981452859351
Recall on test set :  0.5913978494623656
Precision on training set :  0.7440476190476191
Precision on test set :  0.7284768211920529
F1 on training set :  0.6516072980017377
F1 on test set :  0.6528189910979229
```

Fig.34 Metrics score of ADA Boost Classifier

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.92	0.96	0.94	6406
1.0	0.74	0.58	0.65	1294
accuracy			0.90	7700
macro avg	0.83	0.77	0.80	7700
weighted avg	0.89	0.90	0.89	7700

Fig.35 Classification report of training data

Confusion matrix of training set:

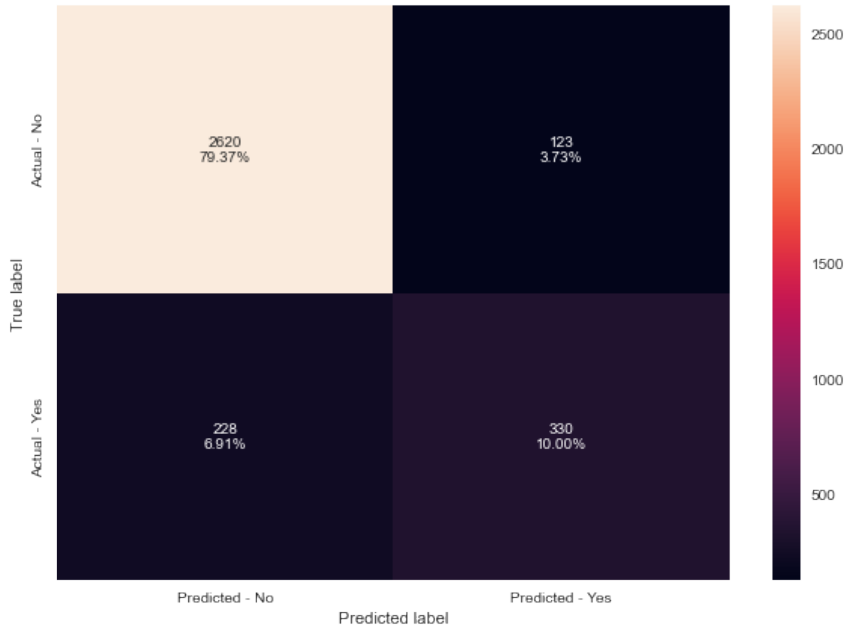


Fig36 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.92	0.96	0.94	2743
1.0	0.73	0.59	0.65	558
accuracy			0.89	3301
macro avg	0.82	0.77	0.80	3301
weighted avg	0.89	0.89	0.89	3301

Fig.37 Classification report of testing data

Confusion matrix of testing data:

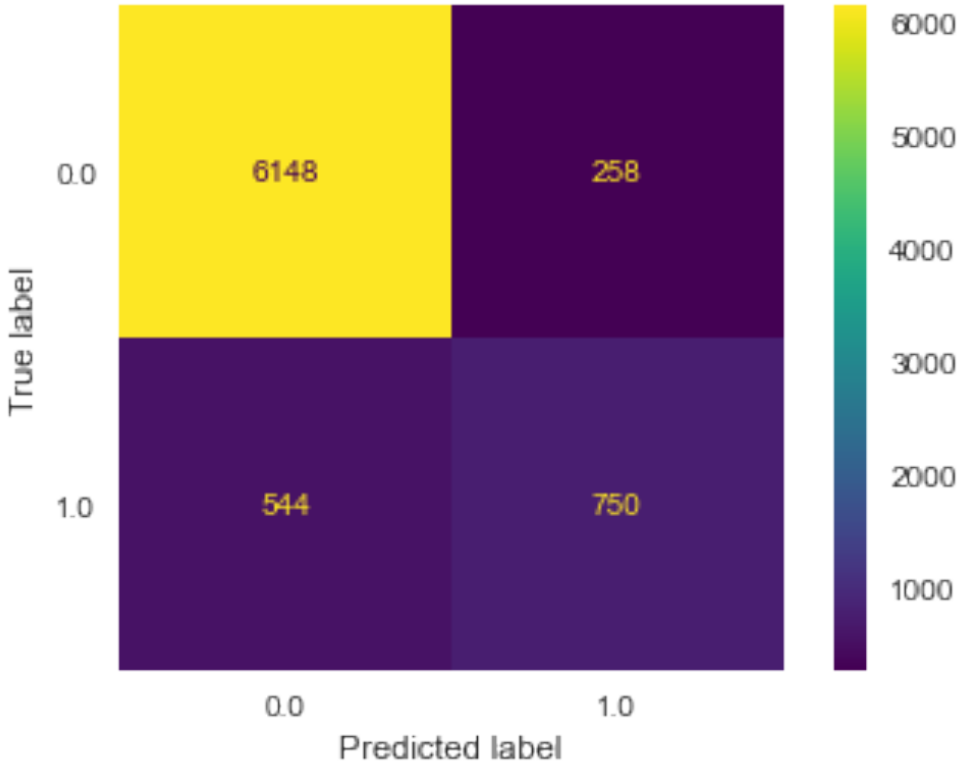


Fig.38 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.916

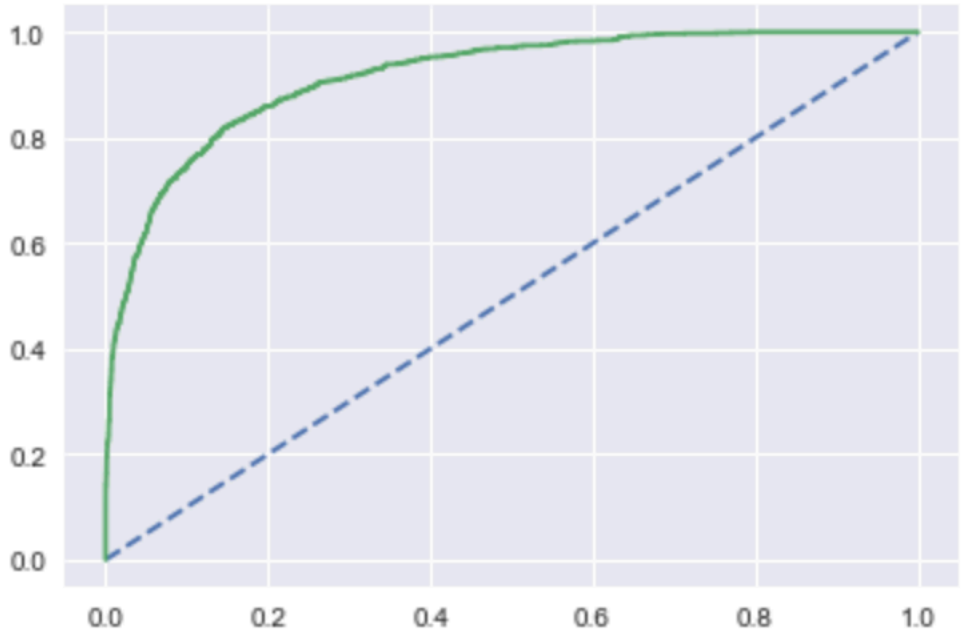


Fig.39 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.908

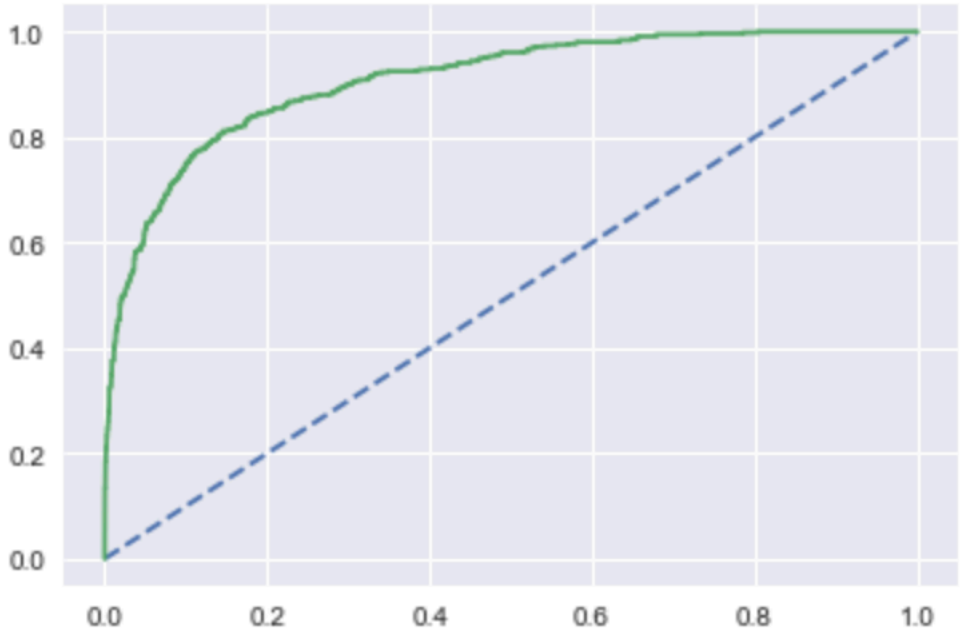


Fig.40 AUC and ROC curve of testing data

Feature Importance:

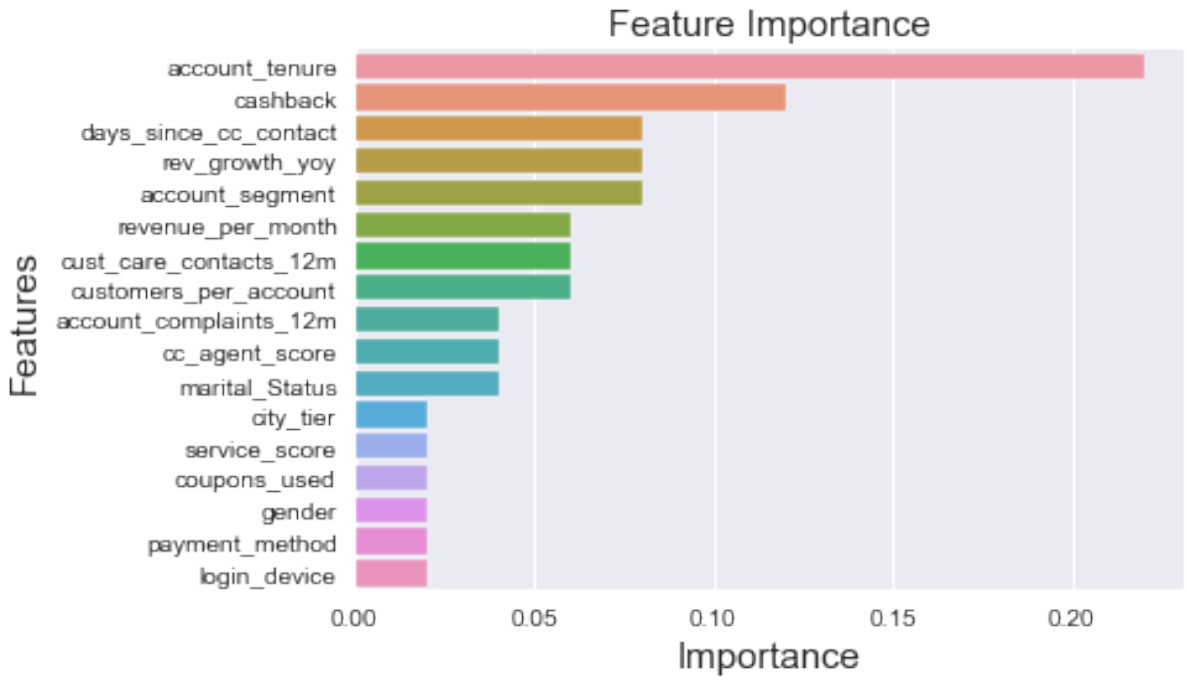


Fig.41 Feature importance obtained from ADA Boost model

INFERENCE:

- Compared to the other models evaluated, the AdaBoost with default parameters model has lower accuracy, recall, precision, and F1 score on the testing dataset.

- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than all of the other models evaluated, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building Gradient Boosting Classifier

A machine learning ensemble approach called gradient boost trains underlying models in a progressive, cumulative, and sequential fashion.

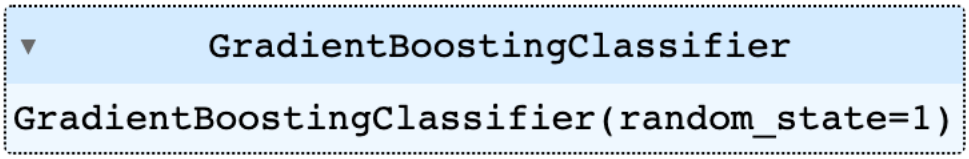


Fig.42 Parameters used in Gradient Boosting Classifier

Metrics score:

```
Accuracy on training set :  0.9176623376623376
Accuracy on test set :  0.9030596788851863
Recall on training set :  0.6236476043276662
Recall on test set :  0.5806451612903226
Precision on training set :  0.8459119496855346
Precision on test set :  0.7902439024390244
F1 on training set :  0.7179715302491102
F1 on test set :  0.6694214876033058
```

Fig.43 Metrics score of Gradient Boosting Classifier

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.93	0.98	0.95	6406
1.0	0.85	0.62	0.72	1294
accuracy			0.92	7700
macro avg	0.89	0.80	0.83	7700
weighted avg	0.91	0.92	0.91	7700

Fig.44 Classification report of training data

Confusion matrix of training set:

Fig.45 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.92	0.97	0.94	2743
1.0	0.79	0.58	0.67	558
accuracy			0.90	3301
macro avg	0.85	0.77	0.81	3301
weighted avg	0.90	0.90	0.90	3301

Fig.46 Classification report of testing data

Confusion matrix of testing data:

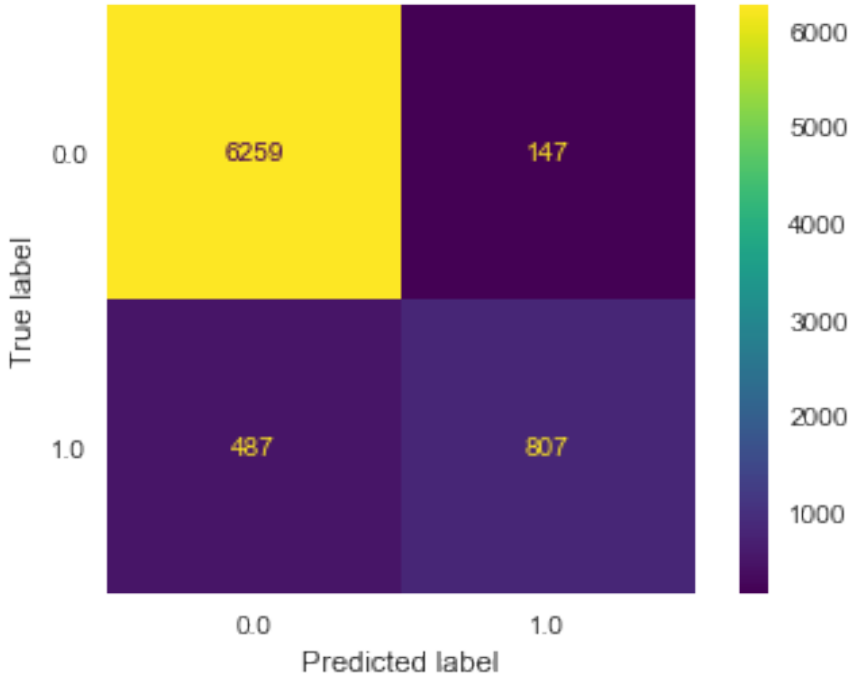


Fig.47 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 0.952
```

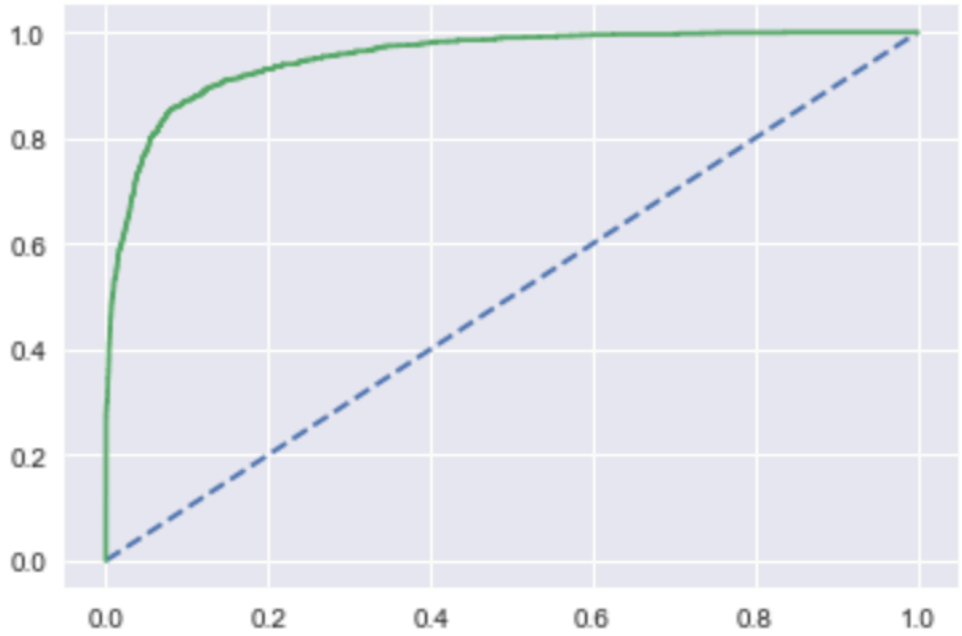


Fig.48 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.935
```

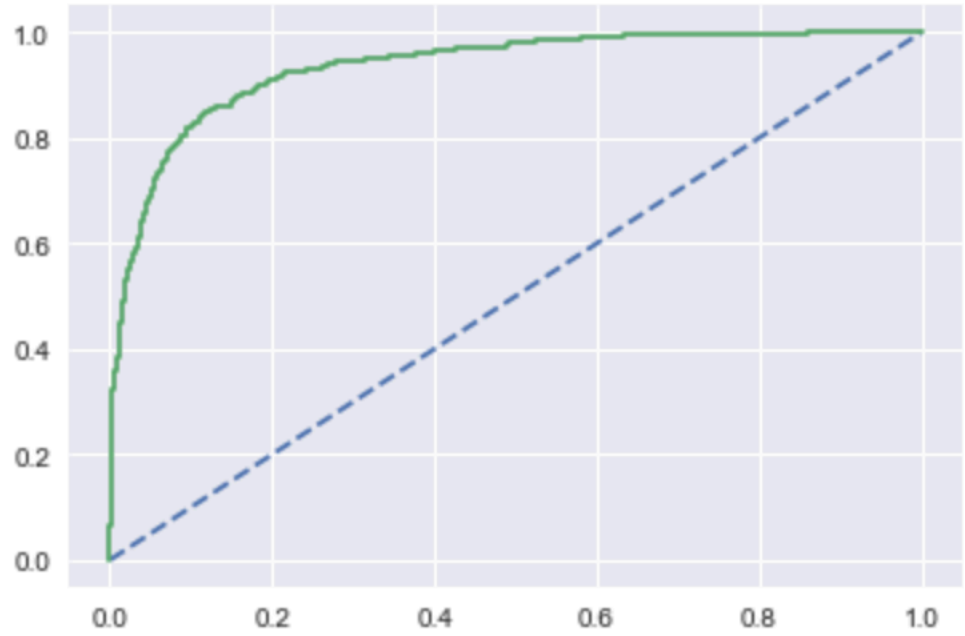


Fig.49 AUC and ROC curve of testing data

Feature Importance:

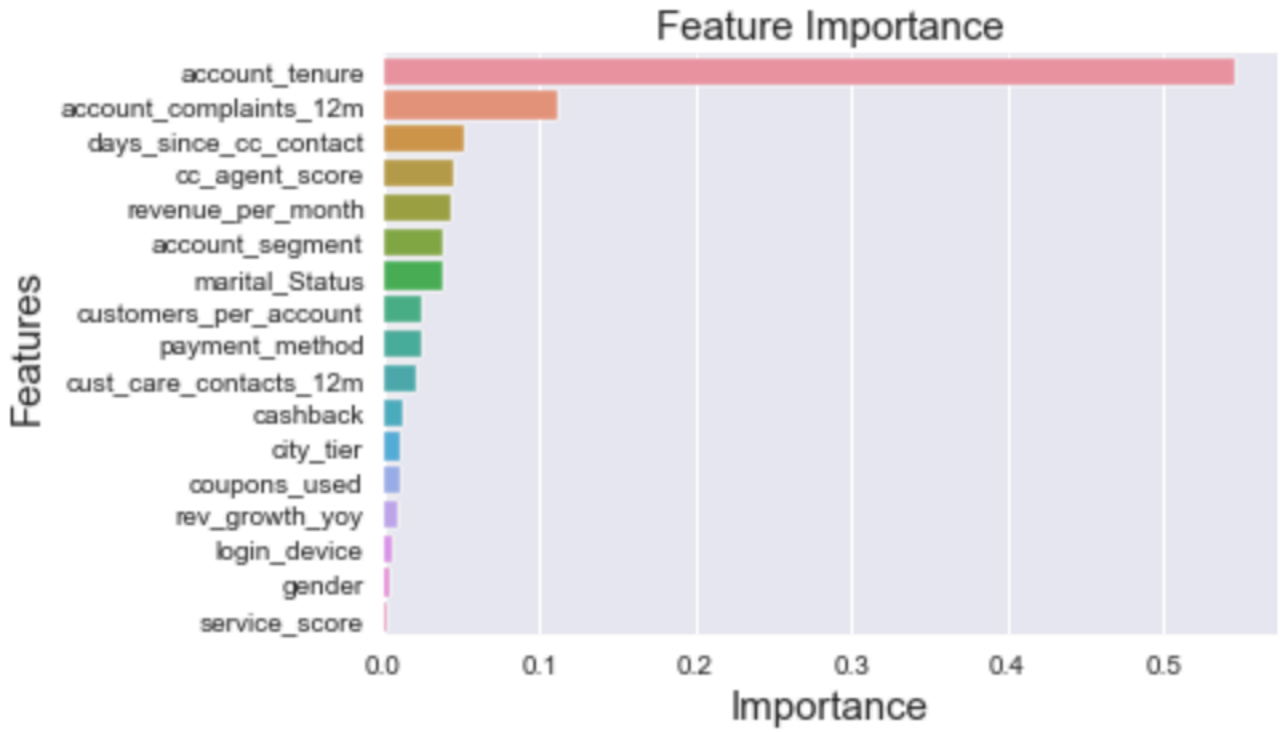


Fig.50 Feature importance obtained from Gradient Boosting model

INFERENCE:

- Compared to the other models evaluated, the Gradient Boosting with default parameters model has lower accuracy, recall, precision, and F1 score on the testing dataset.
- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than all of the other models evaluated, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building XGBoost Classifier

```
▼                                    XGBClassifier
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=100,
              n_jobs=0, num_parallel_tree=1, predictor='auto', random_state=1,
              reg_alpha=0, reg_lambda=1, ...)
```

Fig.51 Parameters used in XGBoost Classifier

Metrics score:

```
Accuracy on training set :  0.9998701298701299
Accuracy on test set :  0.9675855801272342
Recall on training set :  0.999227202472952
Recall on test set :  0.8853046594982079
Precision on training set :  1.0
Precision on test set :  0.9199255121042831
F1 on training set :  0.9996134518747584
F1 on test set :  0.902283105022831
```

Fig.52 Metrics score of XGBoost Classifier

Classification report of training set:

```
Classification Report of Training Data
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      6406
         1.0       1.00      1.00      1.00      1294

    accuracy                           1.00      7700
   macro avg       1.00      1.00      1.00      7700
weighted avg       1.00      1.00      1.00      7700
```

Fig.53 Classification report of training data

Confusion matrix of training set:

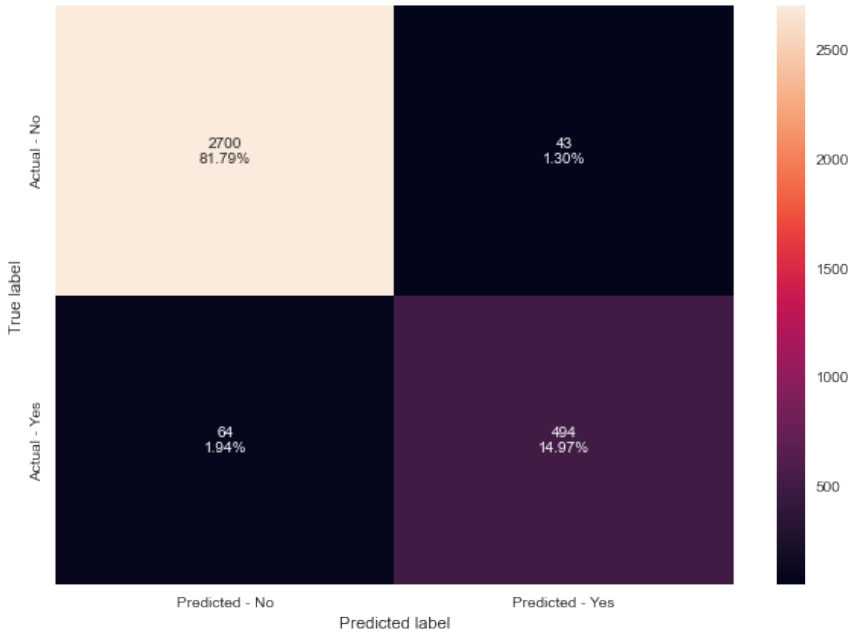


Fig.54 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.98	0.98	0.98	2743
1.0	0.92	0.89	0.90	558
accuracy			0.97	3301
macro avg	0.95	0.93	0.94	3301
weighted avg	0.97	0.97	0.97	3301

Fig.55 Classification report of testing data

Confusion matrix of testing data:

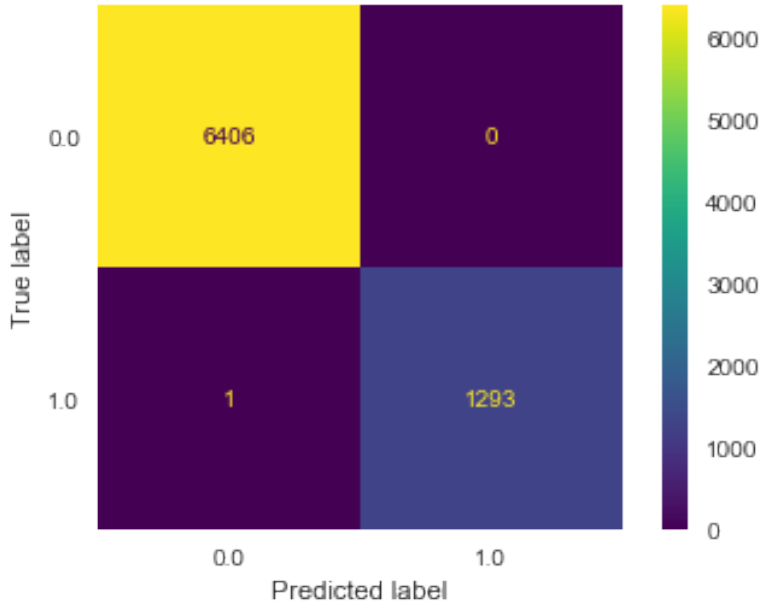


Fig.56 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 1.000

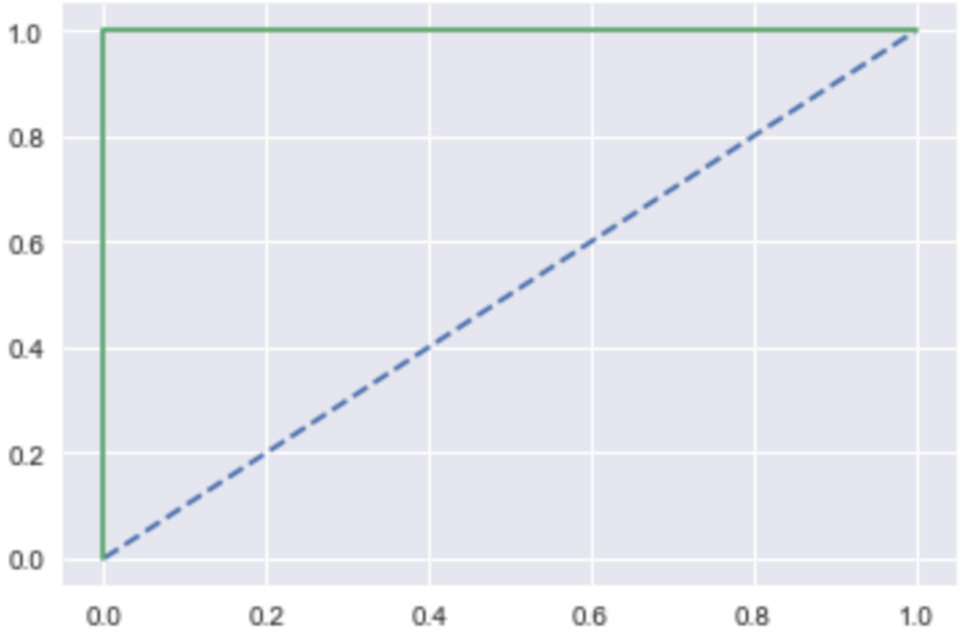


Fig.57 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.991

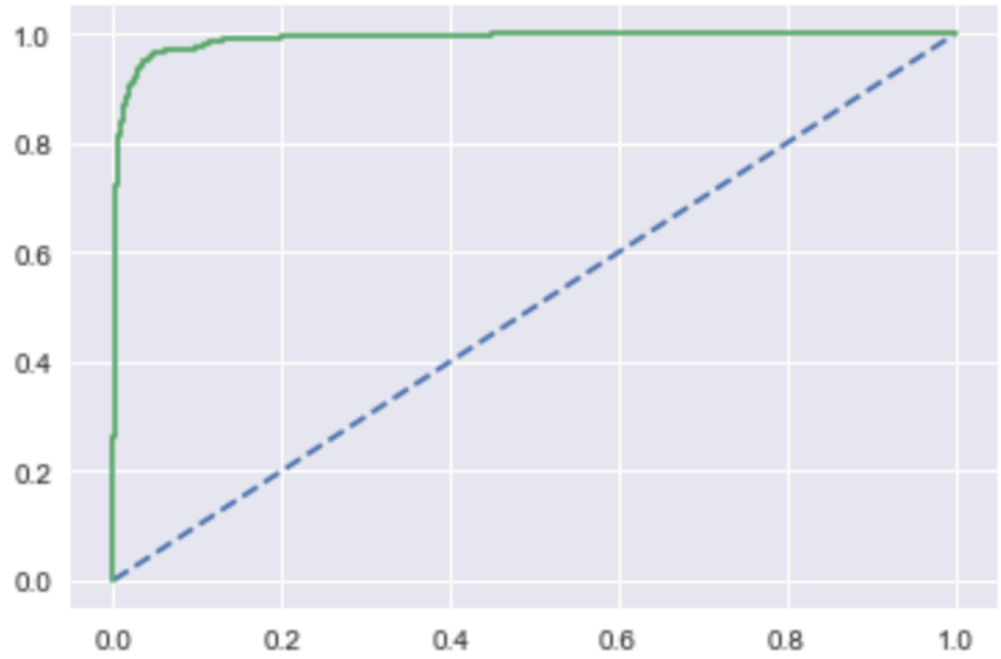


Fig.58 AUC and ROC curve of testing data

Feature Importance:

Fig.59 Feature importance obtained from XGBoost model

INFERENCE:

- Similar to the previous models, the XGBoost model has achieved near-perfect accuracy on the training dataset (1.00), which suggests that it has likely overfit the training data.
- However, it still has relatively high accuracy on the testing dataset (0.97), which suggests that it is able to generalize well to unseen data.
- The recall score for the testing dataset is lower than the training dataset, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives. The F1 score for the testing dataset is lower than the training dataset, indicating a trade-off between precision and recall.
- Overall, this model performs similarly to the Bagging with Decision Tree model, with slightly lower recall and F1 score on the testing dataset, but higher precision on the testing dataset than both the Gradient Boosting Tuned and XGBoost-Tuned models.

Building tuned XGBoost Classifier

```
▾                                    XGBClassifier
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=0.8,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.1, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=5, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=1000,
              n_jobs=4, nthread=4, num_parallel_tree=1, predictor='auto',
              random_state=27, reg_alpha=0, ...)
```

Fig.60 Parameters used in tuned XGBoost Classifier

Metrics score:

```
Accuracy on training set :  1.0
Accuracy on test set :  0.9700090881551046
Recall on training set :  1.0
Recall on test set :  0.9050179211469535
Precision on training set :  1.0
Precision on test set :  0.9165154264972777
F1 on training set :  1.0
F1 on test set :  0.9107303877366998
```

Fig.61 Metrics score of tuned XGBoost Classifier

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	6406
1.0	1.00	1.00	1.00	1294
accuracy			1.00	7700
macro avg	1.00	1.00	1.00	7700
weighted avg	1.00	1.00	1.00	7700

Fig.62 Classification report of training data

Confusion matrix of training set:

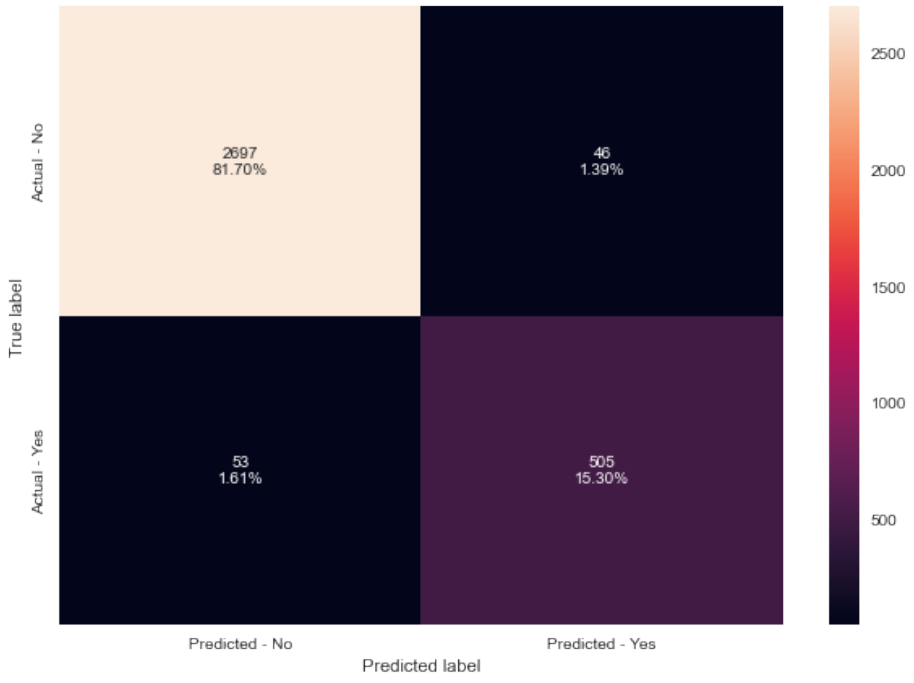


Fig.63 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.98	0.98	0.98	2743
1.0	0.92	0.91	0.91	558
accuracy			0.97	3301
macro avg	0.95	0.94	0.95	3301
weighted avg	0.97	0.97	0.97	3301

Fig.64 Classification report of testing data

Confusion matrix of testing data:

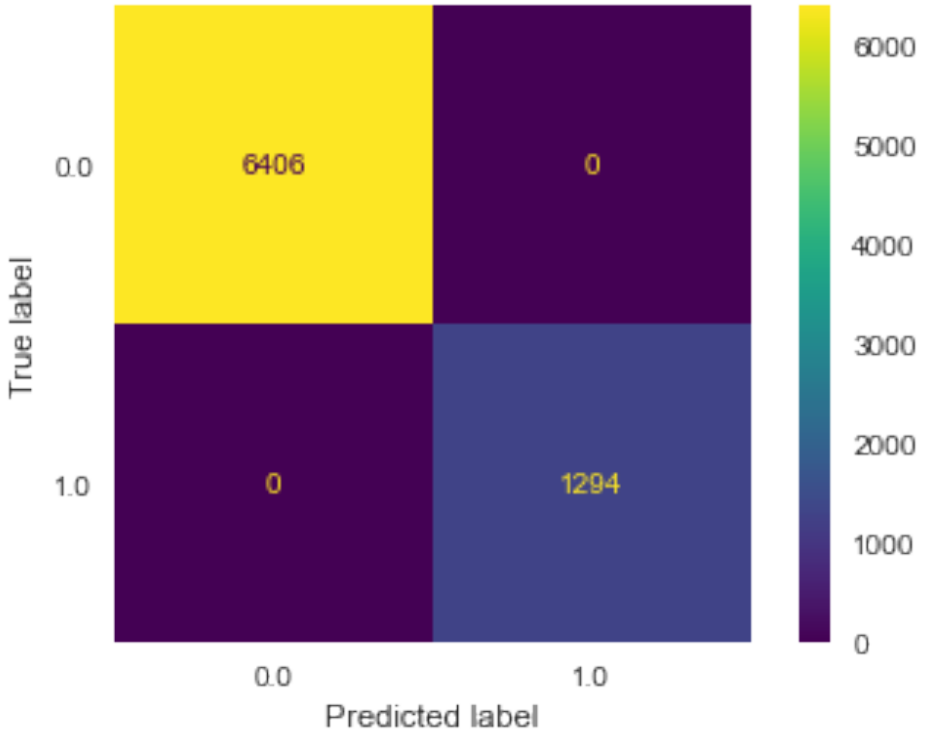


Fig.65 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 1.000
```

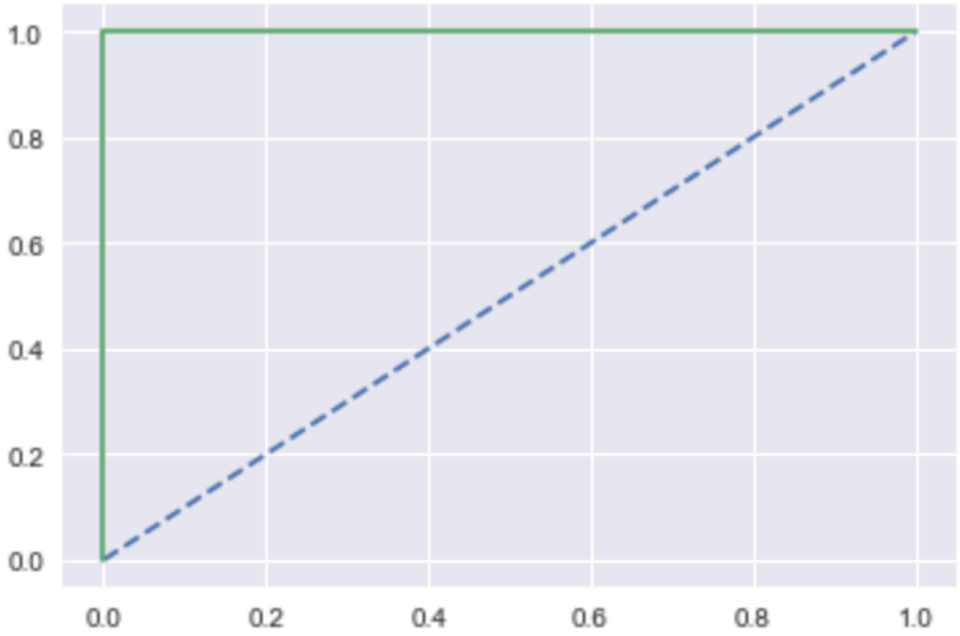


Fig.66 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.992
```

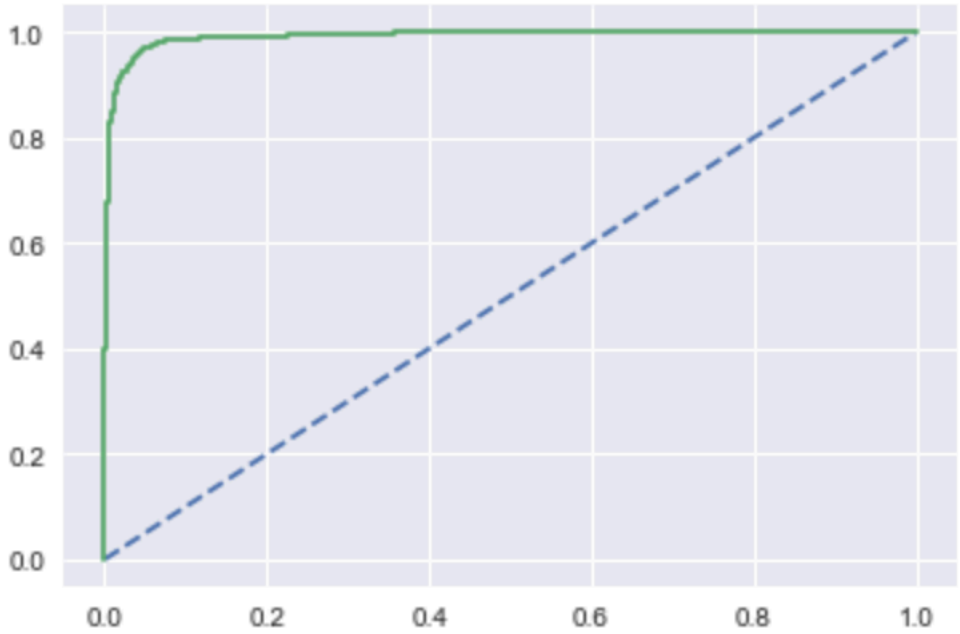


Fig.67 AUC and ROC curve of testing data

INFERENCE:

- Similar to the Gradient Boosting Tuned model, the XGBoost-Tuned model has achieved near-perfect accuracy on the training dataset (1.00), which suggests that it has likely overfit the training data.
- However, it still has relatively high accuracy on the testing dataset (0.97), which suggests that it is able to generalize well to unseen data.
- The recall score for the testing dataset is lower than the training dataset, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is lower than the training dataset, indicating a trade-off between precision and recall.
- Overall, this model also performs well, but not quite as well as the Gradient Boosting Tuned model, with slightly lower recall and F1 score on the testing dataset.

Building tuned ADA Boost model

A meta-estimator called an AdaBoost classifier starts by fitting a classifier to the initial dataset. It then fits additional copies of the classifier to the same dataset, but with the weights of instances that were incorrectly classified being changed so that later classifiers would concentrate more on challenging cases.

Among the crucial hyperparameters are:

The **base estimator** on which the boosted ensemble is based is called base estimator. The base estimator is by default a decision tree with max depth=1 and **n estimators** set to the highest number at which boosting is discontinued. 50 is the default value.

Learning rate: Learning rate reduces each classifier's overall contribution. Between learning rate and n estimators, there is a trade-off.

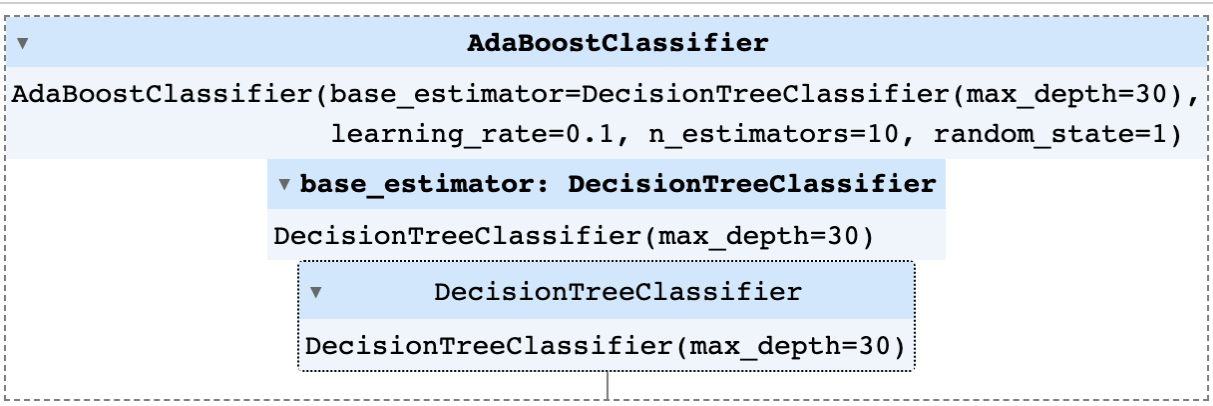


Fig.68 Hyper parameters used for tuned ADA Boost Classifier

Metrics score:

```
Accuracy on training set :  1.0
Accuracy on test set :  0.9418358073311118
Recall on training set :  1.0
Recall on test set :  0.8602150537634409
Precision on training set :  1.0
Precision on test set :  0.8080808080808081
F1 on training set :  1.0
F1 on test set :  0.8333333333333333
```

Fig.69 Metrics score of tuned ADA Boost Classifier

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	6406
1.0	1.00	1.00	1.00	1294
accuracy			1.00	7700
macro avg	1.00	1.00	1.00	7700
weighted avg	1.00	1.00	1.00	7700

Fig.70 Classification report of training data

Confusion matrix of training set:

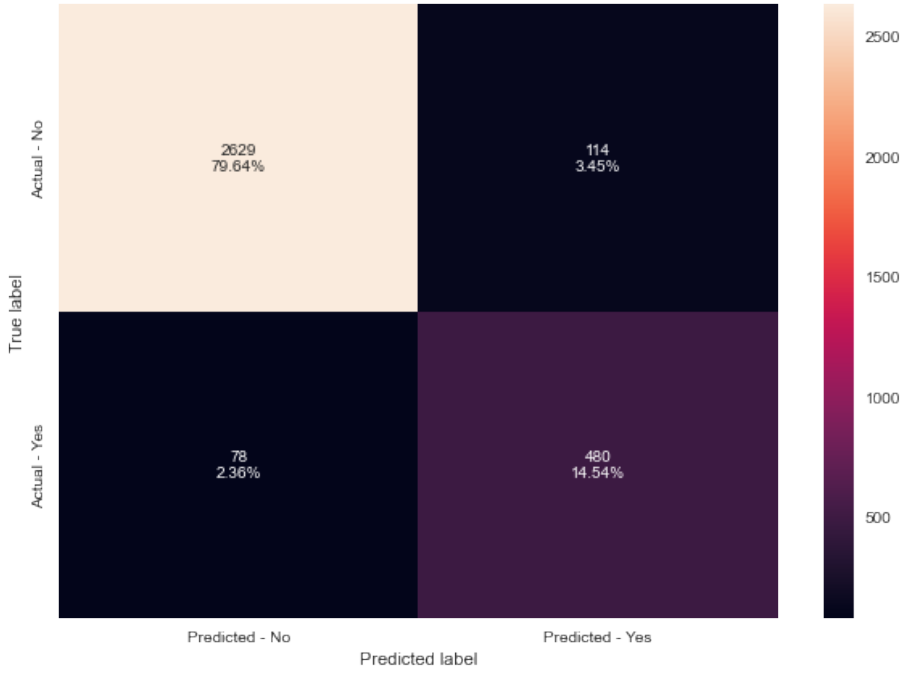


Fig.71 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.97	0.96	0.96	2743
1.0	0.81	0.86	0.83	558
accuracy			0.94	3301
macro avg	0.89	0.91	0.90	3301
weighted avg	0.94	0.94	0.94	3301

Fig.72 Classification report of testing data

Confusion matrix of testing data:

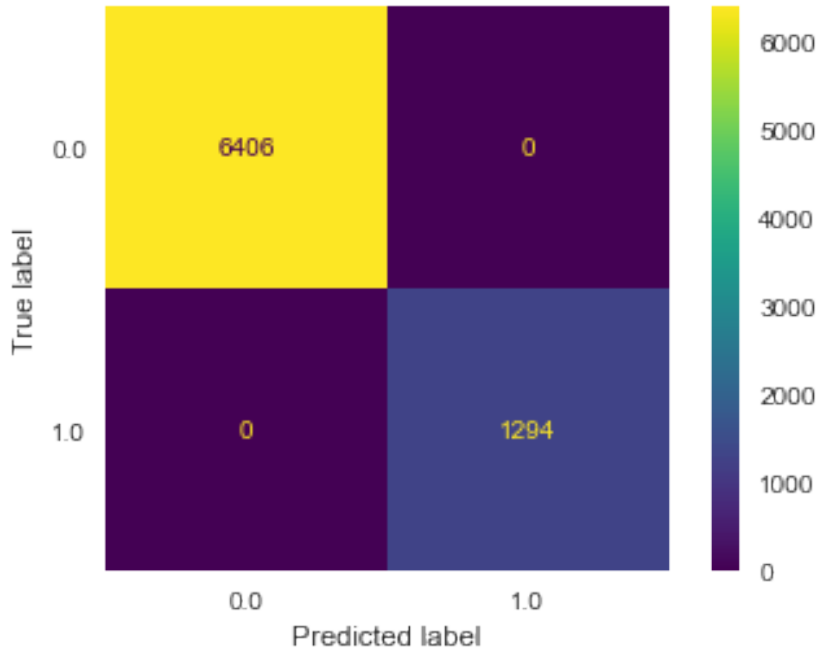


Fig.73 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 1.000
```

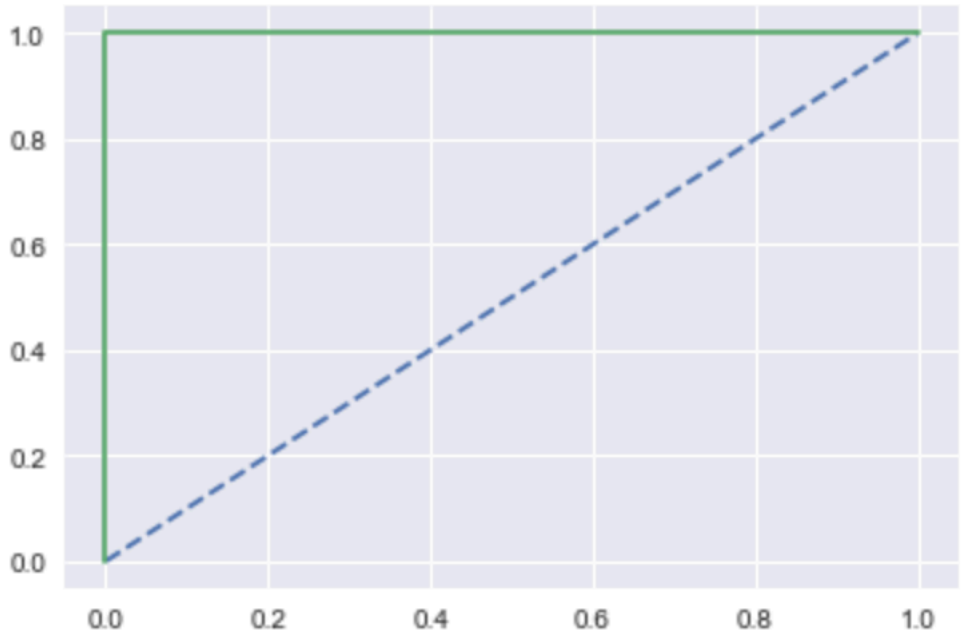


Fig.74 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.909
```

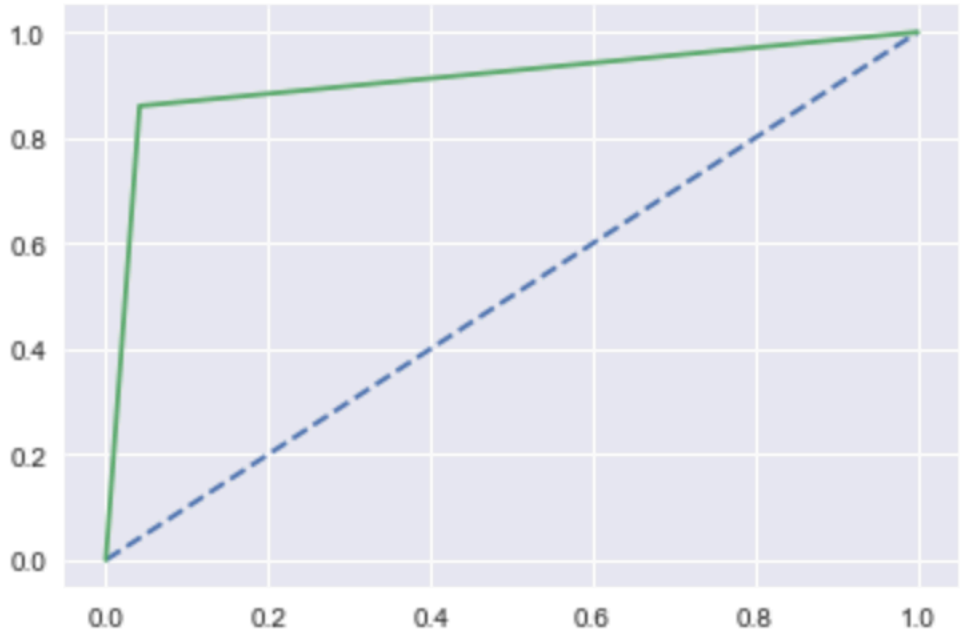


Fig.75 AUC and ROC curve of testing data

INFERENCE:

- Compared to the other models evaluated, the AdaBoost Tuned model has relatively high accuracy on the training dataset, but lower accuracy on the testing dataset, suggesting that it may be overfitting to the training data.

- The recall score for the testing dataset is also relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is lower than the recall score, suggesting that the model produces more false positives.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than the Gradient Boosting Tuned, XGBoost-Tuned, Random Forest, and Bagging with Decision Tree models, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building tuned Gradient Boosting model

The majority of the accessible hyperparameters are same to random forest classifier.

Initial predictions are calculated using an estimator object called init. The initial raw predictions are set to zero if the value is "zero". A DummyEstimator that forecasts the class priors is used by default.

Gradient boosting lacks a class weights argument.

```
▼                    GradientBoostingClassifier
GradientBoostingClassifier(criterion='mse', learning_rate=0.5, lo
ss='deviance',
                           max_depth=9, max_features=11, min_samp
les_leaf=6,
                           min_samples_split=15, n_estimators=20
1,
                           random_state=0)
```

Fig.76 Hyper parameters used for tuned Gradient Boost Classifier

Metrics score:

```
Accuracy on training set :  1.0
Accuracy on test set :  0.98242956679794
Recall on training set :  1.0
Recall on test set :  0.9247311827956989
Precision on training set :  1.0
Precision on test set :  0.9699248120300752
F1 on training set :  1.0
F1 on test set :  0.9467889908256881
```

Fig.77 Metrics score of tuned Gradient Boost Classifier

Classification report of training set:

```
Classification Report of Training Data
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      6406
         1.0       1.00      1.00      1.00      1294

    accuracy                           1.00      7700
   macro avg       1.00      1.00      1.00      7700
weighted avg       1.00      1.00      1.00      7700
```

Fig.78 Classification report of training data

Confusion matrix of training set:

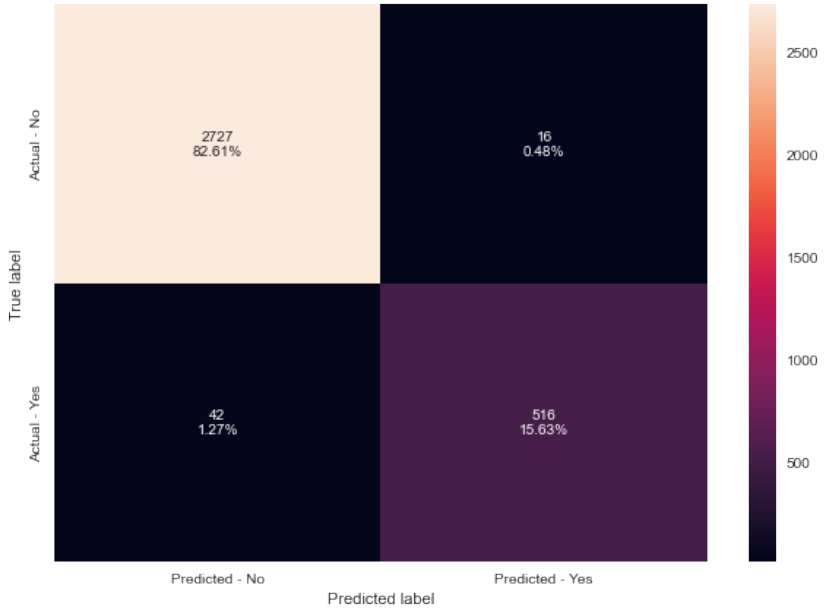


Fig.79 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.98	0.99	0.99	2743
1.0	0.97	0.92	0.95	558
accuracy			0.98	3301
macro avg	0.98	0.96	0.97	3301
weighted avg	0.98	0.98	0.98	3301

Fig.80 Classification report of testing data

Confusion matrix of testing data:

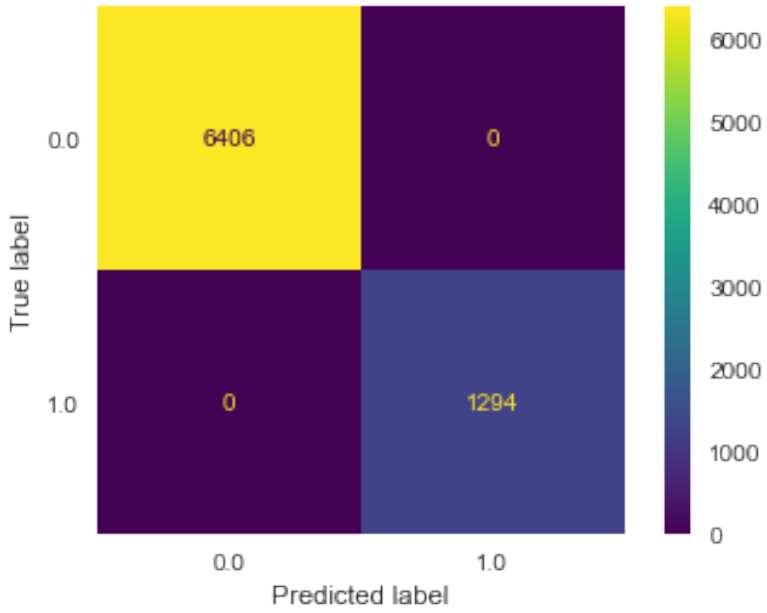


Fig.81 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 1.000
```

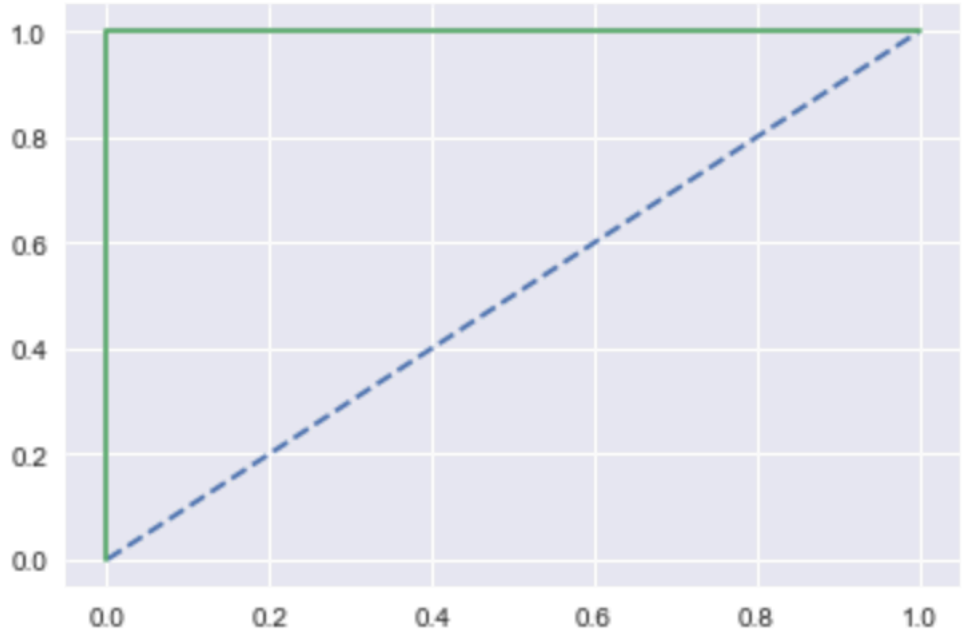


Fig.82 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.996
```

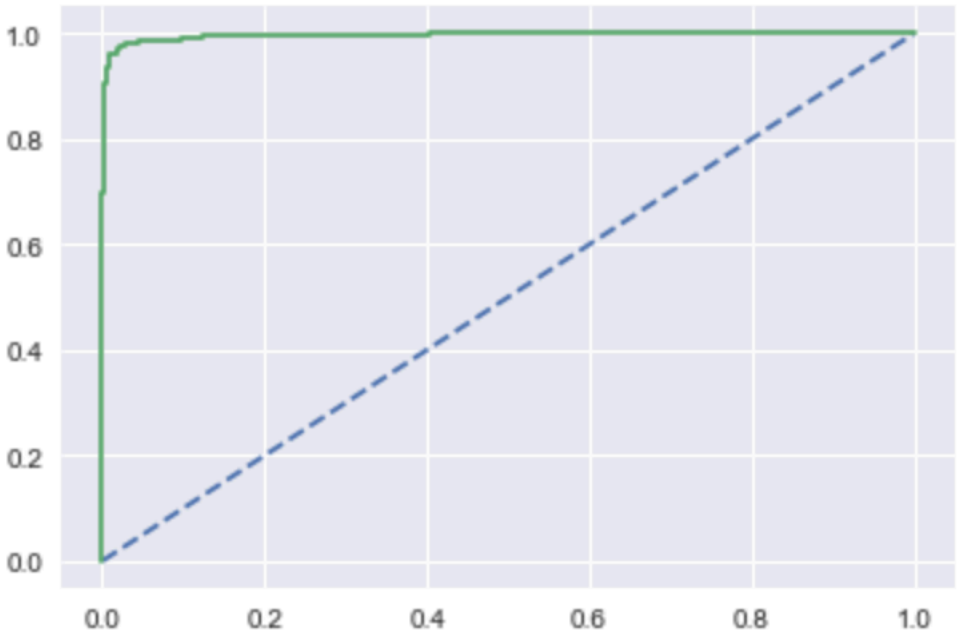


Fig.83 AUC and ROC curve of testing data

Feature Importance:

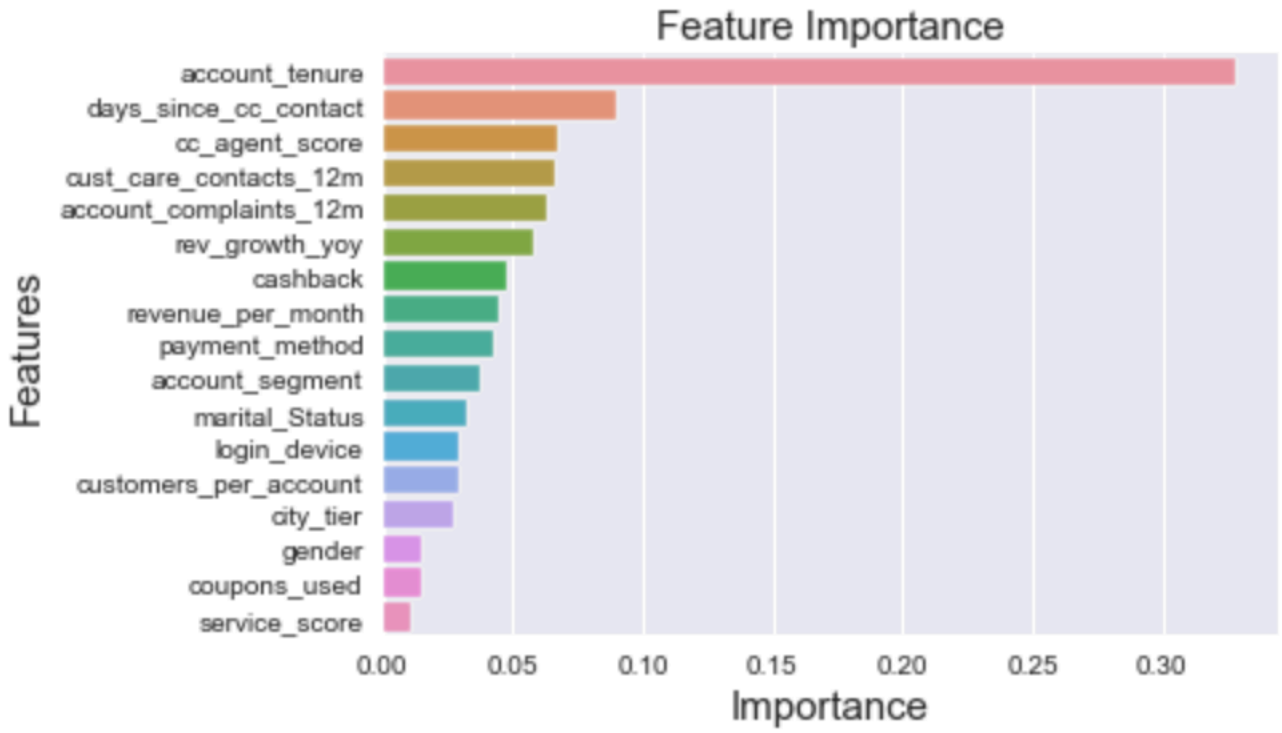


Fig.84 Feature importance of tuned Gradient boost model

INFERENCE:

- This model has achieved near-perfect accuracy on the training dataset (1.00), which suggests that it has likely overfit the training data.
- However, it still has very high accuracy on the testing dataset (0.98), which suggests that it is able to generalize well to unseen data.
- The recall score for both training and testing datasets is high, indicating that the model is able to identify most of the positive cases.
- The precision score for both training and testing datasets is also high, suggesting that the model does not produce many false positives.
- The F1 score for the testing dataset is slightly lower than the training dataset, but still relatively high, indicating a good balance between precision and recall.
- Overall, this model seems to perform very well, with good generalization to new data and a good balance between precision and recall.

Building Random Forest model:

A machine learning approach known as random forest employs bootstrapping to lessen the volatility in the underlying decision trees. Also, it only chooses a small subset of attributes for each split-decision node. Each tree is unique from the others since it is an ensemble of trees with various attributes for each node. Random forest does not need the data to be scaled and is resistant to outliers.

The Random Forest classifier function in Sklearn was used in this modelling assignment to model:

• The default hyperparameters were used to run the base model, and the performance data were recorded. Then, tuning was carried out with GridSearchCV.

• The sections below offer model performance metrics for the basic model and the best model.

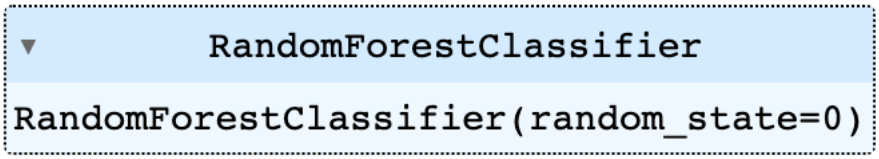


Fig.85 Parameters used for Random forest model

Metrics score:

```
Accuracy on training set :  1.0
Accuracy on test set :  0.9697061496516207
Recall on training set :  1.0
Recall on test set :  0.8637992831541219
Precision on training set :  1.0
Precision on test set :  0.9525691699604744
F1 on training set :  1.0
F1 on test set :  0.9060150375939849
```

Fig.86 Metrics score of Random forest model

Classification report of training set:

```
Classification Report of Training Data
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      6406
         1.0       1.00      1.00      1.00      1294

    accuracy                           1.00      7700
   macro avg       1.00      1.00      1.00      7700
weighted avg       1.00      1.00      1.00      7700
```

Fig.87 Classification report of training data

Confusion matrix of training set:

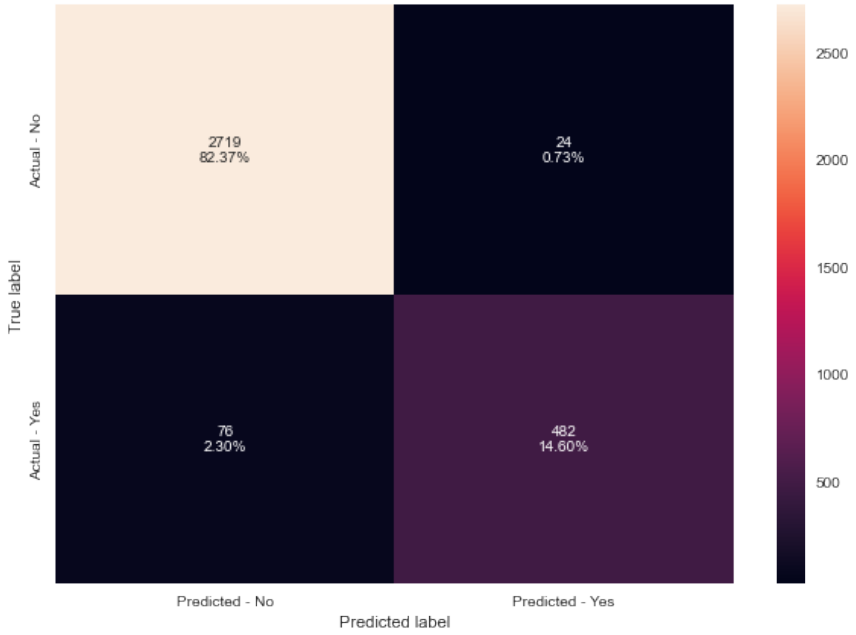


Fig.88 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data				
	precision	recall	f1-score	support
0.0	0.97	0.99	0.98	2743
1.0	0.95	0.86	0.91	558
accuracy			0.97	3301
macro avg	0.96	0.93	0.94	3301
weighted avg	0.97	0.97	0.97	3301

Fig.89 Classification report of testing data

Confusion matrix of testing data:

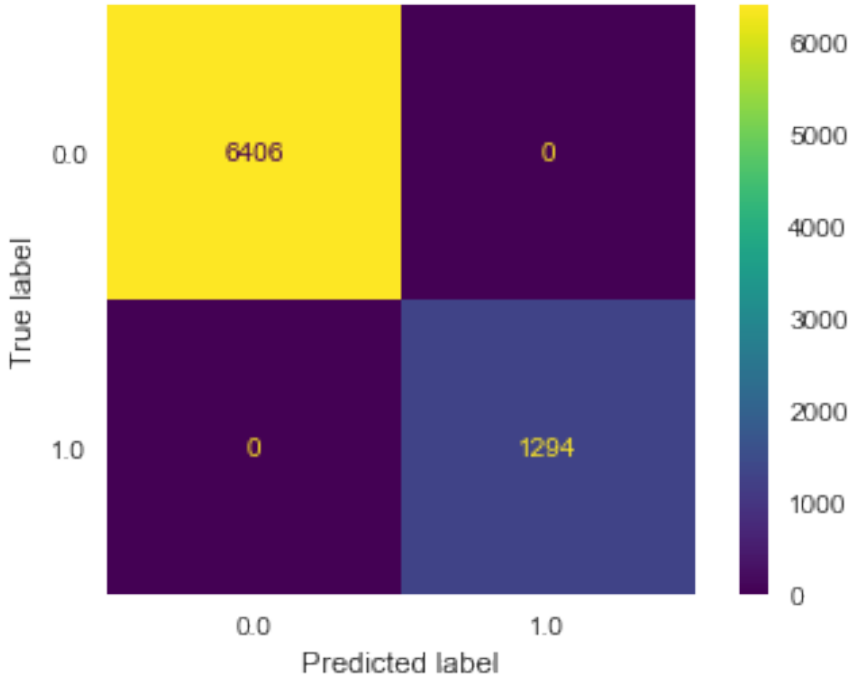


Fig.90 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 1.000
```

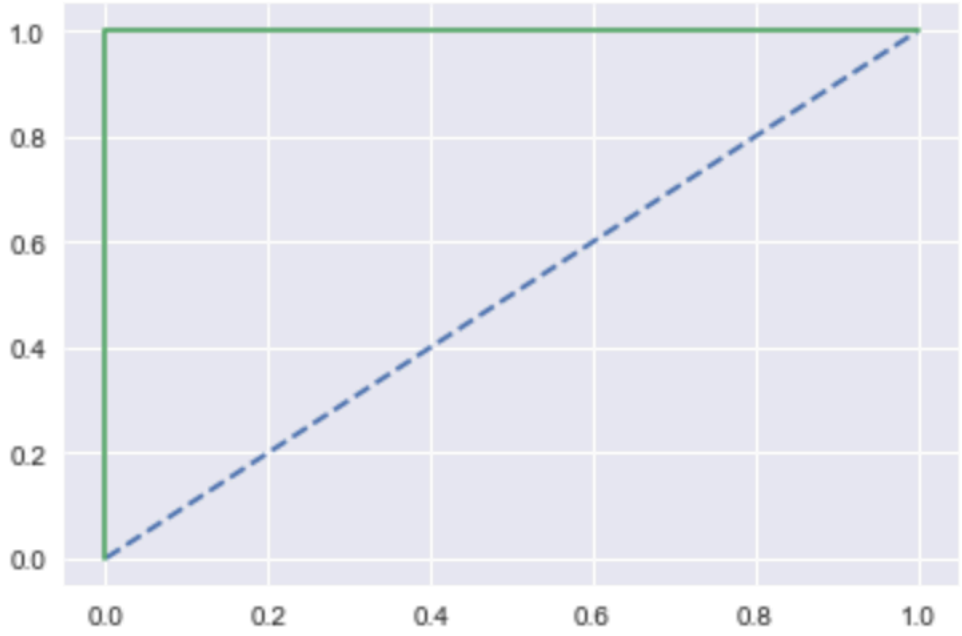


Fig.91 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.993

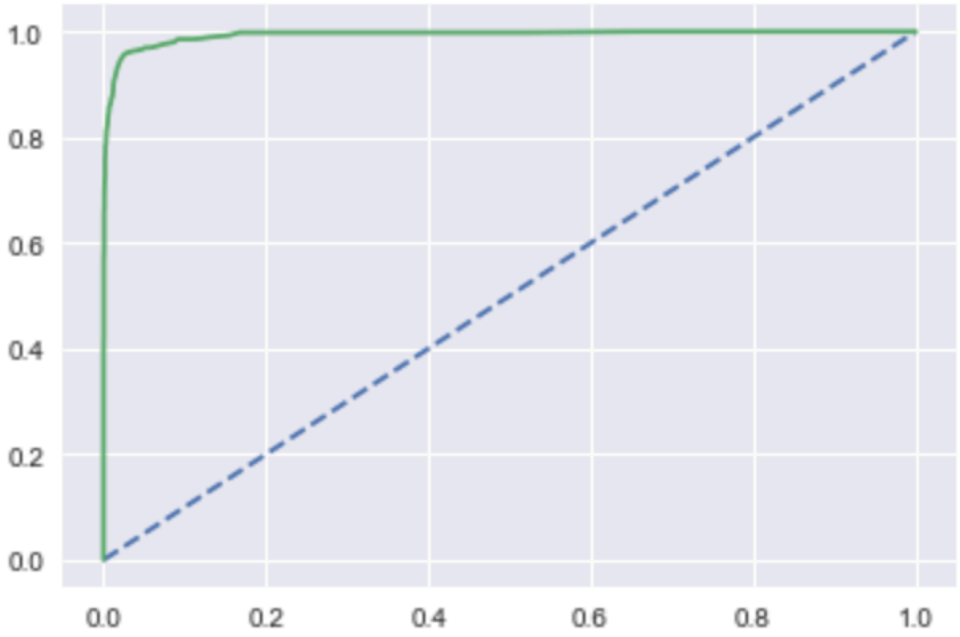


Fig.92 AUC and ROC curve of testing data

Feature Importance:

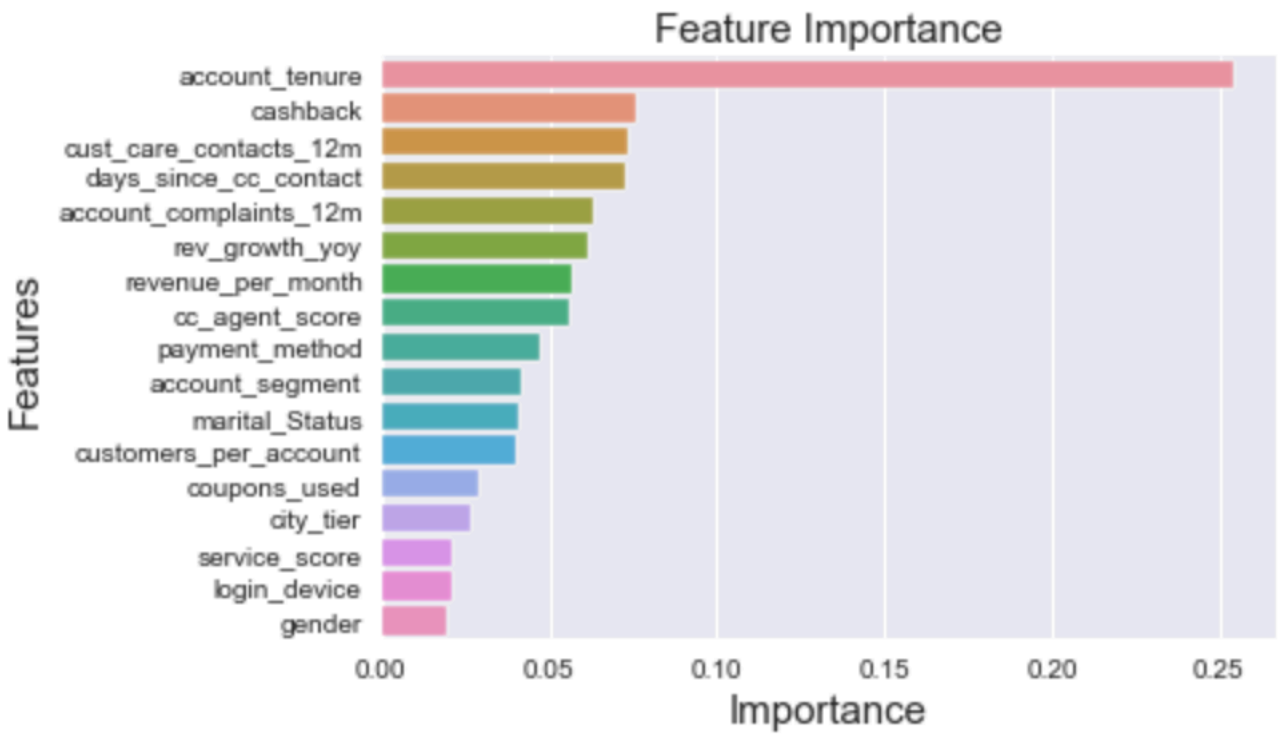


Fig.93 Feature importance of tuned Gradient boost model

INFERENCE:

- Similar to the previous models, the Random Forest model has achieved near-perfect accuracy on the training dataset (1.00), which suggests that it has likely overfit the training data.
- However, it still has relatively high accuracy on the testing dataset (0.97), which suggests that it is able to generalize well to unseen data.
- The recall score for the testing dataset is lower than the training dataset, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is lower than the training dataset, indicating a trade-off between precision and recall.
- Overall, this model also performs well, but not quite as well as the Gradient Boosting Tuned model, with lower recall and F1 score on the testing dataset. However, it has higher precision on the testing dataset than both the Gradient Boosting Tuned and XGBoost-Tuned models.

Building tuned Random Forest model:

The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV.

• As can be seen from the base model, the model had overfit on train dataset (all 1s). During tuning, the tree depth was contained, the minimum samples in each leaf increased to reduce overfit.

```
Best parameters for Random Forest :{'criterion': 'gini', 'max_dept
h': 4, 'max_features': 'sqrt', 'min_samples_split': 9, 'n_estimato
rs': 200}
```

Fig.94 Parameters for tuned Random Forest model

```
▼ RandomForestClassifier
RandomForestClassifier(max_depth=9, n_estimators=300, random_stat
e=1)
```

Fig.95 Best Parameters for tuned Random Forest model

Metrics score:

```
Accuracy on training set :  0.9606493506493506
Accuracy on test set :  0.932747652226598
Recall on training set :  0.7782071097372488
Recall on test set :  0.6720430107526881
Precision on training set :  0.9843597262952102
Precision on test set :  0.9057971014492754
F1 on training set :  0.8692274492878723
F1 on test set :  0.771604938271605
```

Fig.96 Metrics score of tuned Random forest model

Classification report of training set:

```
Classification Report of Training Data
              precision    recall  f1-score   support

         0.0       0.96      1.00      0.98      6406
         1.0       0.98      0.78      0.87      1294

    accuracy                           0.96      7700
   macro avg       0.97      0.89      0.92      7700
weighted avg       0.96      0.96      0.96      7700
```

Fig.97 Classification report of training data

Confusion matrix of training set:

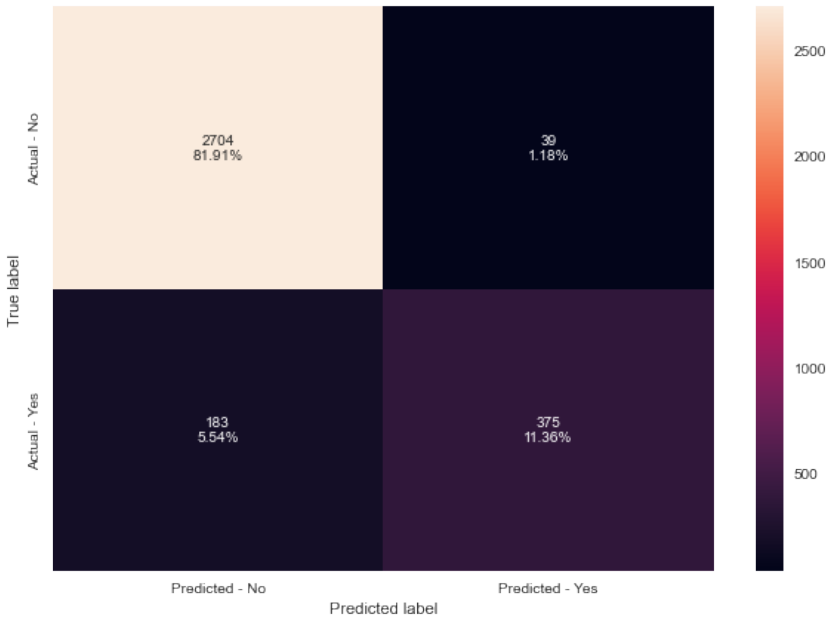


Fig.98 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.94	0.99	0.96	2743
1.0	0.91	0.67	0.77	558
accuracy			0.93	3301
macro avg	0.92	0.83	0.87	3301
weighted avg	0.93	0.93	0.93	3301

Fig.99 Classification report of testing data

Confusion matrix of testing data:

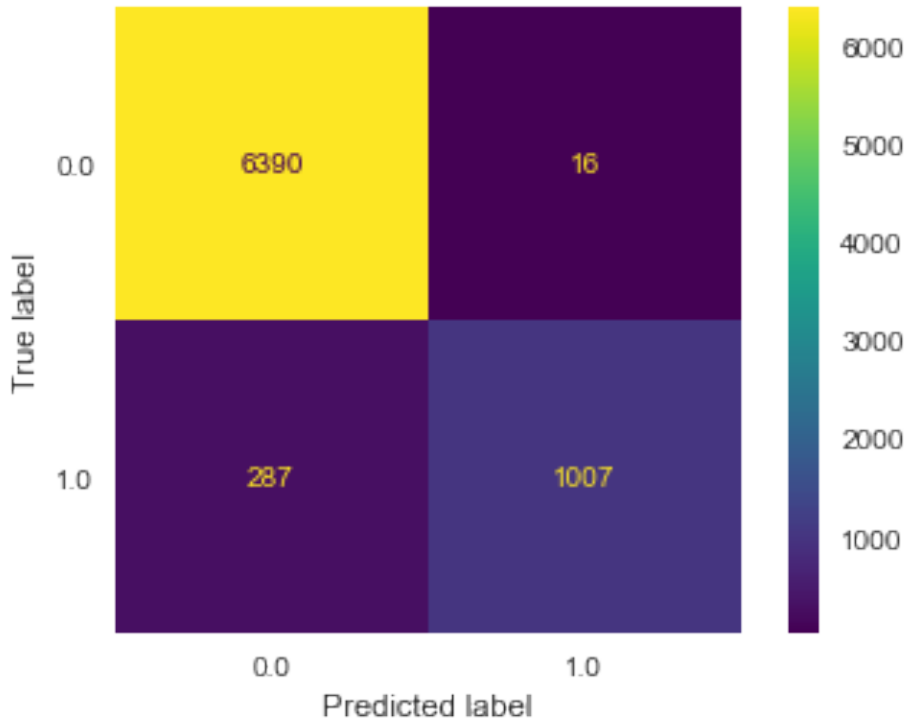


Fig.100 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.991

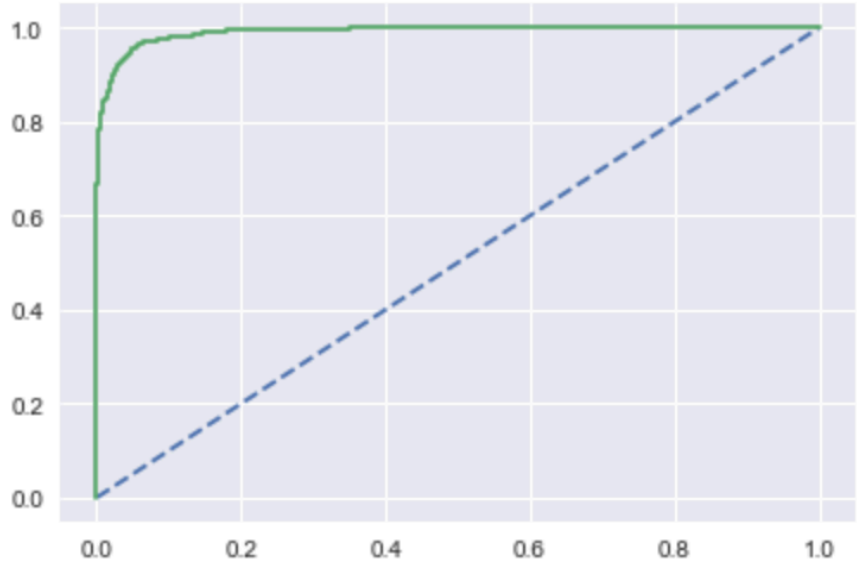


Fig.101 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.971

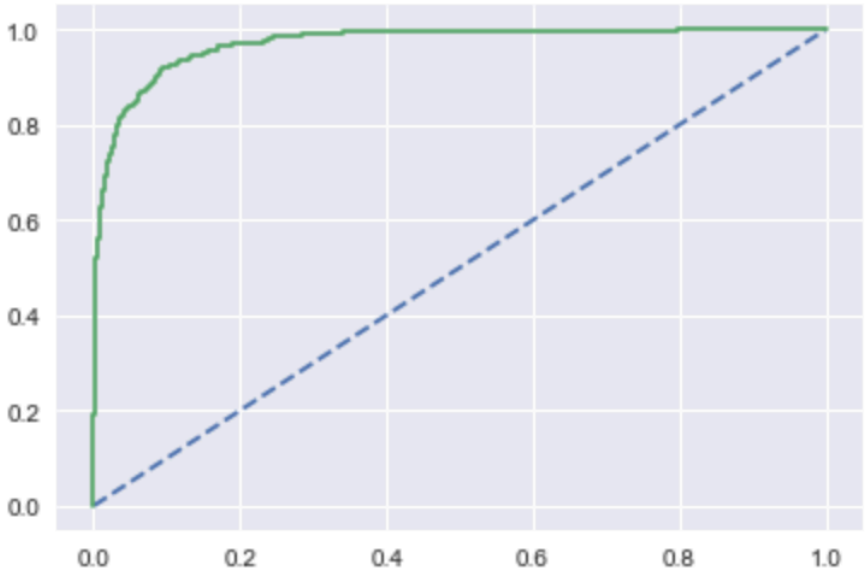


Fig.102 AUC and ROC curve of testing data

INFERENCE:

- Compared to the Random Forest model, the Random Forest - Tuned model has lower accuracy on both the training and testing datasets, but higher precision on the testing dataset, suggesting that the model produces fewer false positives.

- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than the Gradient Boosting Tuned, XGBoost-Tuned, and Bagging with Decision Tree models, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building ANN model:

An effective machine learning approach for classification and regression is the Artificial Neural Network (ANN). The intricate patterns in the underlying data may be learned by this method. Overfitting is a possibility, but it may be avoided by employing the hyperparameters during the tuning exercise.

The Multilayer Perceptron function of Sklearn was utilised in this modelling exercise to model:

Sklearn's Standard Scaler was used to scale the data because the model needs it.

• The baseline model was run using the default hyperparameters on a scaled dataset with outliers removed, and the performance metrics were recorded. Further tuning was carried out with GridSearchCV.

• The sections below offer model performance metrics for the basic model and the best model.

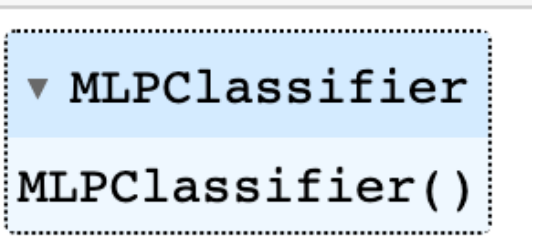


Fig.103 ANN using MLPClassifier

Metrics score:

```
Accuracy on training set :  0.9476623376623377
Accuracy on test set :  0.926991820660406
Recall on training set :  0.7774343122102009
Recall on test set :  0.7150537634408602
Precision on training set :  0.8974130240856378
Precision on test set :  0.8295218295218295
F1 on training set :  0.8331262939958592
F1 on test set :  0.768046198267565
```

Fig.104 Metrics score of ANN model

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.96	0.98	0.97	6406
1.0	0.90	0.78	0.83	1294
accuracy			0.95	7700
macro avg	0.93	0.88	0.90	7700
weighted avg	0.95	0.95	0.95	7700

Fig.105 Classification report of training data

Confusion matrix of training set:

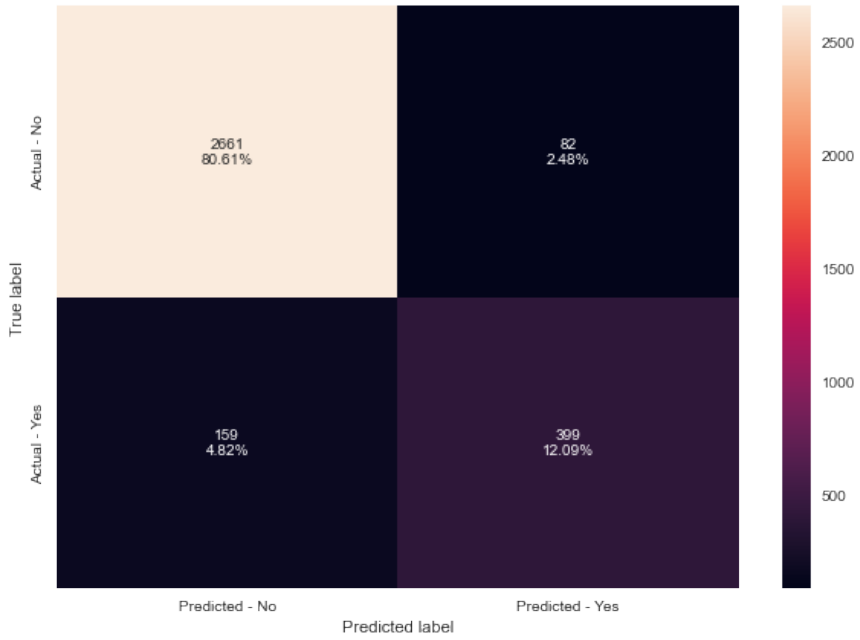


Fig.106 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.94	0.97	0.96	2743
1.0	0.83	0.72	0.77	558
accuracy			0.93	3301
macro avg	0.89	0.84	0.86	3301
weighted avg	0.92	0.93	0.92	3301

Fig.107 Classification report of testing data

Confusion matrix of testing data:

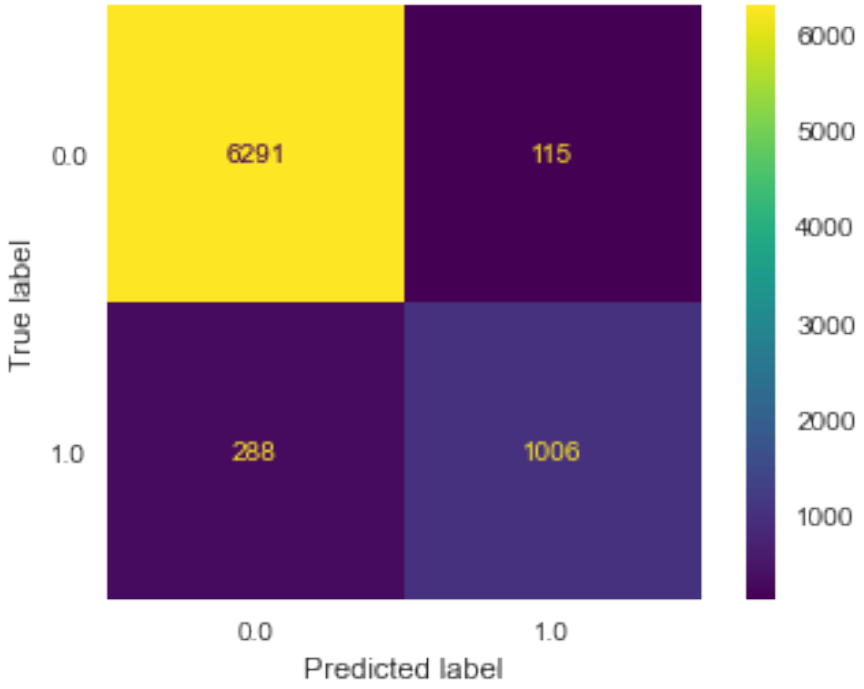


Fig.108 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.975

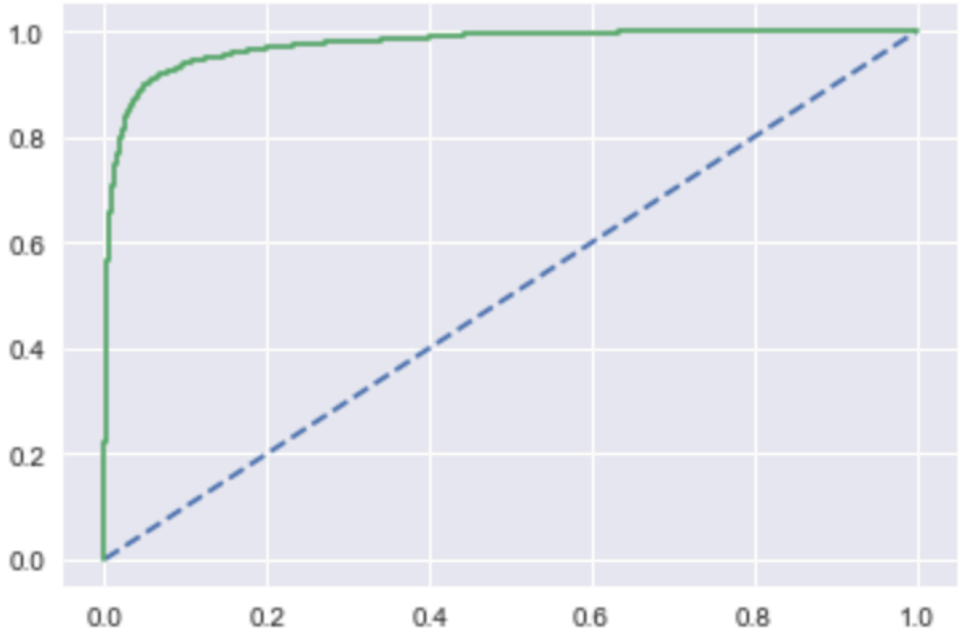


Fig.109 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.954

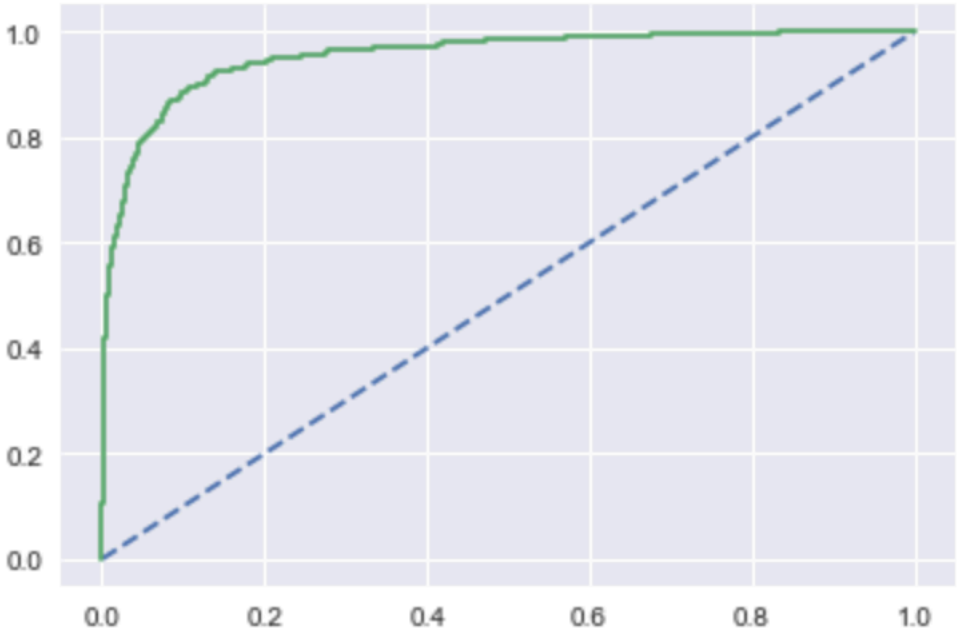


Fig.110 AUC and ROC curve of testing data

INFERENCE:

- Compared to the other models evaluated, the ANN model has lower accuracy on both the training and testing datasets.

- The recall score for the testing dataset is relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than the Gradient Boosting Tuned, XGBoost-Tuned, Random Forest, Bagging with Decision Tree, and KNN-Tuned models, with lower accuracy, recall, precision, and F1 score on the testing dataset.

Building tuned ANN model:

An effective machine learning approach for classification and regression is the Artificial Neural Network (ANN). The intricate patterns in the underlying data may be learned by this method. Overfitting is a possibility, but it may be avoided by employing the hyperparameters during the tuning exercise.

```
{'activation': 'tanh', 'alpha': 0.001, 'hidden_layer_sizes': 100, 'learning_rate': 'constant', 'solver': 'adam'}

MLPClassifier(activation='tanh', alpha=0.001, hidden_layer_sizes=100,
              max_iter=5000, random_state=1, verbose=True)
```

Fig.111 Best Parameters used for tuned ANN model

Metrics score:

```
Accuracy on training set :  0.9780392156862745
Accuracy on test set :  0.884859474161378
Recall on training set :  0.9636785162287481
Recall on test set :  0.8745519713261649
Precision on training set :  0.9928343949044586
Precision on test set :  0.8954128440366973
F1 on training set :  0.9780392156862745
F1 on test set :  0.884859474161378
```

Fig.112 Metrics score of tuned ANN model

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.99	1.00	1.00	6406
1.0	0.99	0.96	0.98	1294
accuracy			0.99	7700
macro avg	0.99	0.98	0.99	7700
weighted avg	0.99	0.99	0.99	7700

Fig.113 Classification report of training data

Confusion matrix of training set:

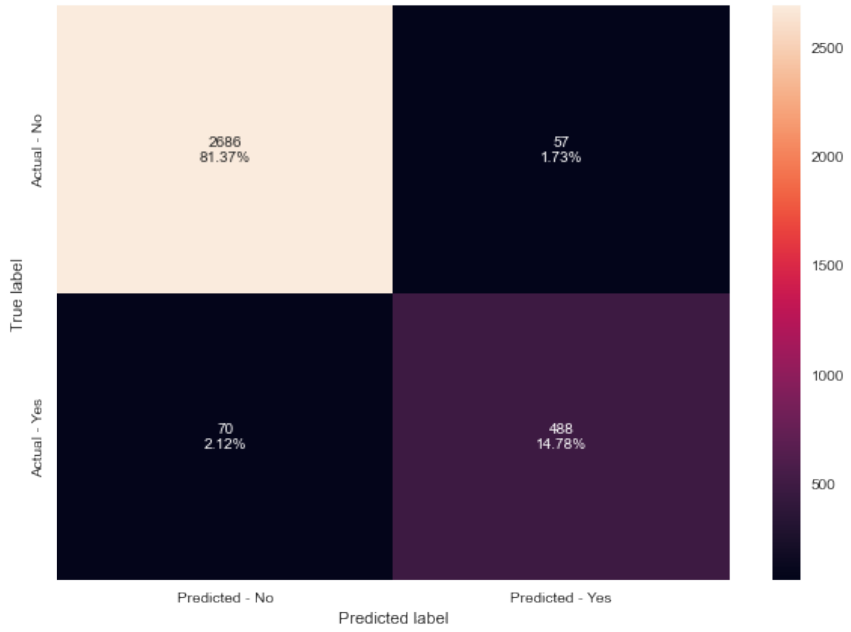


Fig.114 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.97	0.98	0.98	2743
1.0	0.90	0.87	0.88	558
accuracy			0.96	3301
macro avg	0.94	0.93	0.93	3301
weighted avg	0.96	0.96	0.96	3301

Fig.115 Classification report of testing data

Confusion matrix of testing data:

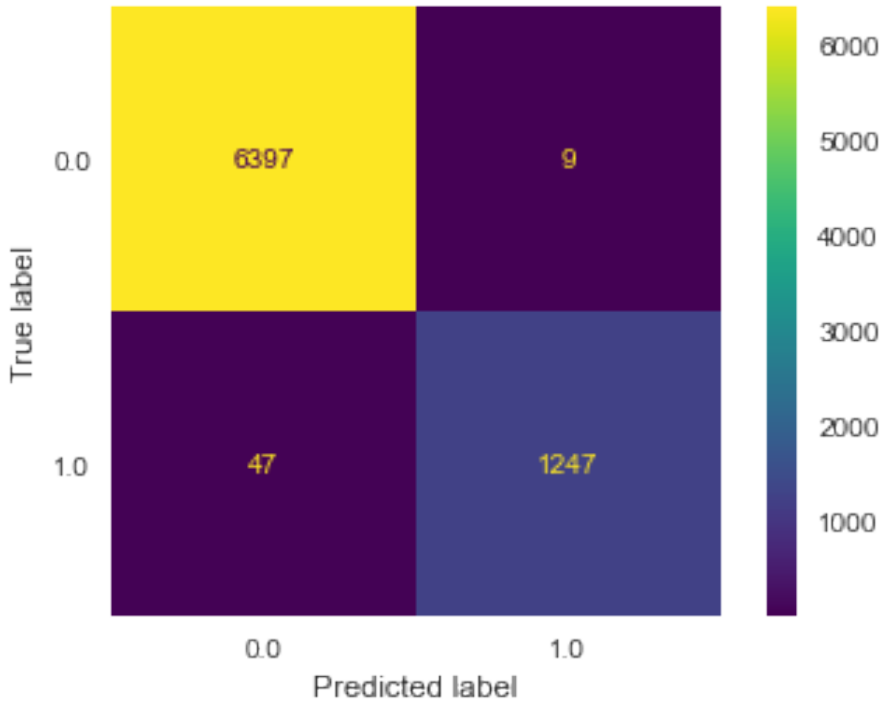


Fig.116 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 0.999
```

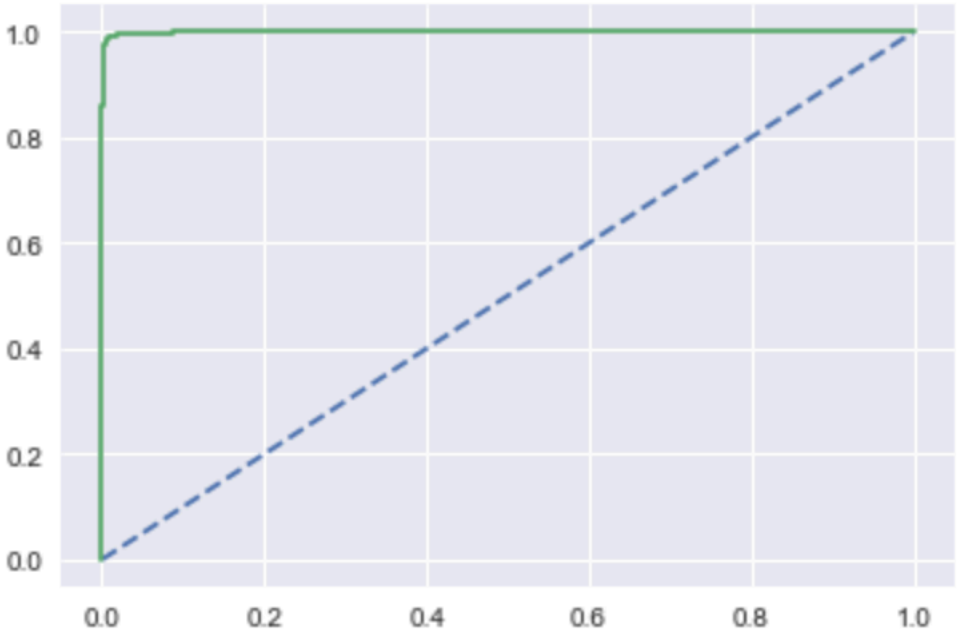


Fig.117 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.979
```

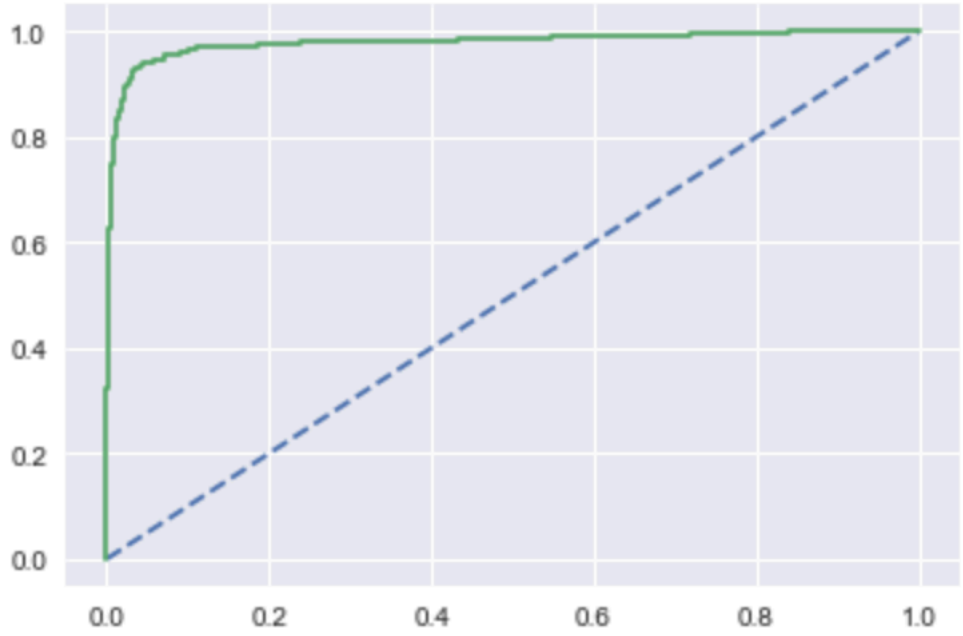


Fig.118 AUC and ROC curve of testing data

INFERENCE:

- The model's performance on the training dataset is very high, with accuracy, recall, precision, and F1 score all above 0.98.
- However, the performance on the testing dataset is much lower, with an accuracy score of 0.88, indicating the model may be overfitting to the training data.
- The recall score is relatively high, indicating that the model is able to identify most of the positive cases, but the precision score is lower, indicating that the model produces a moderate number of false positives.
- The F1 score is also relatively low, indicating a trade-off between precision and recall.
- Overall, while the ANN model with tuned hyperparameters performs well on the training data, it may not generalize well to new data.

Building KNN model:

K-Nearest Neighbors to the provided datapoint are examined by the KNN classifier. It bases its determination of the target value on its neighbours. KNN operates on the presumption that all data points that are close to one another belong to the same class. Moreover, the model is uninterpretable and a black box. It may be computationally costly and require more memory to retain training data because it is non-parametric. Moreover, it tends to overfit.

The decision has been made not to choose this model as the best model, even though it was extensively adjusted and tested on the provided data for the aforementioned reasons. So, just the most basic information about this method is provided here.

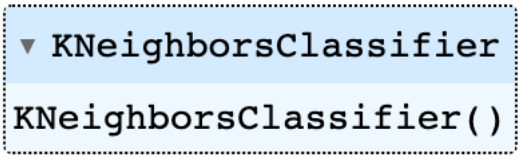


Fig.119 Parameters used for KNN model

Metrics score:

```
Accuracy on training set :  0.9771428571428571
Accuracy on test set :  0.9512269009391093
Recall on training set :  0.8964451313755796
Recall on test set :  0.8010752688172043
Precision on training set :  0.9650582362728786
Precision on test set :  0.8993963782696177
F1 on training set :  0.9294871794871794
F1 on test set :  0.8473933649289099
```

Fig.120 Metrics score of KNN model

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.98	0.99	0.99	6406
1.0	0.97	0.90	0.93	1294
accuracy			0.98	7700
macro avg	0.97	0.94	0.96	7700
weighted avg	0.98	0.98	0.98	7700

Fig.121 Classification report of training data

Confusion matrix of training set:

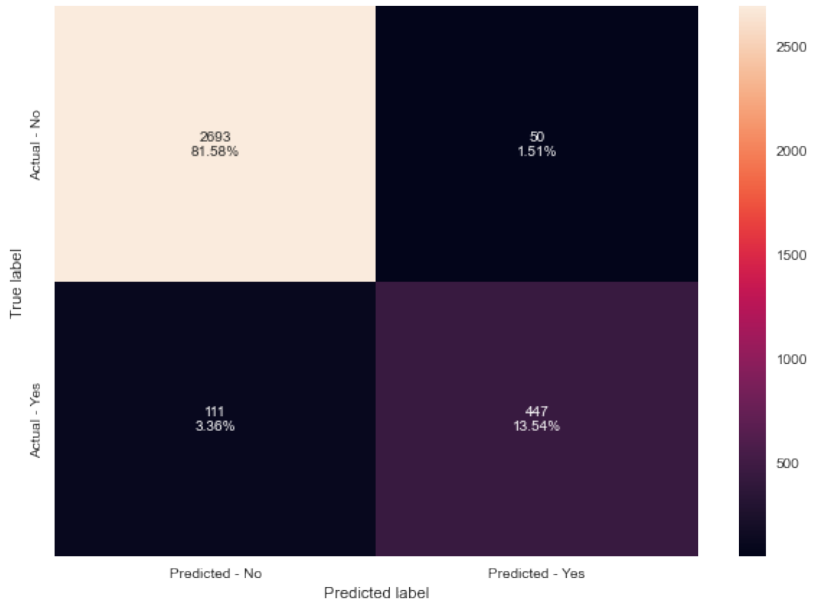


Fig.122 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.96	0.98	0.97	2743
1.0	0.90	0.80	0.85	558
accuracy			0.95	3301
macro avg	0.93	0.89	0.91	3301
weighted avg	0.95	0.95	0.95	3301

Fig.123 Classification report of testing data

Confusion matrix of testing data:

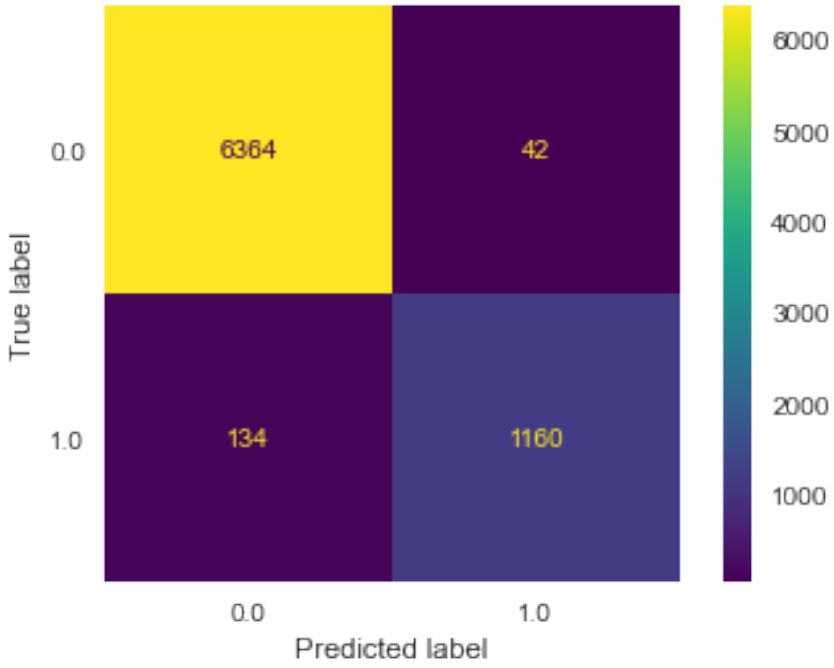


Fig.124 Confusion matrix of testing data

AUC and ROC curve of training data:

AUC: 0.995

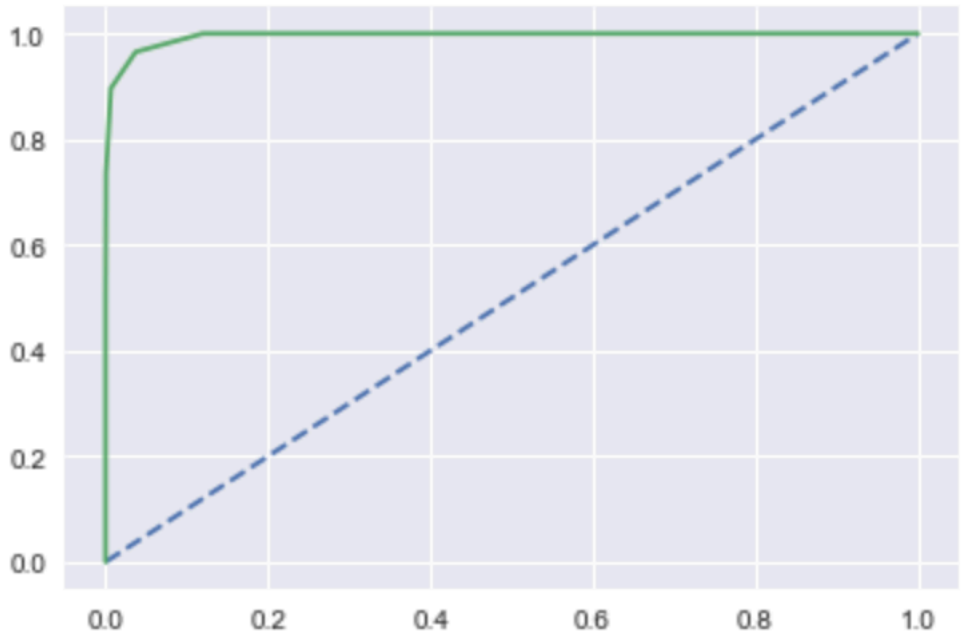


Fig.125 AUC and ROC curve of training data

AUC and ROC curve of testing data:

AUC: 0.976

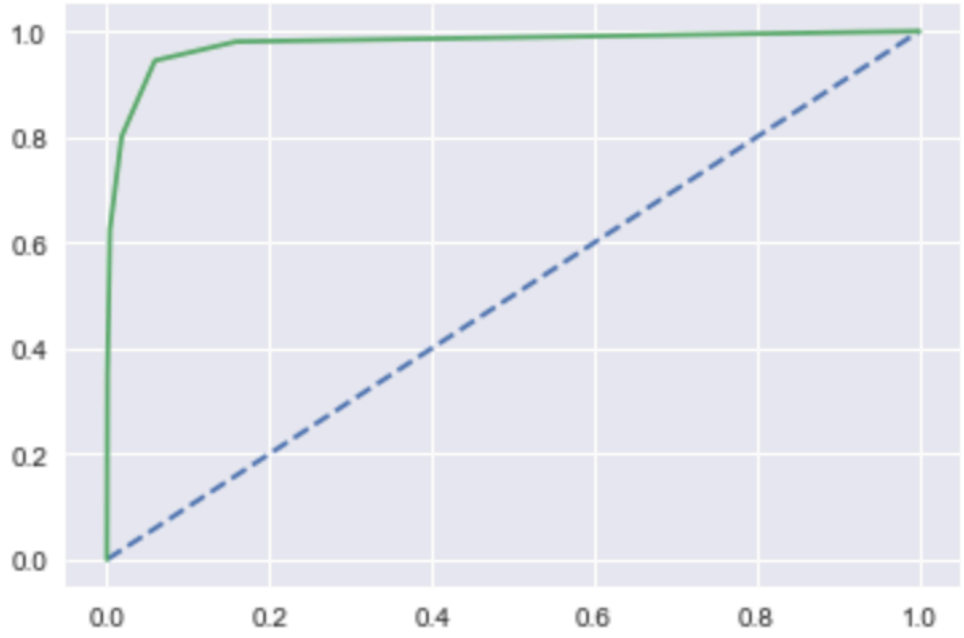


Fig.126 AUC and ROC curve of testing data

INFERENCE:

- Compared to the KNN-Tuned model, the KNN model has slightly higher accuracy on both the training and testing datasets, but still lower than the other models evaluated.
- The recall score for the testing dataset is also relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than the other models evaluated, with lower accuracy, recall, and F1 score on the testing dataset than all of the other models, and lower precision on the testing dataset than the Gradient Boosting Tuned, XGBoost-Tuned, and Bagging with Decision Tree models.

Building tuned KNN model:

```
Best parameters for KNN: {'n_neighbors': 3}
```

Fig.127 Parameters used for tunedKNN model

Metrics score:

```
Accuracy on training set :  0.9858441558441559
Accuracy on test set :  0.96485913359588
Recall on training set :  0.9412673879443586
Recall on test set :  0.8888888888888888
Precision on training set :  0.973621103117506
Precision on test set :  0.9018181818181819
F1 on training set :  0.9571709233791749
F1 on test set :  0.8953068592057761
```

Fig.128 Metrics score of tuned KNN model

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	0.99	0.99	0.99	6406
1.0	0.97	0.94	0.96	1294
accuracy			0.99	7700
macro avg	0.98	0.97	0.97	7700
weighted avg	0.99	0.99	0.99	7700

Fig.129 Classification report of training data

Confusion matrix of training set:

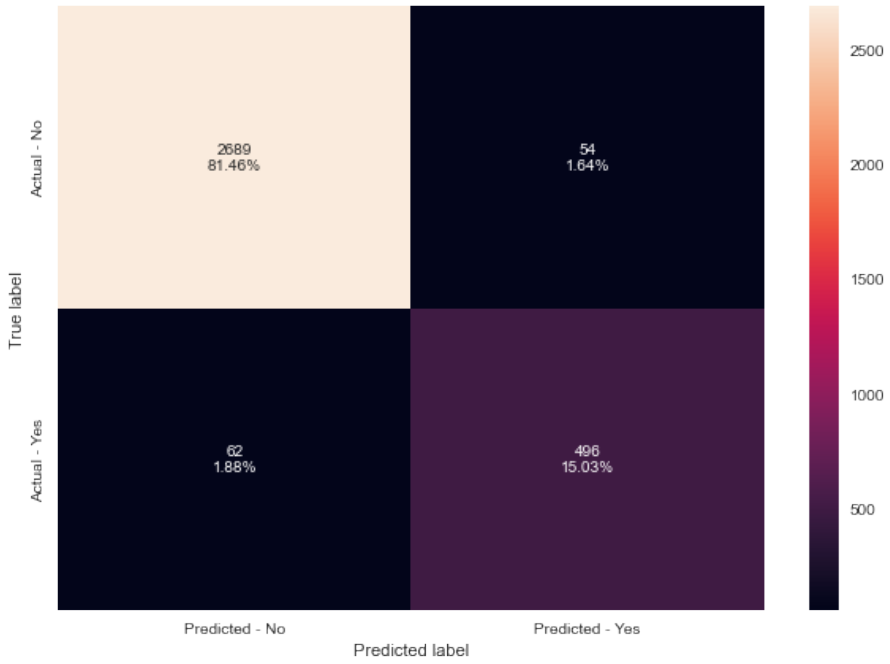


Fig.130 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.98	0.98	0.98	2743
1.0	0.90	0.89	0.90	558
accuracy			0.96	3301
macro avg	0.94	0.93	0.94	3301
weighted avg	0.96	0.96	0.96	3301

Fig.131 Classification report of testing data

Confusion matrix of testing data:

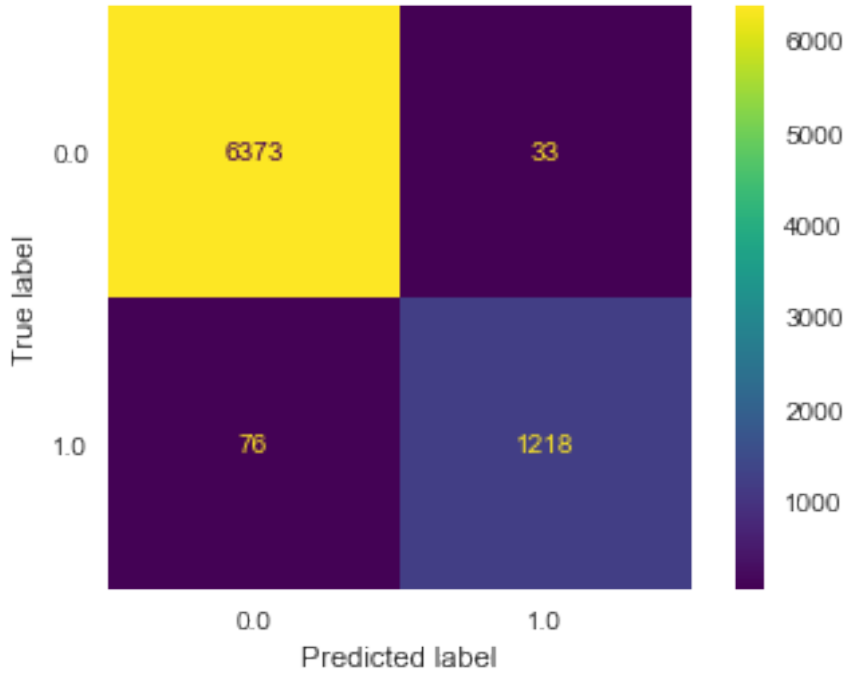
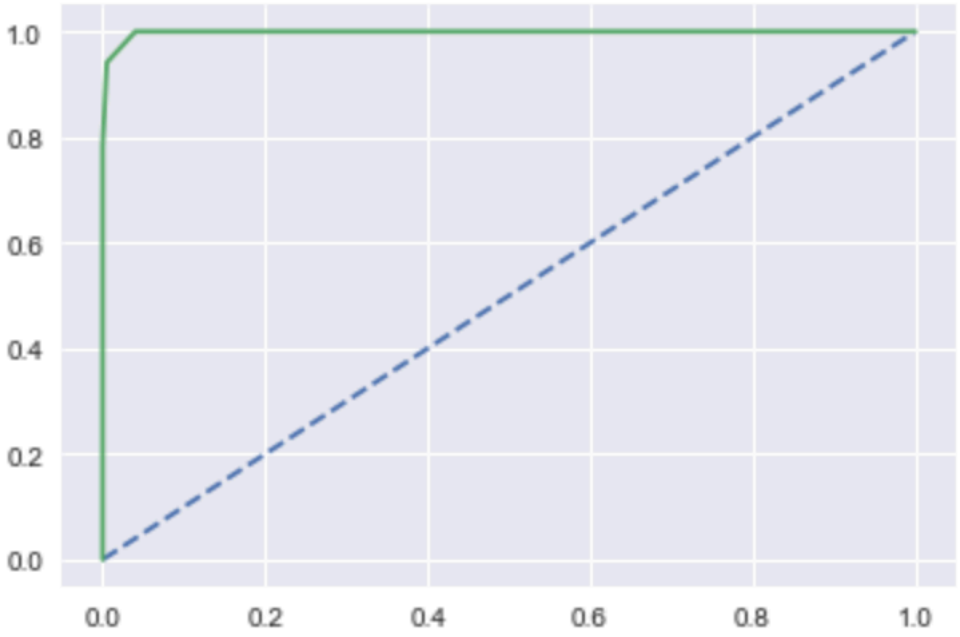


Fig.132 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 0.998
```



$

Fig.133 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.977
```

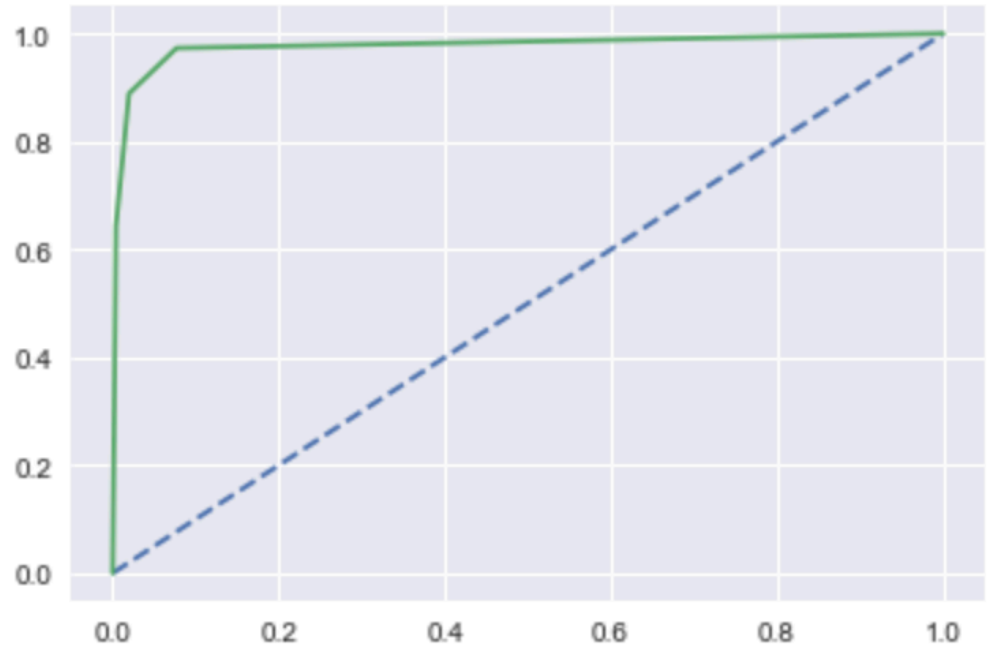


Fig.134 AUC and ROC curve of testing data

INFERENCE:

- Compared to the previous models, the KNN-Tuned model has lower accuracy on both the training and testing datasets, indicating that it may not be as effective at predicting outcomes.
- The recall score for the testing dataset is also relatively low, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is also relatively low, indicating a trade-off between precision and recall.
- Overall, this model performs worse than the other models evaluated, with lower accuracy, recall, and F1 score on the testing dataset than all of the other models, and lower precision on the testing dataset than the Gradient Boosting Tuned, XGBoost-Tuned, and Bagging with Decision Tree models.

Building a model of bagging with decision tree as base estimator:

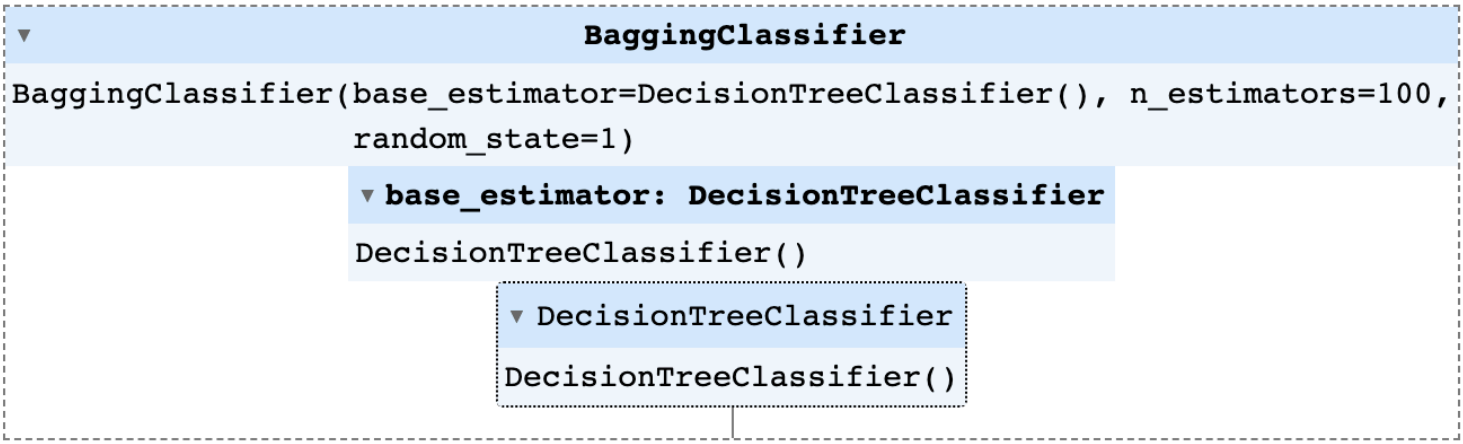


Fig.135 Parameters used for bagging with decision tree as base estimator model

Metrics score:

```
Accuracy on training set :  1.0
Accuracy on test set :  0.965
Recall on training set :  1.0
Recall on test set :  0.853
Precision on training set :  1.0
Precision on test set :  0.935
F1 on training set :  1.0
F1 on test set :  0.892
```

Fig.136 Metrics score of bagging with decision tree as base estimator model

Classification report of training set:

Classification Report of Training Data

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	6406
1.0	1.00	1.00	1.00	1294
accuracy			1.00	7700
macro avg	1.00	1.00	1.00	7700
weighted avg	1.00	1.00	1.00	7700

Fig.137 Classification report of training data

Confusion matrix of training set:

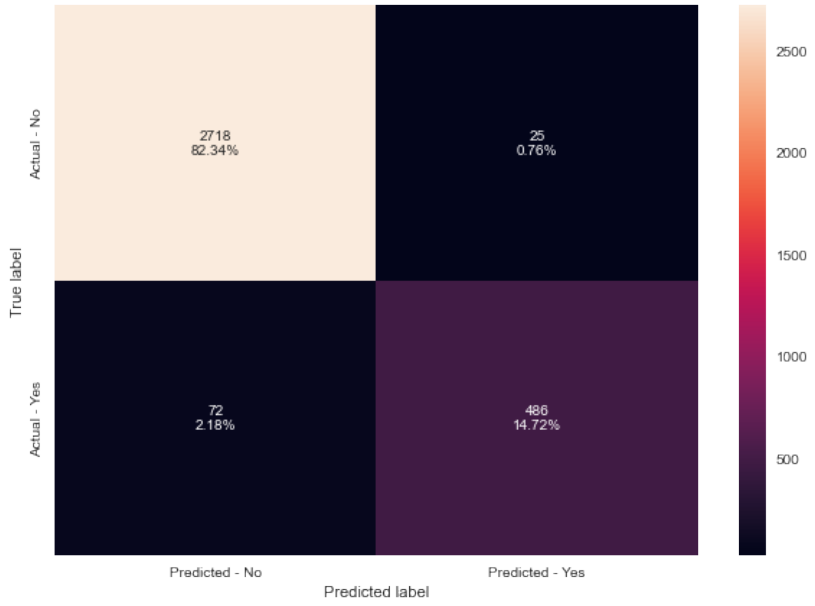


Fig.138 Confusion matrix of training data

Classification report of testing data:

Classification Report of Testing Data

	precision	recall	f1-score	support
0.0	0.97	0.99	0.98	2743
1.0	0.94	0.85	0.89	558
accuracy			0.97	3301
macro avg	0.95	0.92	0.94	3301
weighted avg	0.96	0.97	0.96	3301

Fig.139 Classification report of testing data

Confusion matrix of testing data:

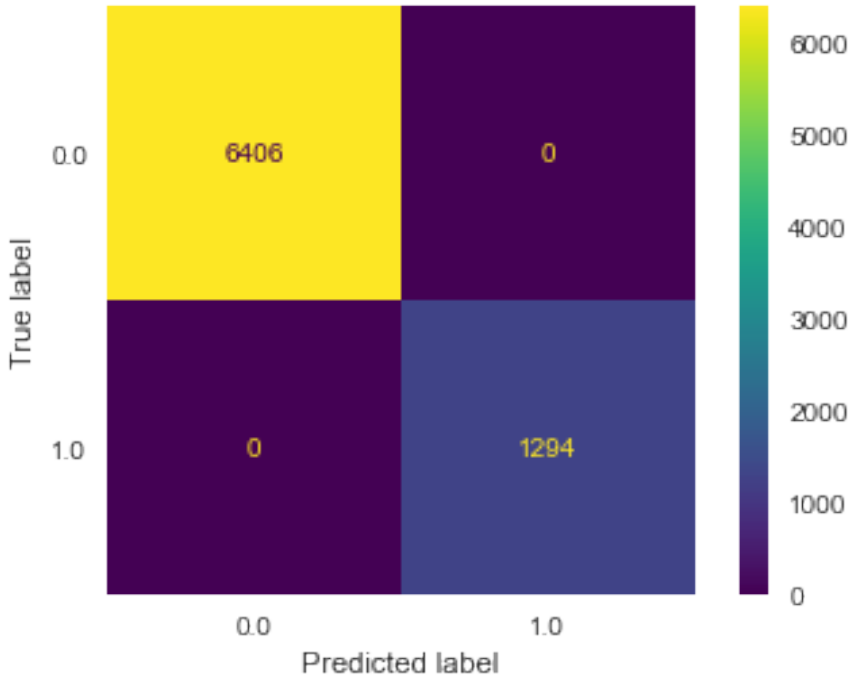


Fig.140 Confusion matrix of testing data

AUC and ROC curve of training data:

```
AUC: 1.000
```

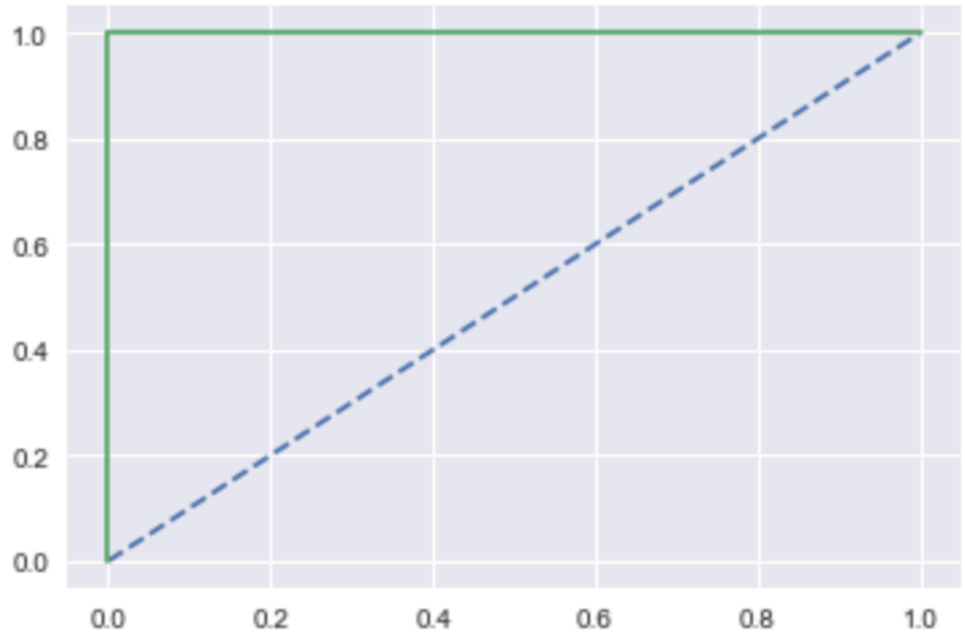


Fig.141 AUC and ROC curve of training data

AUC and ROC curve of testing data:

```
AUC: 0.989
```

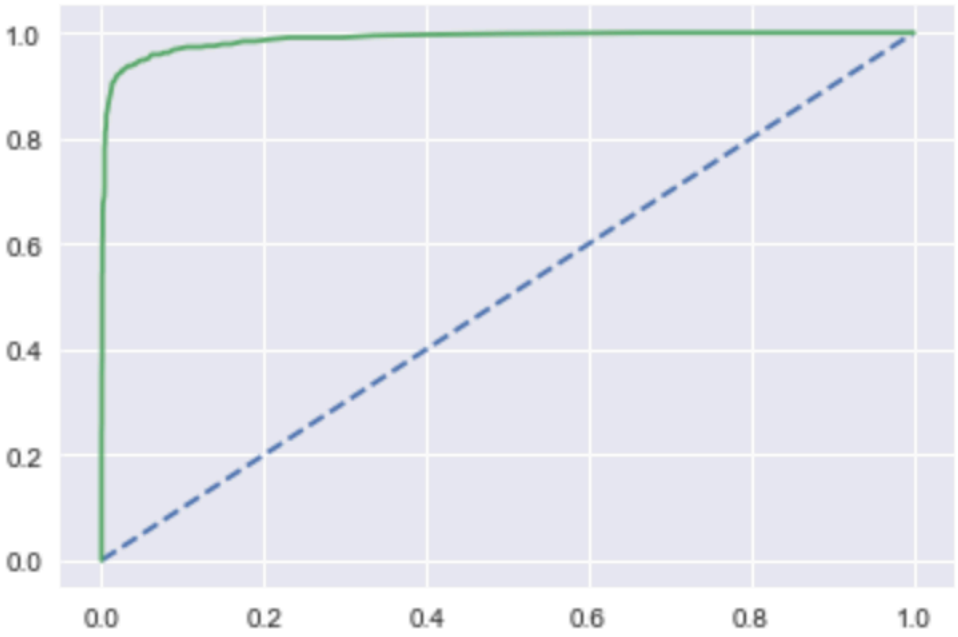


Fig.142 AUC and ROC curve of testing data

INFERENCE:

- Similar to the previous models, the Bagging with Decision Tree model has achieved near-perfect accuracy on the training dataset (1.00), which suggests that it has likely overfit the training data.
- However, it still has relatively high accuracy on the testing dataset (0.97), which suggests that it is able to generalize well to unseen data.
- The recall score for the testing dataset is lower than the training dataset, indicating that the model is not able to identify all of the positive cases.
- The precision score for the testing dataset is higher than the recall score, suggesting that the model produces fewer false positives.
- The F1 score for the testing dataset is lower than the training dataset, indicating a trade-off between precision and recall.
- Overall, this model performs similarly to the Random Forest model, with slightly lower recall and F1 score on the testing dataset, but higher precision on the testing dataset than both the Gradient Boosting Tuned and XGBoost-Tuned models.

Comparing all models:

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_f1	Test_f1
9	Gradient Boosting Tuned	1.00	0.98	1.00	0.92	1.00	0.97	1.00	0.95
8	XGBoost-Tuned	1.00	0.97	1.00	0.91	1.00	0.92	1.00	0.91
10	Random Forest	1.00	0.97	1.00	0.86	1.00	0.95	1.00	0.91
16	Bagging with Decision Tree	1.00	0.97	1.00	0.87	1.00	0.95	1.00	0.91
7	XGBoost	1.00	0.97	1.00	0.89	1.00	0.92	1.00	0.90
15	KNN-Tuned	0.99	0.96	0.94	0.89	0.97	0.90	0.96	0.90
13	ANN-Tuned	0.98	0.88	0.96	0.87	0.99	0.90	0.98	0.88
14	KNN	0.98	0.95	0.90	0.80	0.97	0.90	0.93	0.85
5	AdaBoost Tuned	1.00	0.94	1.00	0.86	1.00	0.81	1.00	0.83
11	Random Forest - Tuned	0.96	0.93	0.78	0.67	0.98	0.91	0.87	0.77
12	ANN	0.95	0.93	0.78	0.72	0.90	0.83	0.83	0.77
6	Gradient Boosting with default parameters	0.92	0.90	0.62	0.58	0.85	0.79	0.72	0.67
4	AdaBoost with default paramters	0.90	0.89	0.58	0.59	0.74	0.73	0.65	0.65
2	Logistic Regression-Tuned	0.88	0.88	0.45	0.46	0.74	0.73	0.56	0.57
1	Logistic Regression	0.88	0.88	0.43	0.45	0.75	0.74	0.55	0.56
3	LDA	0.88	0.88	0.41	0.44	0.74	0.74	0.53	0.55
0	Logistic Regression with Smote	0.79	0.77	0.82	0.81	0.43	0.41	0.56	0.55

Fig.143 Comparison table of all models

CONCLUSION:

- Based on the above comparison table, the Gradient Boosting model with tuned parameters is the best suited for a classification problem like Customer Churn analysis.
- This is because it has the highest accuracy score on the testing dataset (0.98), which indicates that it can accurately predict whether a customer is likely to churn or not.
- Additionally, it also has high recall (0.92) and precision (0.97) scores on the testing dataset, which means that it has a low false negative rate and a low false positive rate, respectively. This is important in a churn analysis because the cost of retaining a customer who was predicted to churn is lower than losing a customer who was predicted to stay.
- Therefore, a model with high precision and recall is preferable for this problem.
- Finally, the high F1 score (0.95) on the testing dataset suggests that this model has a good balance between precision and recall, making it the best optimized model for the customer churn analysis problem.
- Compared to XGBoost-Tuned, Gradient Boosting Tuned has a slightly higher accuracy score on the testing dataset (0.98 vs. 0.97) and higher precision score on the testing dataset (0.97 vs. 0.92). However, XGBoost-Tuned has a slightly higher recall score on the testing dataset (0.91 vs. 0.92) and a comparable F1 score (0.91 vs. 0.95). Overall, the two models are very similar, but Gradient Boosting Tuned has a slight edge in terms of precision and accuracy.
- Compared to Random Forest, Gradient Boosting Tuned has a higher accuracy score on the testing dataset (0.98 vs. 0.97), a higher recall score on the testing dataset (0.92 vs. 0.86), and a comparable F1 score (0.95 vs. 0.91). However, Random Forest has a higher precision score on the testing dataset (0.95 vs. 0.97). Overall, Gradient Boosting Tuned performs slightly better in terms of accuracy and recall, but Random Forest performs slightly better in terms of precision.
- Compared to Bagging with Decision Tree, Gradient Boosting Tuned has a higher accuracy score on the testing dataset (0.98 vs. 0.97), a higher recall score on the testing dataset (0.92 vs. 0.87), and a higher F1 score on the testing dataset (0.95 vs. 0.91). However, Bagging with Decision Tree has a higher precision score on the testing dataset (0.95 vs. 0.97). Overall, Gradient Boosting Tuned performs better in terms of accuracy, recall, and F1 score, while Bagging with Decision Tree performs slightly better in terms of precision.
- Compared to KNN-Tuned, Gradient Boosting Tuned has a higher accuracy score on the testing dataset (0.98 vs. 0.96), a higher recall score on the testing dataset (0.92 vs. 0.89), and a higher F1 score on the testing dataset (0.95 vs. 0.90). However, KNN-Tuned has a slightly higher precision score on the testing dataset (0.90 vs. 0.97). Overall, Gradient Boosting Tuned performs better in terms of accuracy, recall, and F1 score, while KNN-Tuned performs slightly better in terms of precision.
- Compared to the other models, the default Gradient Boosting model has lower performance scores, including lower accuracy, recall, precision, and F1 scores on both the training and testing datasets. This suggests that the default model is not well-optimized for the churn analysis problem and may benefit from hyperparameter tuning.

- Overall, the Gradient Boosting Tuned model performs well across multiple metrics, including accuracy, precision, recall, and F1 score, making it the best optimized model for the customer churn analysis problem.

Model interpretations and business implications:

- The high accuracy score of 1.0 on the training set and 0.9824 on the test set indicate that the tuned gradient boosting model is performing very well in predicting customer churn, and it is not overfitting.
- The recall score of 1.0 on the training set and 0.9247 on the test set indicates that the model is able to identify most of the customers who are likely to churn, and only a small percentage of those who are not likely to churn will be identified as churners.
- The high precision score of 1.0 on the training set and 0.9699 on the test set indicates that the model is able to accurately predict the customers who are likely to churn, and only a small percentage of those who are not likely to churn will be identified as churners.
- The high f1-score of 1.0 on the training set and 0.9467 on the test set suggests that the model is able to balance the trade-off between precision and recall.
- The AUC score of 1.0 on the training set and 0.996 on the test set suggest that the model has a high ability to distinguish between positive and negative classes, which indicates a good discrimination power of the model.
- Tuned Gradient boosting:

Fig.144 Feature importance of tuned gradient boosting model

- **INFERENCE:**

- This model has picked account_tenure to be the most important feature followed by days_since_cc_contact an cc_agent_score.
- According to EDA, the turnover is higher for shorter tenures, particularly in the first year. In order to maintain a customer's satisfaction after acquisition, the first year is crucial.
- Compared to current customers, churned consumers had more recent interactions with customer service. According to EDA, active clients have a longer median number of days since their previous connection.
- Prior to churning, recently contacted consumers have called customer service. According to the tuned gradient boosting model, the customer satisfaction rating is also significant. This is in line with the trends and revelations that came from EDA.

- Random forest:

Fig.145 Feature importance of Random forest model

- **INFERENCE:**
  - This model has picked account_tenure, cashback and cust_care_contacts_12m as its 3 important features.
  - According to EDA, the turnover is higher for shorter tenures, particularly in the first year. In order to maintain a customer's satisfaction after acquisition, the first year is crucial.
  - Cashback feature indicates the cash received back by customers during each payment. If this is also in lead then we can say that customers are satisfied with the company's service and they tend to continue the service for the next few months at the least.

- The third feature indicates how many times all of the account's customers have been in touch with customer service over the past 12 months. If their experience was good then the service will also be prolonged.

- XG boost model:

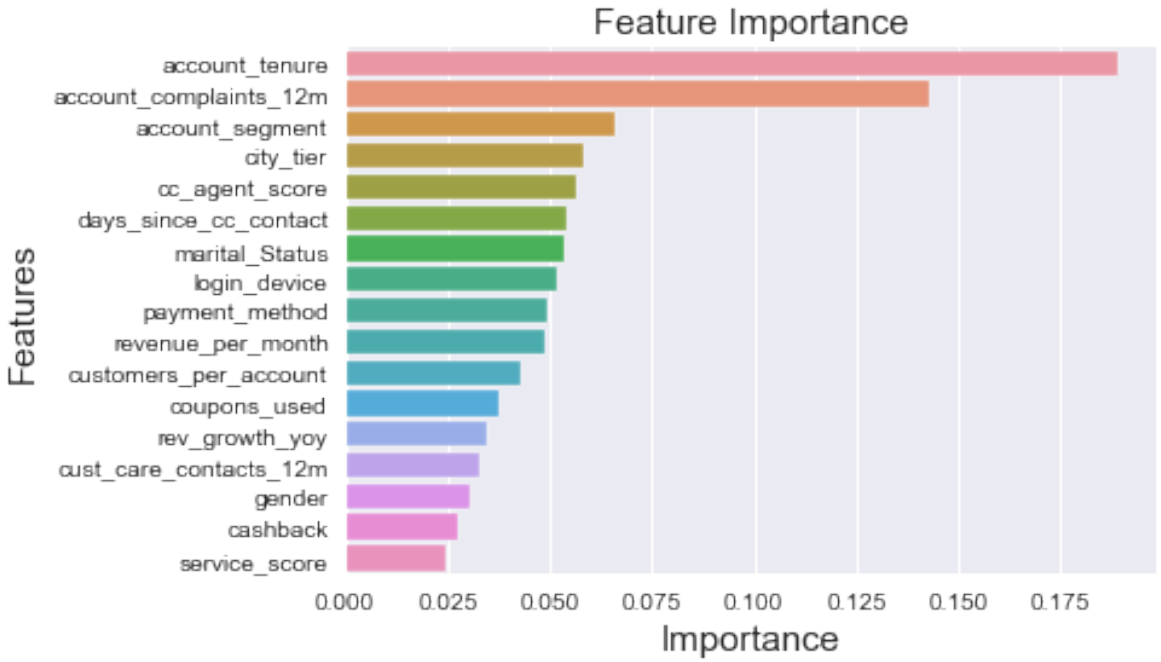


Fig.146 Feature importance of XG Boost model

- **INFERENCE:**
  - This model has picked account_tenure, account_complaints_12m and account_segment as its 3 important features.
  - Based on the exploratory data analysis (EDA), it has been observed that the likelihood of turnover is greater for customers who have shorter tenures, especially within the first year. This indicates that the initial year of acquiring a customer is critical for ensuring their satisfaction and preventing them from leaving.
  - Based on the model, the next indicator of potential customer churn is if a complaint was lodged in the past year. EDA findings suggest that if a complaint was made, there is a higher likelihood of churn.
  - If customer segmentation is identified as the third important feature in the XG Boost model for customer churn analysis, it implies that customers with different profiles and behavior patterns have different likelihoods of churning. Therefore, segmenting customers based on their attributes and behavior can help companies identify high-risk customers who are likely to churn and implement targeted retention strategies to prevent them from leaving. This approach allows companies to focus their retention efforts on customers who are most likely to churn and avoid wasting resources on customers who are unlikely to leave.

Business implications:

- The business implications of a well-performing customer churn model are significant. By accurately predicting the customers who are likely to churn, companies can proactively take steps to retain those customers.
- This could include offering special incentives or discounts, providing personalized support, or improving the quality of their products or services.
- Additionally, being able to identify customers who are likely to churn can help companies to focus their marketing and customer retention efforts more effectively, resulting in higher customer satisfaction and increased revenue in the long run.
- By accurately identifying customers who are likely to churn, the business can proactively engage with them through targeted campaigns or personalized offers. This can improve customer satisfaction and loyalty, and ultimately lead to improved retention rates.
- Acquiring new customers can be costly for businesses, and it is often more cost-effective to retain existing customers. By identifying potential churners, businesses can focus their retention efforts on these customers, which can result in significant cost savings.
- By utilizing advanced machine learning techniques to predict customer churn, businesses can gain a competitive advantage over their competitors. This can lead to increased market share, higher revenue, and greater profitability.
- By proactively engaging with potential churners, businesses can improve the overall customer experience. This can lead to increased customer satisfaction, positive word-of-mouth referrals, and improved brand reputation.
- By targeting campaigns and offers to potential churners, businesses can improve the ROI of their marketing efforts. This can result in higher conversion rates, increased revenue, and greater profitability.
- Overall, the optimal model's metrics indicate that it is highly accurate in predicting customer churn, which can lead to a range of positive business implications, including improved retention, cost savings, competitive advantage, improved customer experience, and improved marketing ROI.

Business recommendations:

- The data collected shows a mix of services provided and customer ratings, indicating that there is room for improvement in the services offered.
- The business can increase its visibility in tier 2 cities to acquire new customers.
- By promoting payment methods such as standing instructions in bank accounts or UPI, the business can provide a hassle-free and secure experience for customers.
- The service scores show a lot of room for improvement and the business can conduct a survey to better understand customer expectations.

- Improving customer experience can be achieved by training customer care executives and providing them with the tools they need to deliver better service.
- By offering customized plans based on customer spend and tenure, the business can improve customer loyalty and retention.
- Offering family floater plans for married customers can be a good way to increase customer satisfaction and loyalty.
- These insights provide a starting point for the business to consider new strategies for improving customer experience and reducing churn. Further analysis and research may be necessary to determine the best approach for each business and customer segment.
- By understanding customer behavior and preferences, the business can tailor its services and marketing efforts to meet the needs and expectations of its target audience. This will not only improve customer satisfaction, but also drive business growth and success.
- Business can analyze the customer churning pattern based on gender and address the reasons why male customers are more likely to churn than female customers.
- Focus on improving the customer experience for the "Regular Plus" segment, as customers in this segment have shown a higher churn rate.
- Offer tailored packages for single customers, perhaps with added benefits or discounts to reduce churn rates among this customer segment.
- Conduct a detailed study of customer preferences for accessing services through mobile devices and see if there are any pain points that can be addressed to reduce churn rates among mobile users.
- Develop marketing strategies aimed at increasing customer retention and loyalty, such as loyalty programs or rewards for repeat business.
- Offer financial incentives or flexible payment options to reduce churn among customers who prefer debit or credit cards.
- Invest in technology to streamline customer service processes and provide faster and more efficient service, which could help to improve customer satisfaction and reduce churn.