

Capstone Project – Project Notes -2 Submission

|  |  |
| --- | --- |
| NAME | DEEPAK SINGH |
| PROJECT | CUSTOMER CHURN |
| GROUP | PGP-DSBA |
| SUBMISSION DATE | 4TH FEB 2024 |

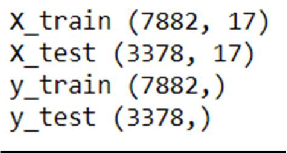
**Model Building**

In this part of capstone project, we will move towards various model building after EDA and data cleaning performed earlier followed by model tuning and assessing the performance over different metrics Accuracy, F1 Score, Recall, Precession, ROC curve, AUC score, Confusion matrix and classification report. We will choose the model which does not underfit or overfit along with the best accuracy in place.

### Splitting Data into Train and Test Dataset: -

Following the accepted market practice, we have divided data into Train and Test dataset into 70:30 ratio and building various models on training dataset and testing for accuracy over testing dataset.

### Below is the shape of Train and Test dataset: -

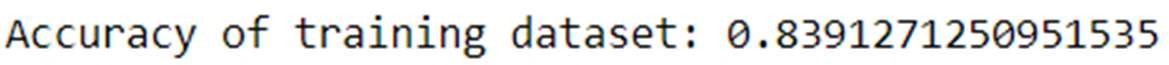
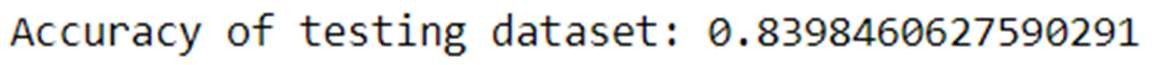


***Fig 1: - Shape of training and test dataset***

### Building Logistic Regression Model on Dataset: -

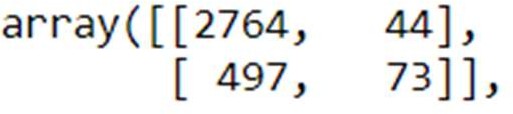
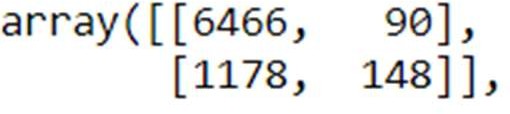
* **Building model with default hyperparameters: -**

Post splitting data into training and testing data set we fitted logistic regression model into training dataset and performed prediction on training and testing dataset using the same model. We made the first model with default hyperparameters with default solver as Ibfgs. Below are the accuracy scores obtained from this model: -



***Fig 2: - Accuracy from Logistic Regression***

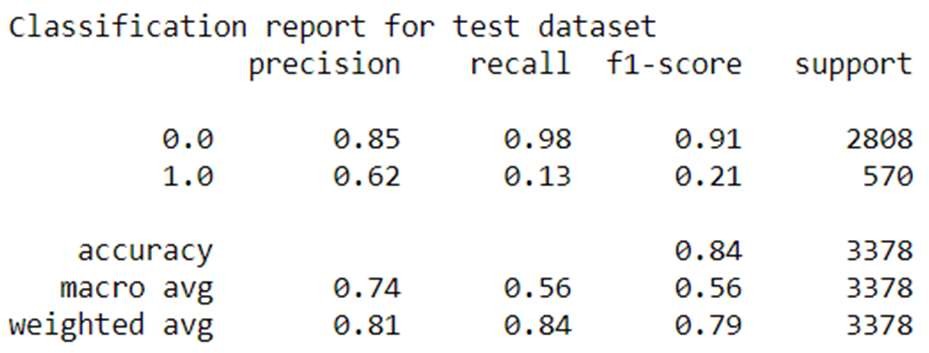
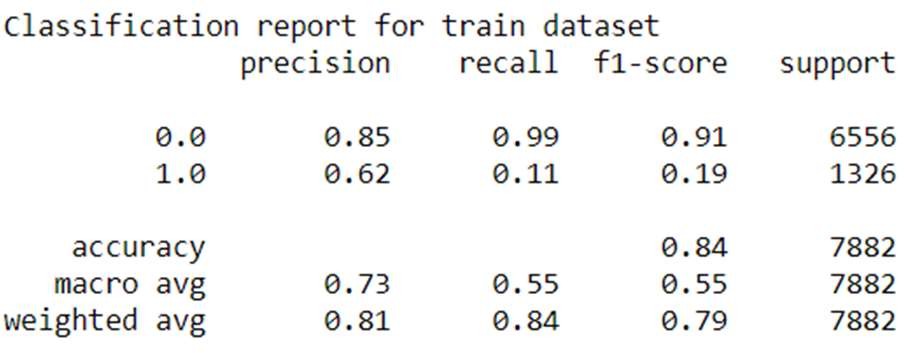
Below is the confusion matrix obtained from this model: -



### Train Test

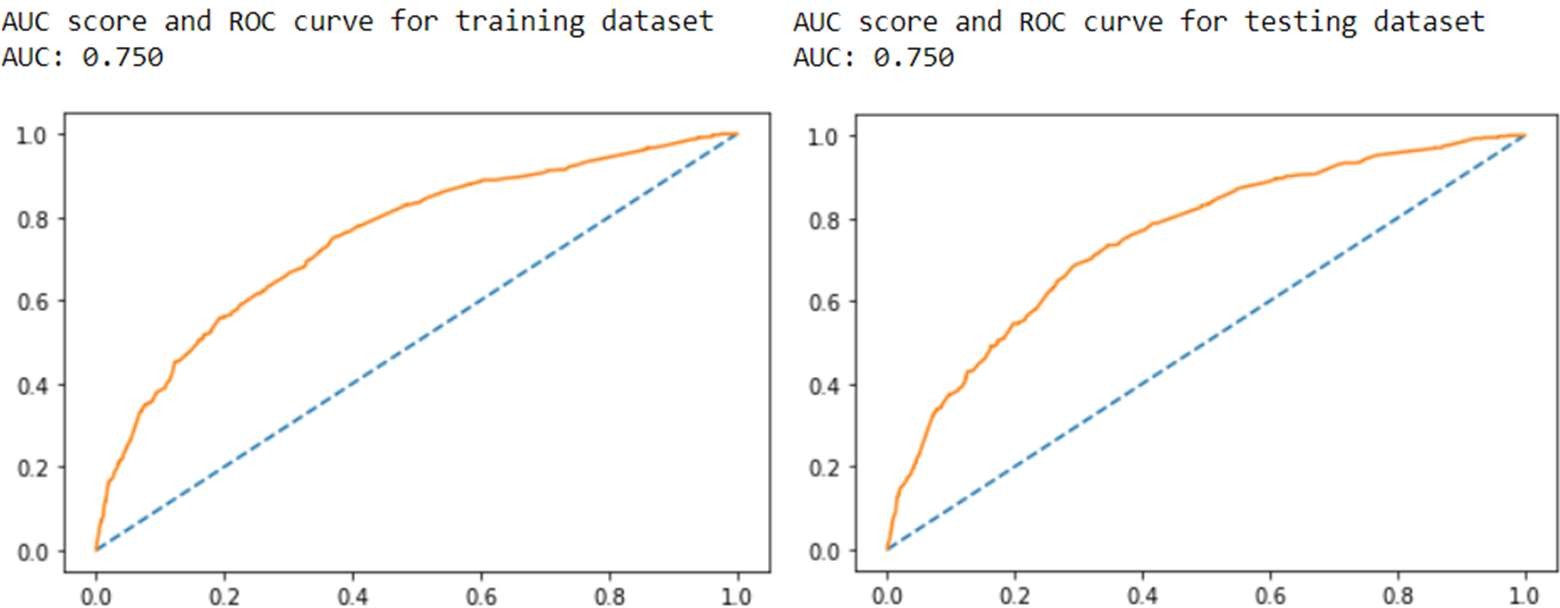
***Fig 3: - Confusion Matrix from Logistic Regression***

Below is the classification report obtained from this model: -



***Fig 4: - Classification Report from Logistic Regression***

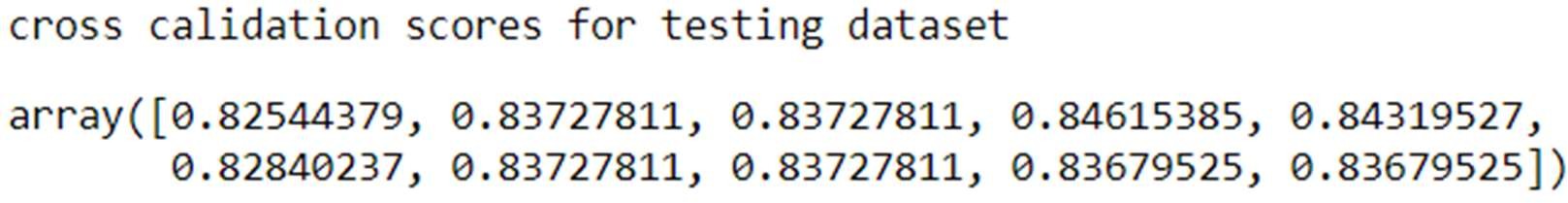
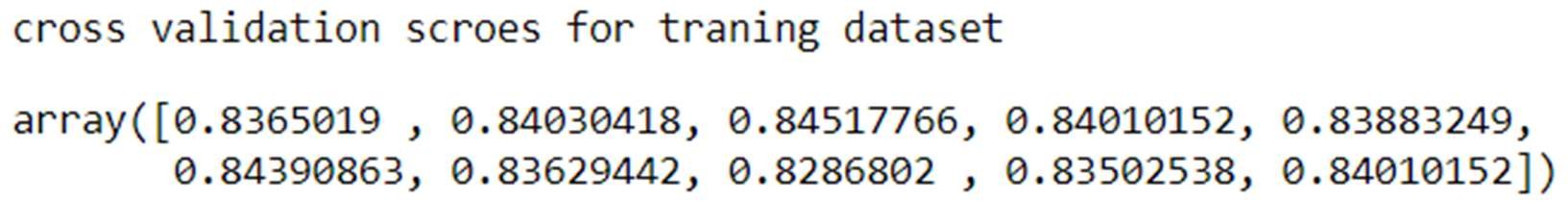
Below is the AUC score and ROC curve obtained from this model: -



***Fig 5: - ROC Curve & AUC Score From Logistic Regression***

Below are the 10-fold cross validation for logistic regression with default values: -

Cross-validation is a process to check if the built model is correct or not. Below are the 10-fold cross validation scores: -



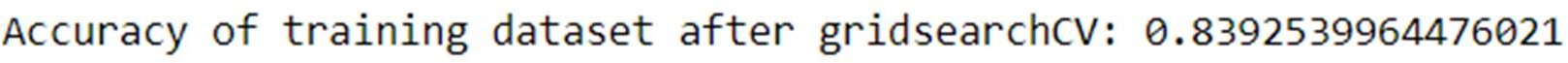
***Fig 6: - Cross Validation Scores From Logistic Regression***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

### Building model using GridSearchCV and analyzing the best parameters: -

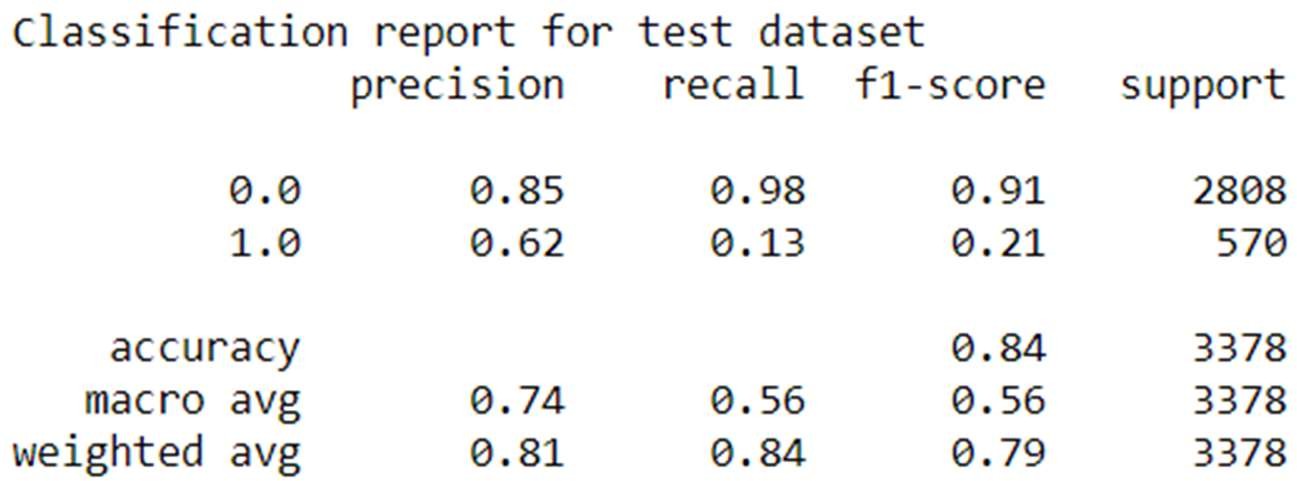
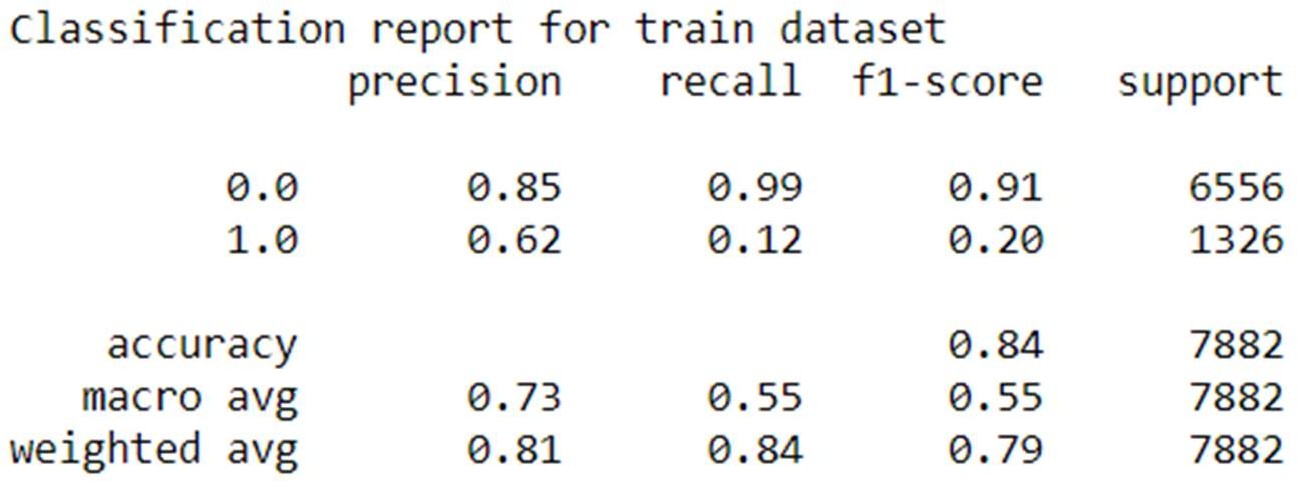
We use GridSearchCV to find an optimal combination of hyperparameters that minimizes a predefined loss function to give better results. Performed GridSearchCV with various hyper- parameters like “solver”, “penalty” and “tol” and we can find that “ibfgs” solver along with “none” penalty worked as best parameters for this dataset. However, we have also observed that the difference in accuracy for train and test dataset is very marginal and not much of significant.

Below are the accuracy scores obtained from this model using GridSearchCV: -



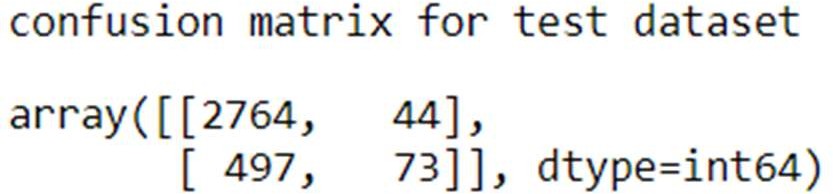
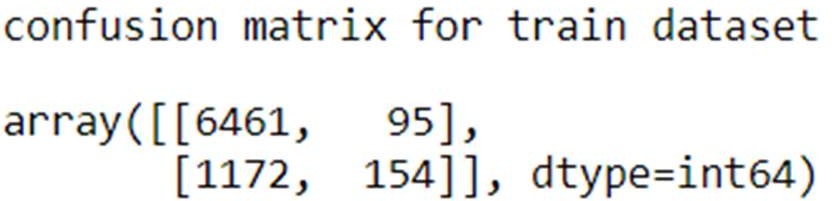
***Fig 7: - Accuracy from Logistic Regression using hyper-parameter***

Below is the classification report obtained from this model using GridSearchCV: -



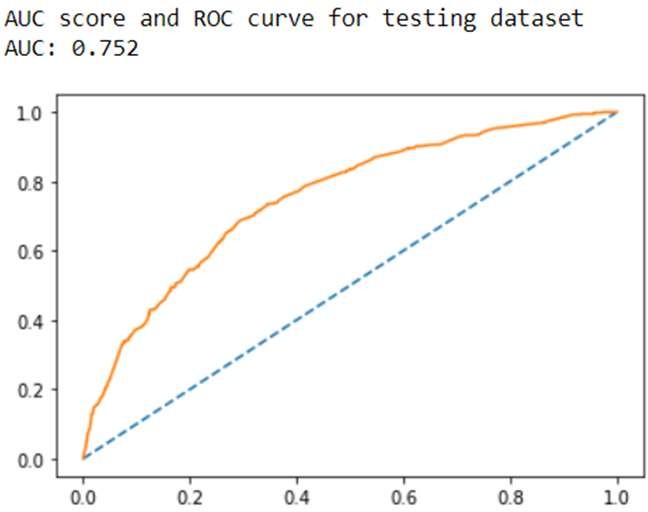
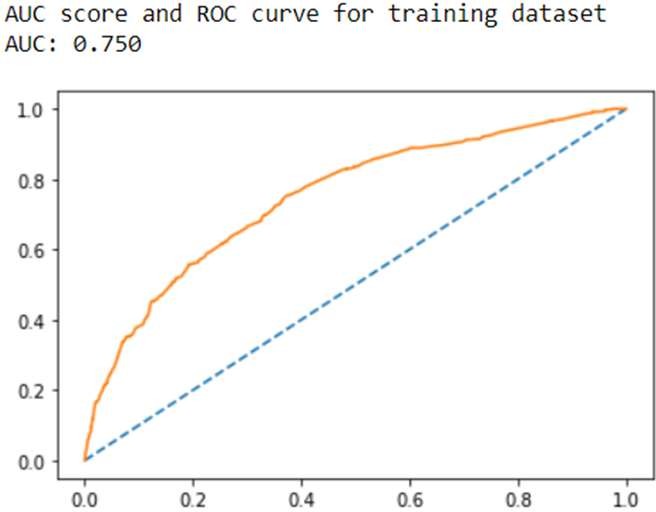
***Fig 8: - Classification Report from Logistic Regression using hyper-parameter***

Below is the confusion matrix obtained from this model using GridSearchCV: -



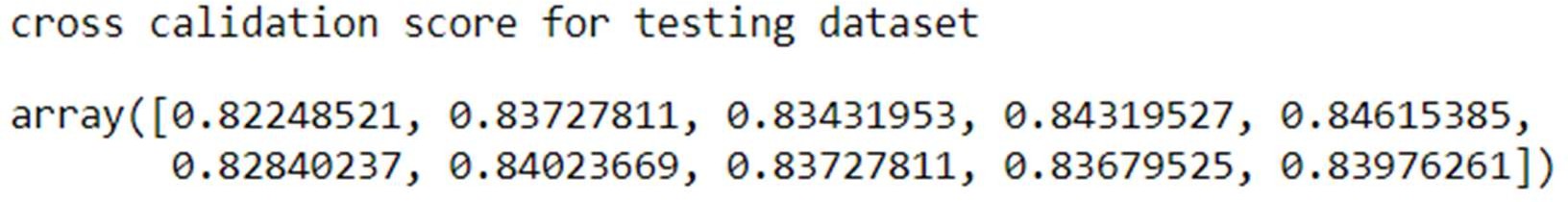
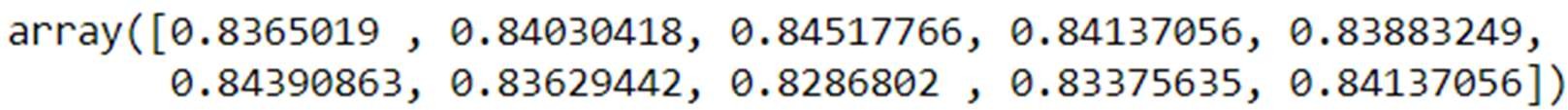
***Fig 9: - Confusion Matrix from Logistic Regression using hyper-parameter***

Below is the AUC score and ROC curve obtained from this model using GridSearchCV: -



***Fig 10: - ROC Curve and AUC Score from Logistic Regression using hyper-parameter***

Below are the 10-fold cross validation scores: -



***Fig 10: - Cross Validation Scores From Logistic Regression using hyper-parameter***

**Building model using SMOTE: -**

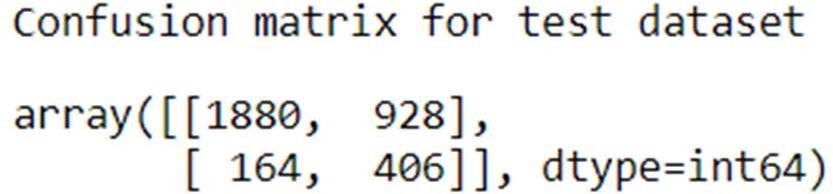
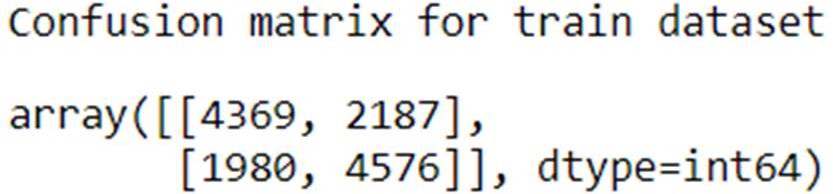
In our previous analysis we have seen that the data is imbalanced in nature. We can use SMOTE technique to balance the data and then tried building model on the balanced data to check if we can see some significant improvement in accuracy for training and testing dataset. After building model on balanced dataset and checking on accuracy we can see that the performance is not that significant in terms of accuracy.

Below are the accuracy scores obtained from balanced data: -



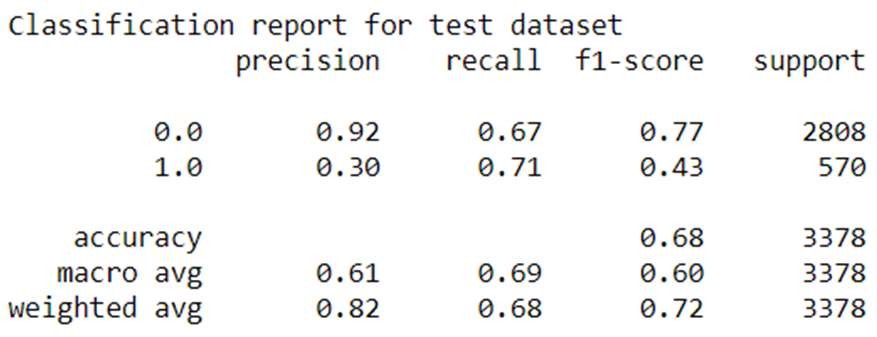
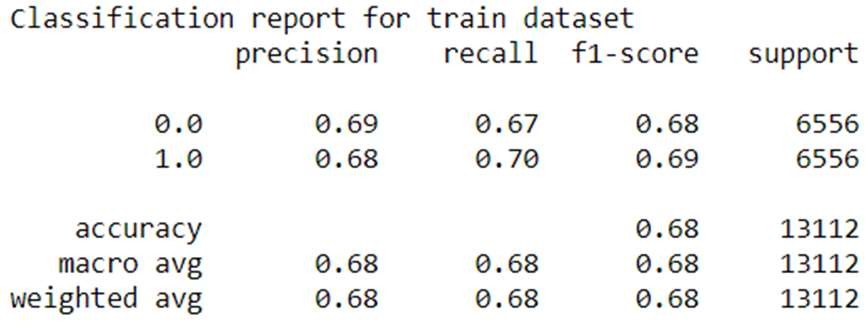
***Fig 11: - Accuracy Score from Logistic Regression with SMOTE***

Below is the confusion matrix obtained from balanced data: -



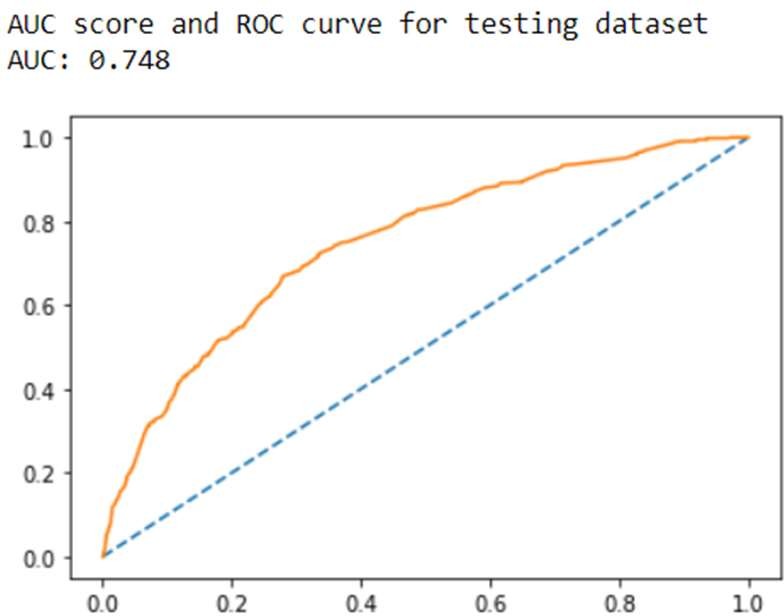
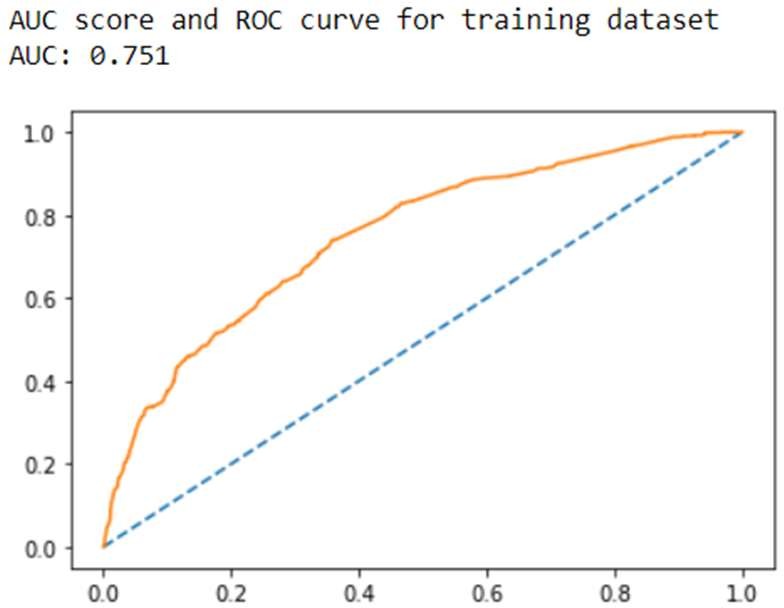
***Fig 12: - Confusion Matrix from Logistic Regression with SMOTE***

Below is the classification report obtained from balanced data: -



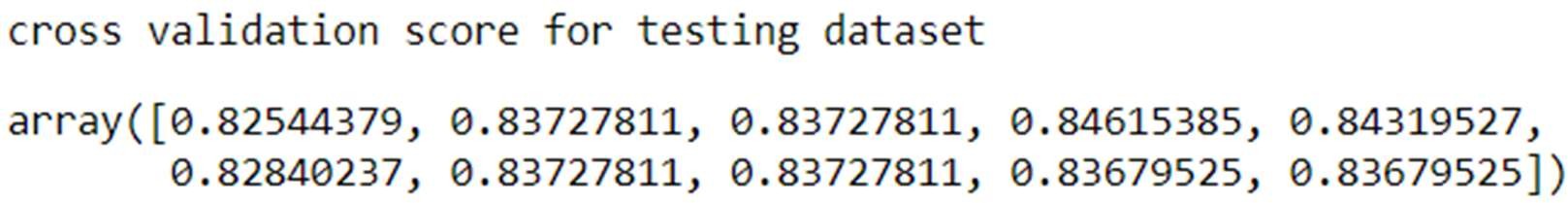
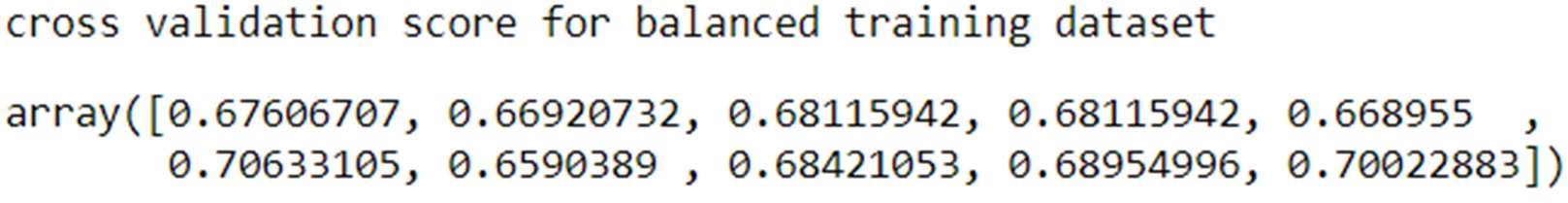
***Fig 13: - Classification Report from Logistic Regression with SMOTE***

Below are the AUC scores and ROC curve obtained from balanced data: -



***Fig 14: - ROC Curve & AUC Scores From Logistic Regression with SMOTE***

Below are the 10-fold cross validation scores: -



***Fig 15: - Cross Validation Scores From Logistic Regression with SMOTE***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

### Inference/Conclusion from Logistic Regression Model: -

From the above we can conclude that the data is neither “Overfit” nor “Underfit” in nature. And we can also inference that the model built using GridSearchCV is best optimized considering the best parameters obtained. However, the accuracy scores along with recall, precision, F1 values, ROC curve and AUC score are not that significant as compared with models built with default values and balanced data (SMOTE). Model built on balanced data set using SMOTE technique works well in training dataset however we can see a significant deplete in accuracy when it comes to testing dataset.

### Building Linear Discriminant Analysis Model (LDA)

Building model with default hyperparameters: -

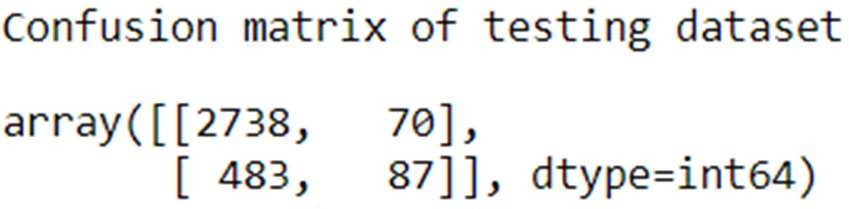
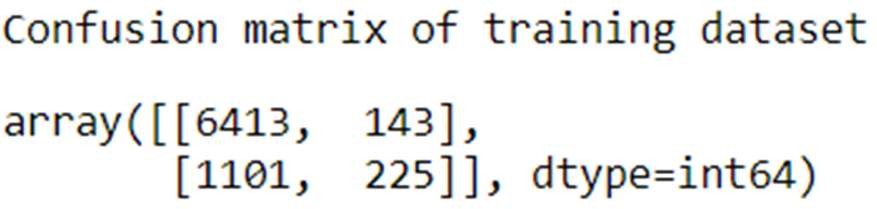
From the above split into training and testing dataset we are building Linear Discriminant Analysis (LDA) model to check it this can outperform Logistic regression model and we can choose form best for further predictions. Firstly, we are building model using default values of LDA. That is, solver as “svd” and shrinkage as “none”.

Below are the accuracy scores obtained from this model: -



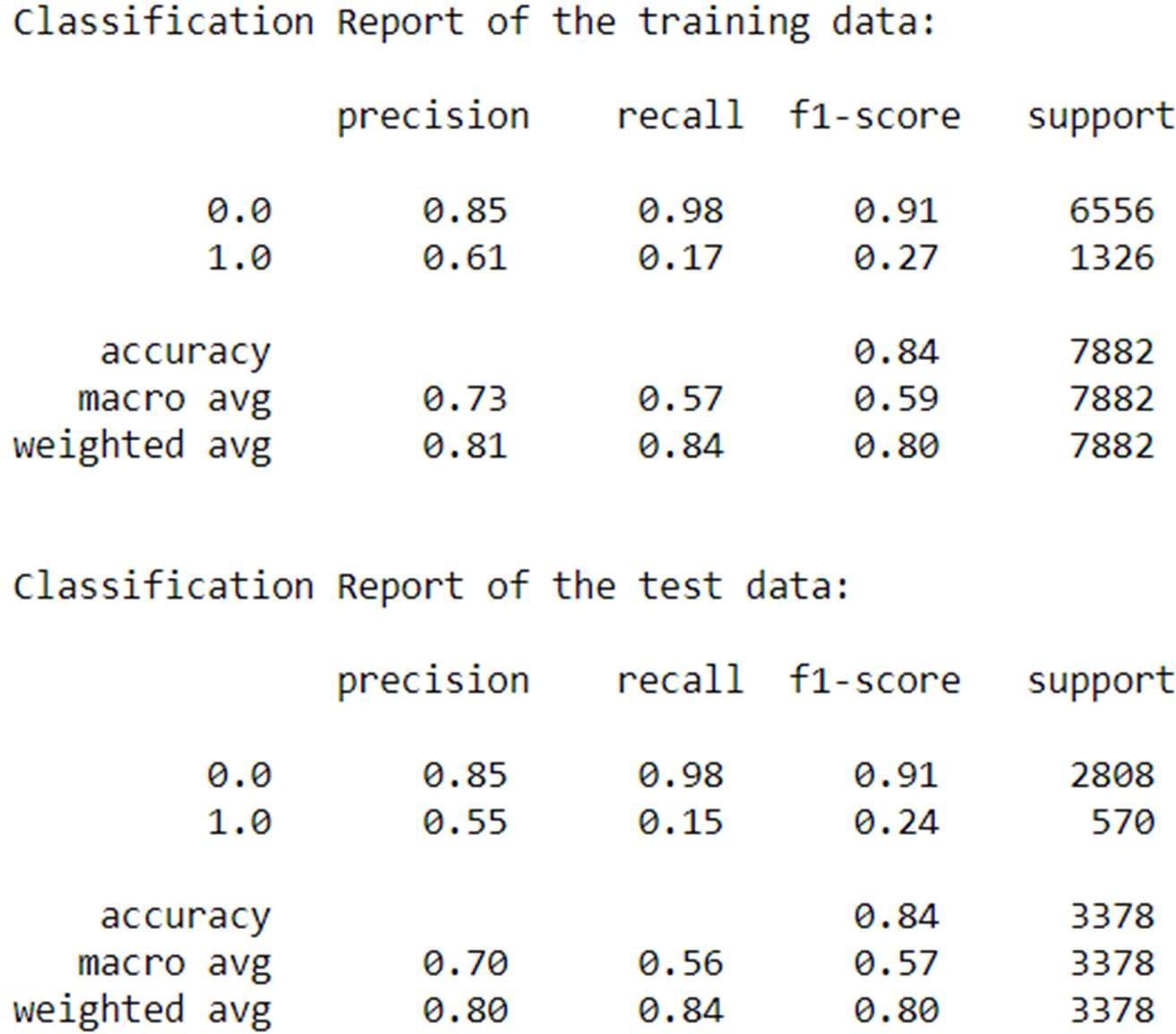
***Fig 16: - Accuracy from LDA***

Below is the confusion matrix obtained from this model: -



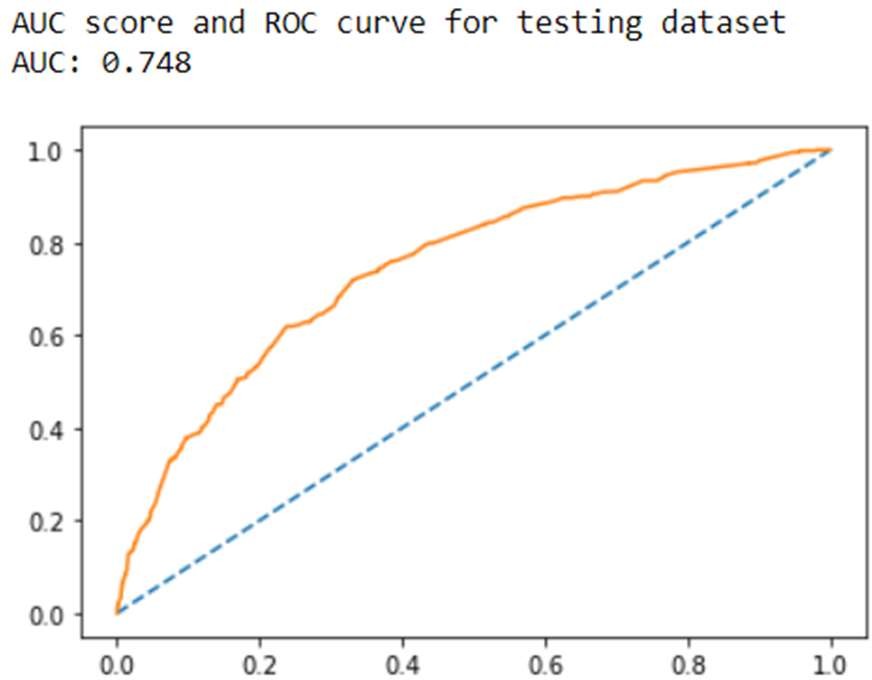
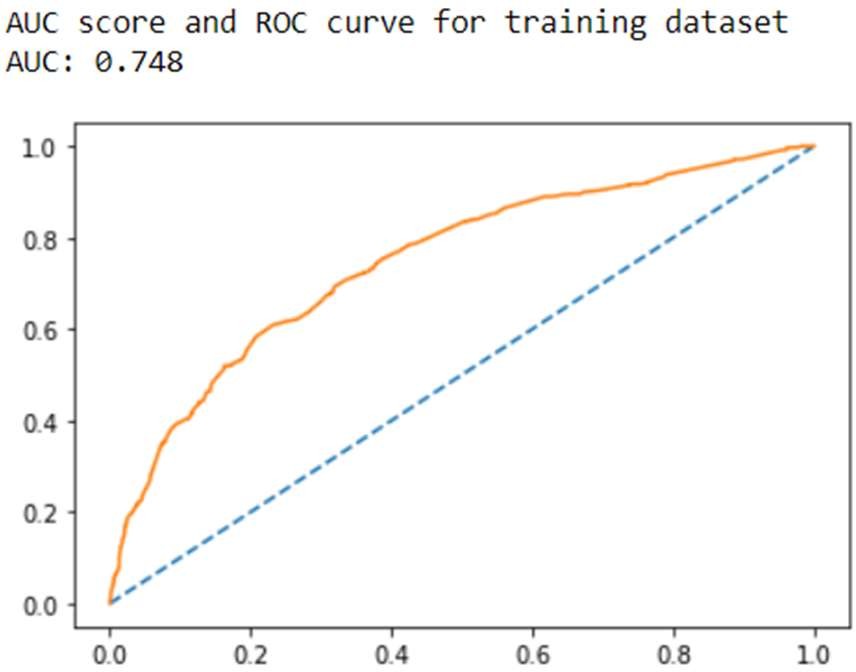
***Fig 17: - Confusion Matrix from LDA***

Below is the classification report obtained from this model: -



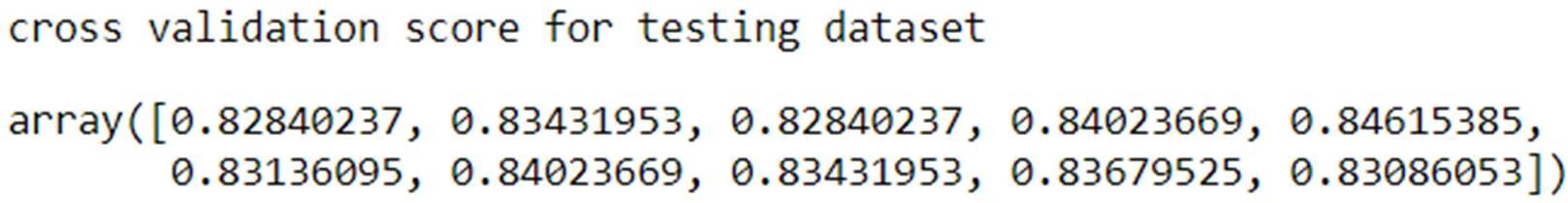
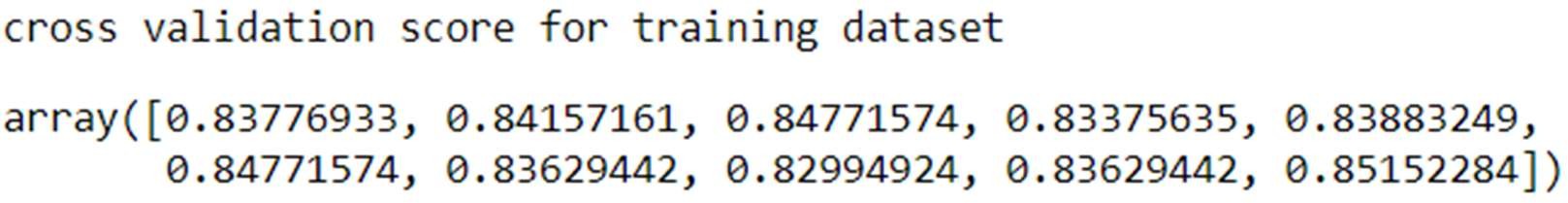
***Fig 18: - Classification Report from LDA***

Below are the AUC scores and ROC curves obtained from this model: -



***Fig 19: - ROC Curve and AUC Score From LDA***

Below are the 10-fold cross validation scores: -



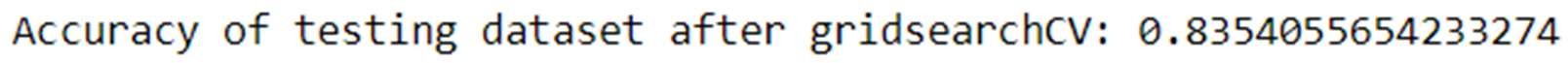
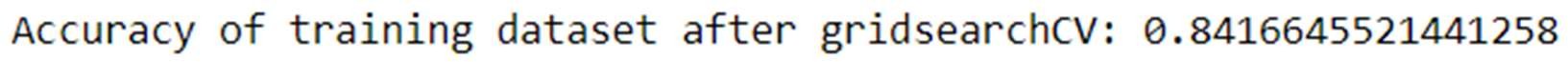
***Fig 20: - Cross Validation Score from LDA***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

### Building model using GridSearchCV and analyzing the best parameters: -

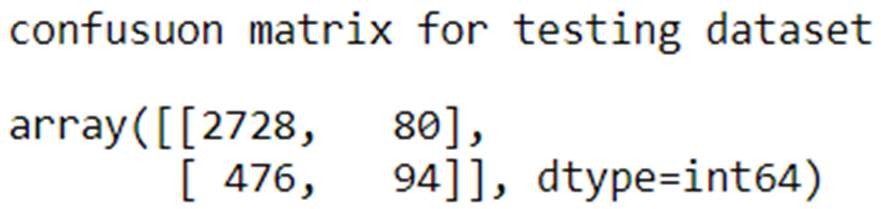
Using GridSearchCV function we tried finding the best parameters to further tune-in the above model for better accuracy and we have find that shrinkage as “auto”, solver as “lsqr” and tol value of “0.001” gives the best model considering accuracy, precision, recall, F1, ROC curve and AUC score.

Below are the accuracy scores obtained from model built using GridSearchCV: -



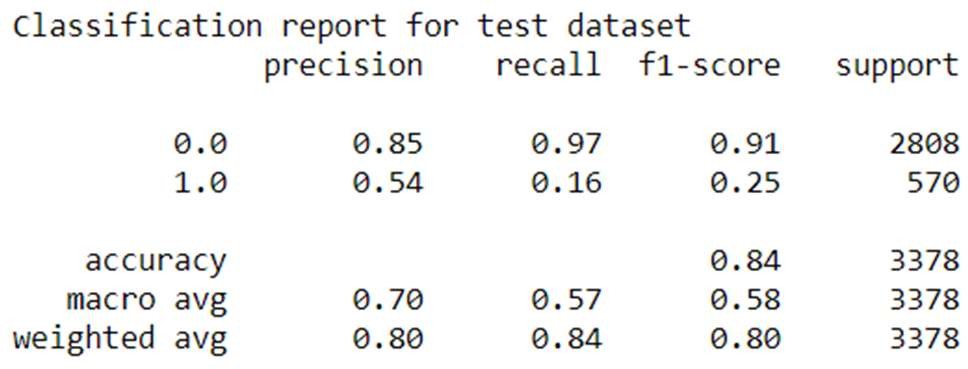
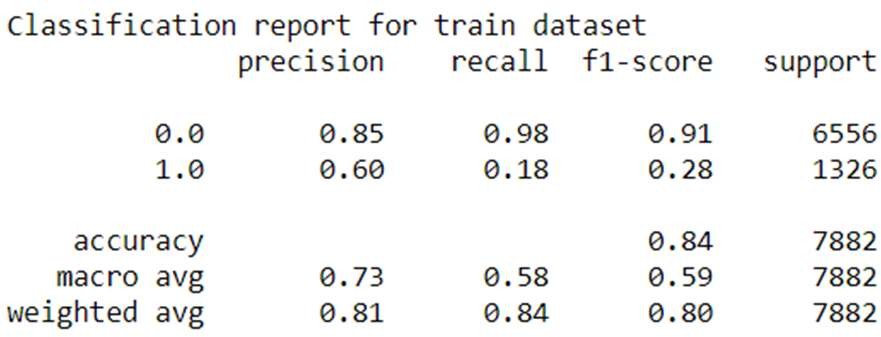
***Fig 21: - Accuracy Score from LDA with Hypertuning***

Below is the confusion matrix obtained from model built using GridSearchCV: -



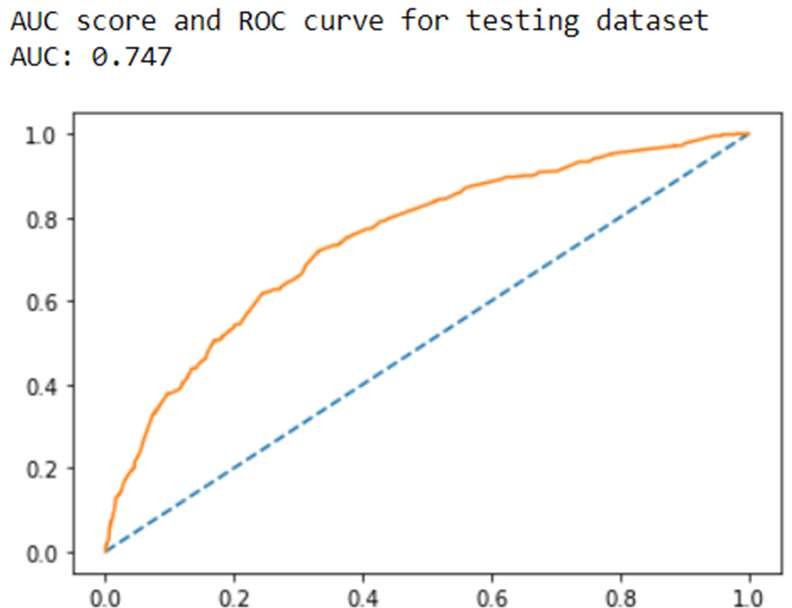
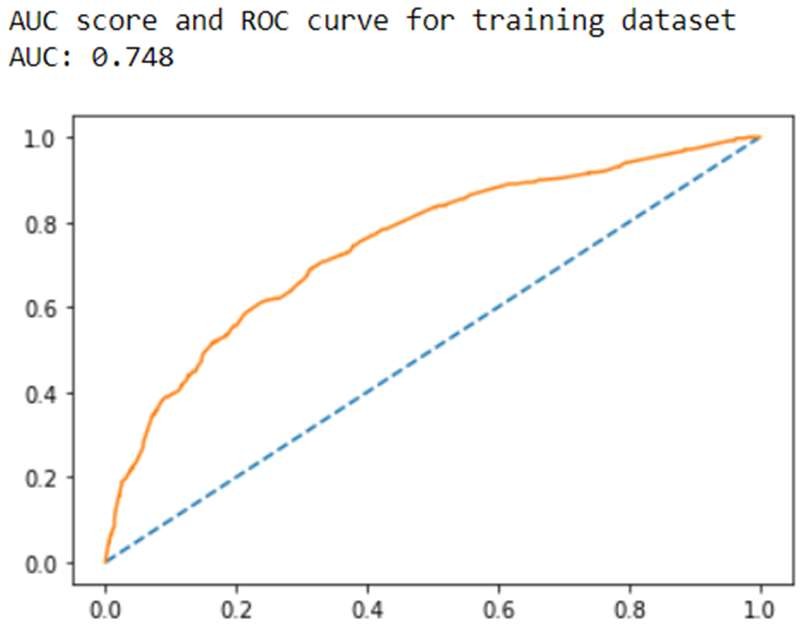
***Fig 22: - Confusion Matrix from LDA with Hypertuning***

Below are the classification reports obtained from model built using GridSearchCV: -



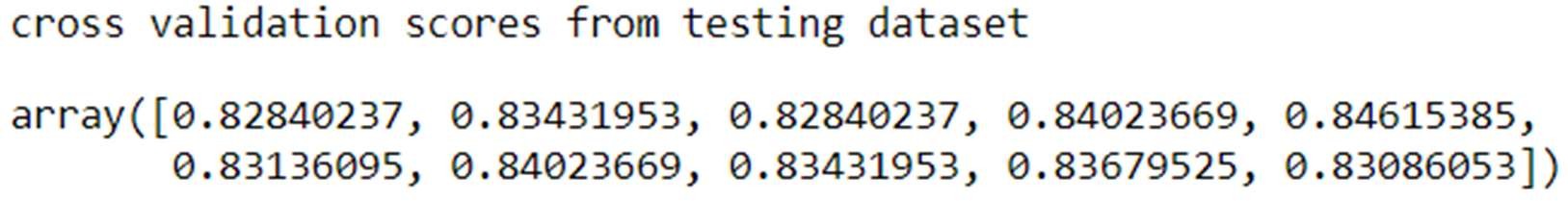
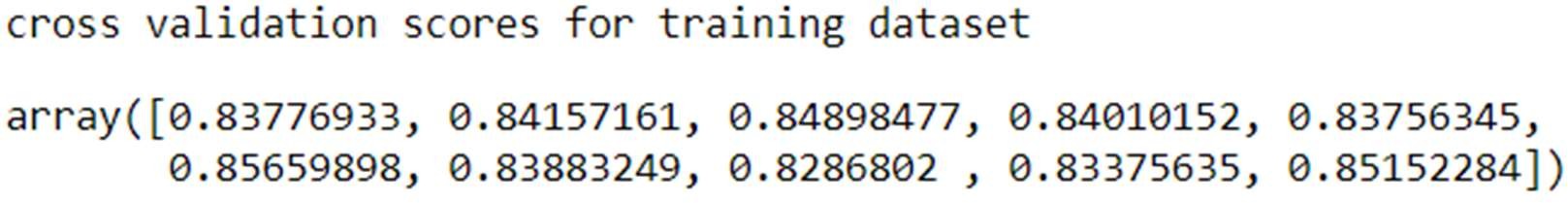
***Fig 23: - Classification Report from LDA with Hypertuning***

Below are the ROC curve and AUC scores obtained from model built using GridSearchCV: -



***Fig 24: - ROC Curve and AUC Score From LDA with Hypertuning***

Below are the 10-fold cross validation scores: -



***Fig 71: - Cross Validation Scores From LDA with Hypertuning***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

**Building LDA model using SMOTE: -**

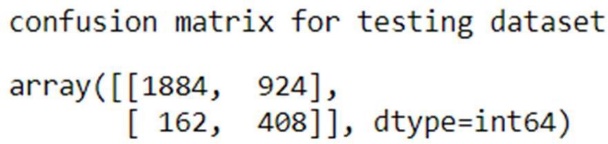
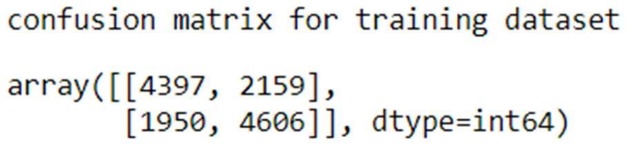
From above descriptive analysis we can conclude that the original data provided is imbalance in nature and by using SMOTE technique we can balance the data to check if the model can outperform when data is balanced. We have applied SMOTE technique to oversample the data and to obtain a balanced dataset.

Below are the accuracy scores obtained from balanced dataset: -



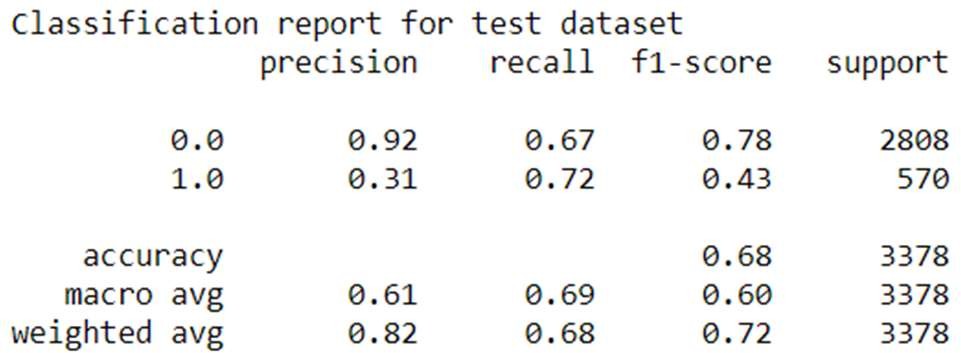
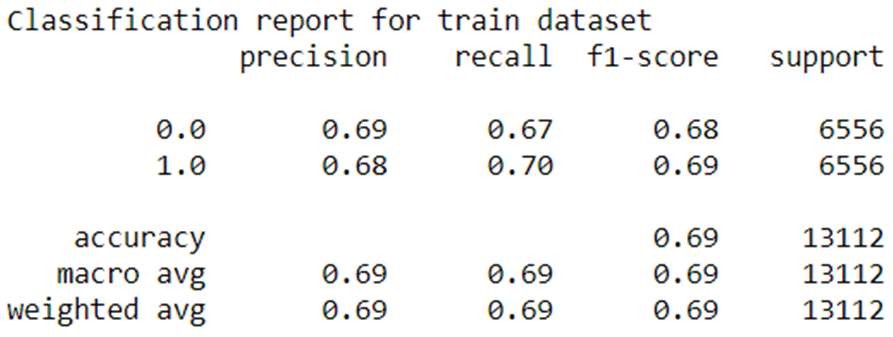
***Fig 25: - Accuracy from LDA with SMOTE***

Below is the confusion matrix obtained from balanced dataset: -



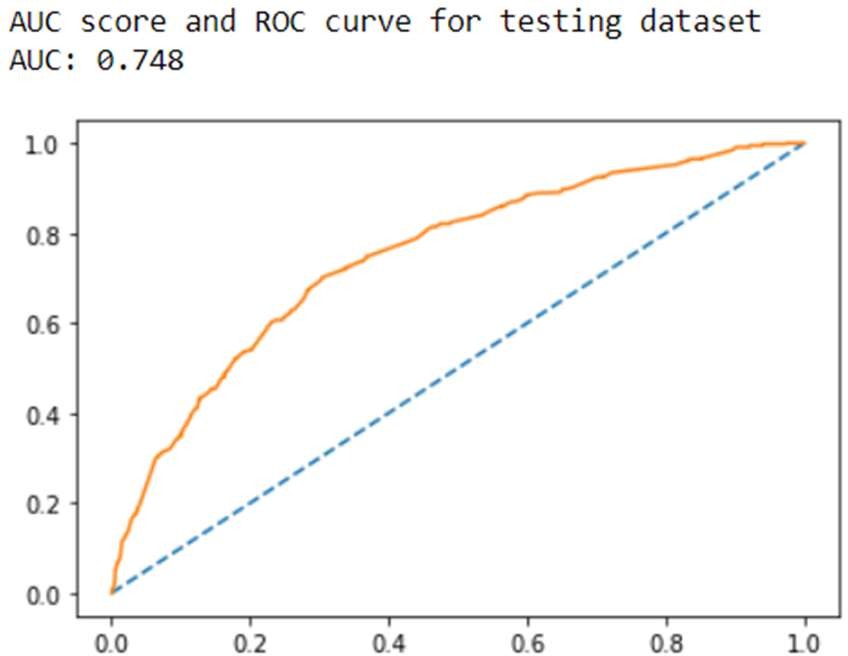
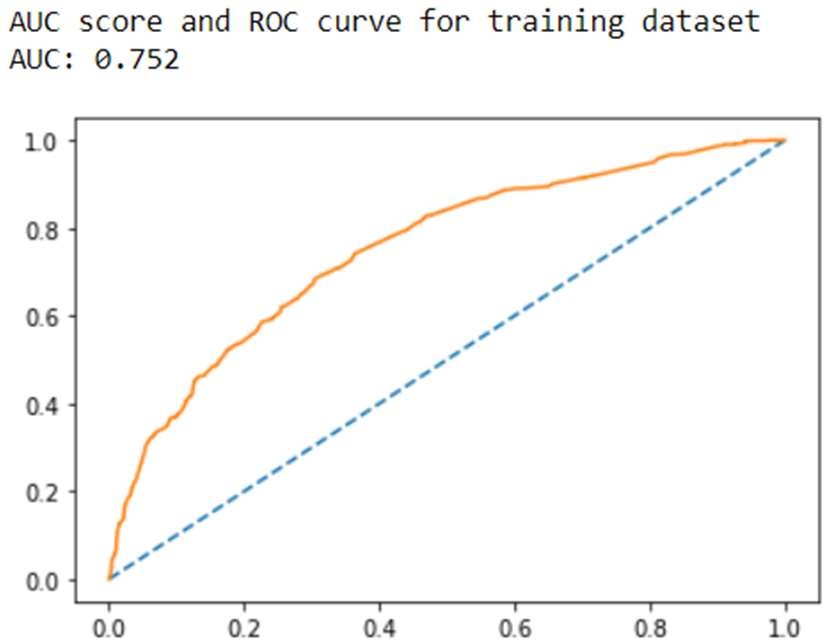
***Fig 26: - Confusion Matrix from LDA with SMOTE***

Below is the classification report obtained from balanced dataset: -



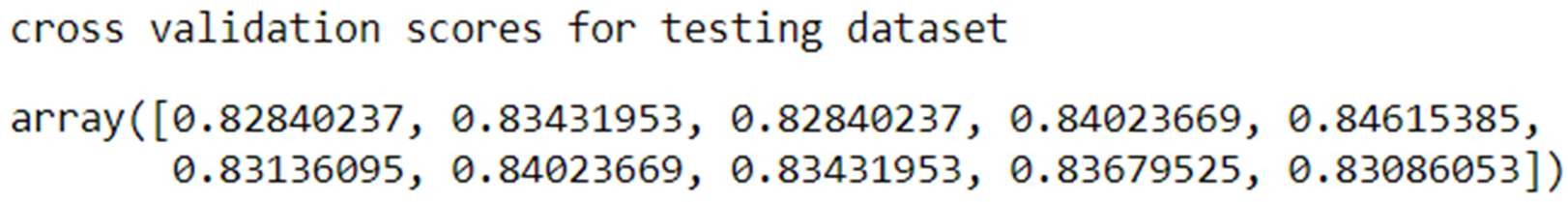
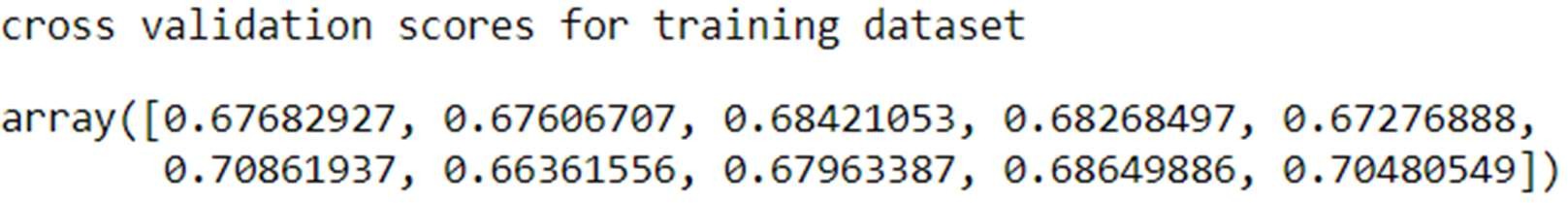
***Fig 27: - Classification Report from LDA with SMOTE***

Below are the ROC curve and AUC scores obtained from balanced dataset: -



***Fig 28: - ROC Curve and AUC Score From LDA with SMOTE***

Below are the 10-fold cross validation scores: -



***Fig 29: - Cross Validation Scores From LDA with SMOTE***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

**Inference/Conclusion from LDA Model: -**

From the above we can conclude that the data is neither “Overfit” nor “Underfit” in nature. And we can also inference that the model built using the parameters from GridSearchCV is best optimized. However, the accuracy scores along with recall, precision, F1 values, ROC curve and AUC score are not that significant as compared with models built with default values and model built using GridSearchCV. Model built on balance data set using SMOTE technique works well in training dataset however we can see a significant deplete in accuracy when it comes to testing dataset.

**BUILDING KNN MODEL: -**

Building model with default hyperparameters: -

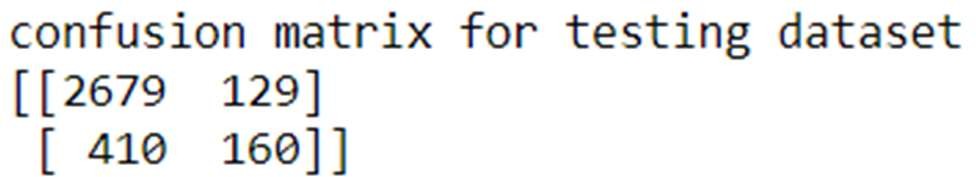
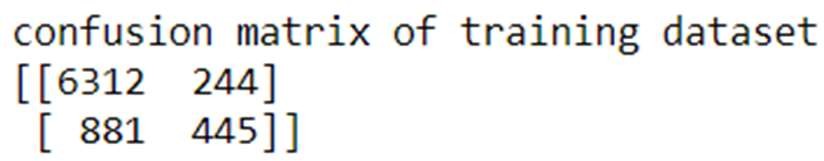
Post splitting data into training and testing data set we fitted KNN model into training dataset and performed prediction on training and testing dataset using the same model. We made the first model with default hyperparameters with default value of n\_neighbour as “5”.

Below are the accuracy scores obtained from this model: -



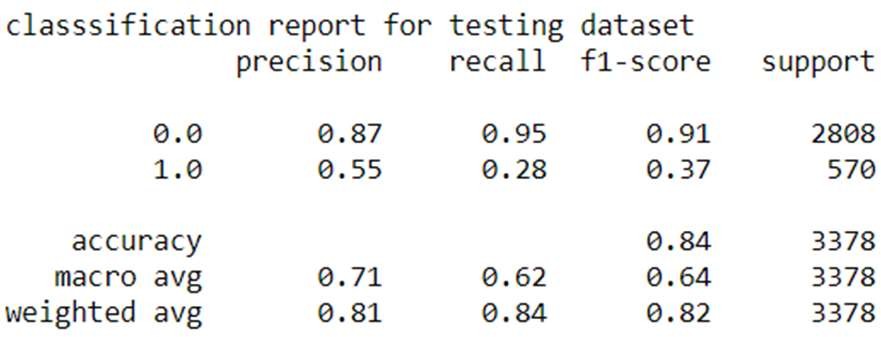
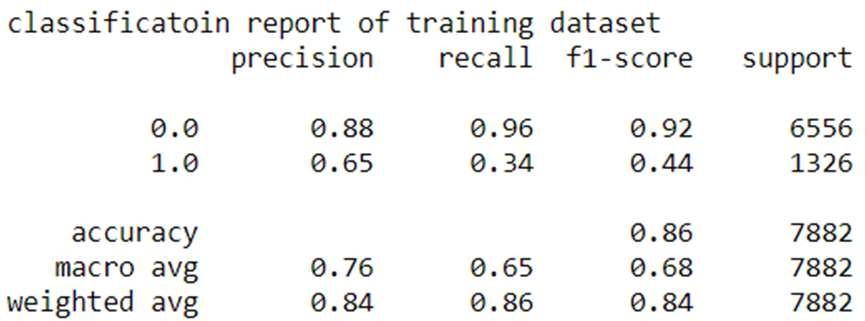
***Fig 30: - Accuracy Scores From KNN***

Below are the confusion matrices obtained from this model: -



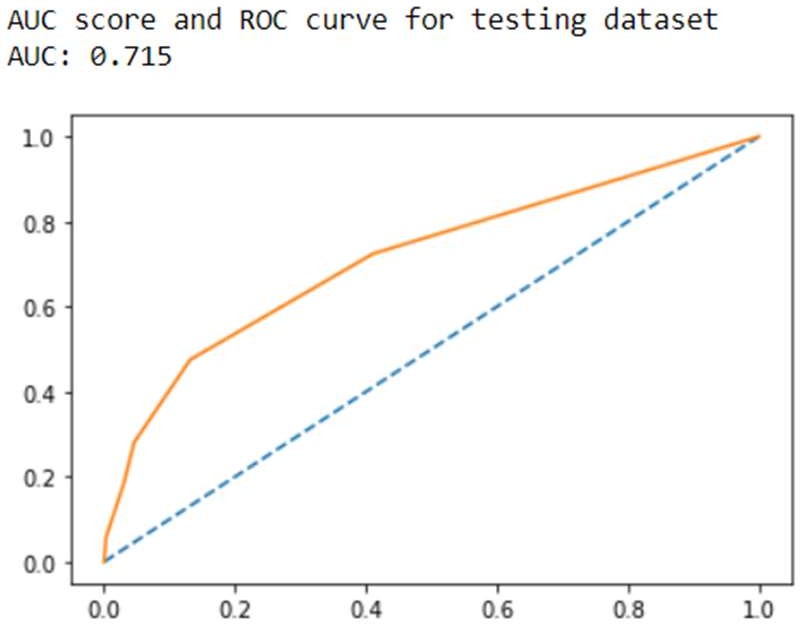
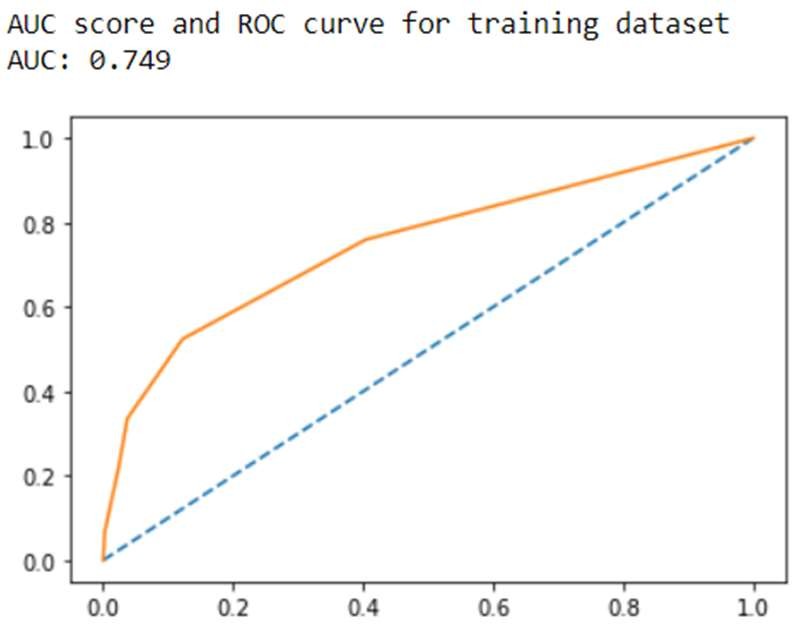
***Fig 31: - Confusion Matrix from KNN***

Below are the classification Report obtained from this model: -



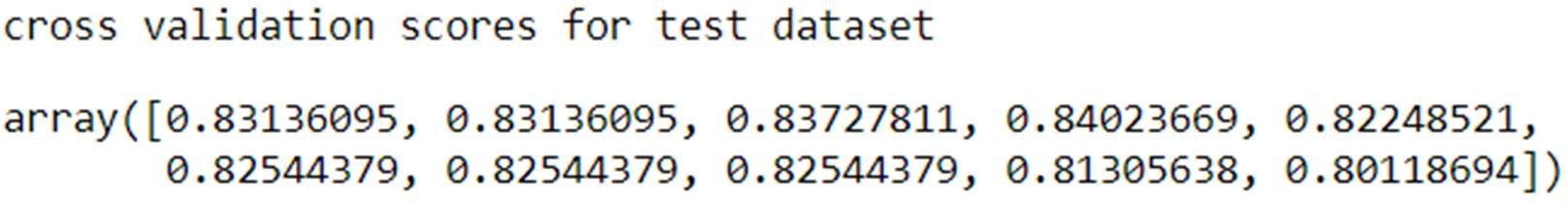
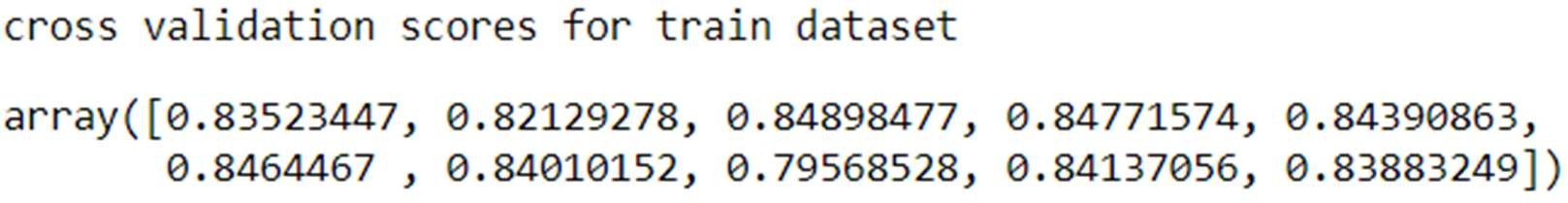
***Fig 32: - Classification Report from KNN***

Below are the AUC scores and ROC curves obtained from this model: -



***Fig 33: - ROC Curve and AUC Scores From KNN***

Below are the 10-fold cross validation scores: -



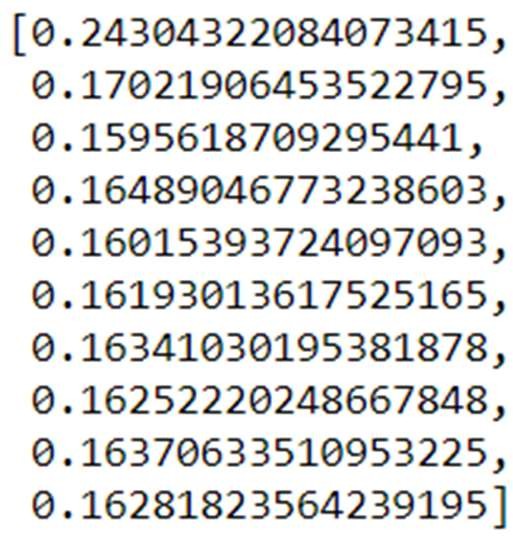
***Fig 34: - Cross Validation Scores From KNN***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

## Find the right value of n\_neighbor: -

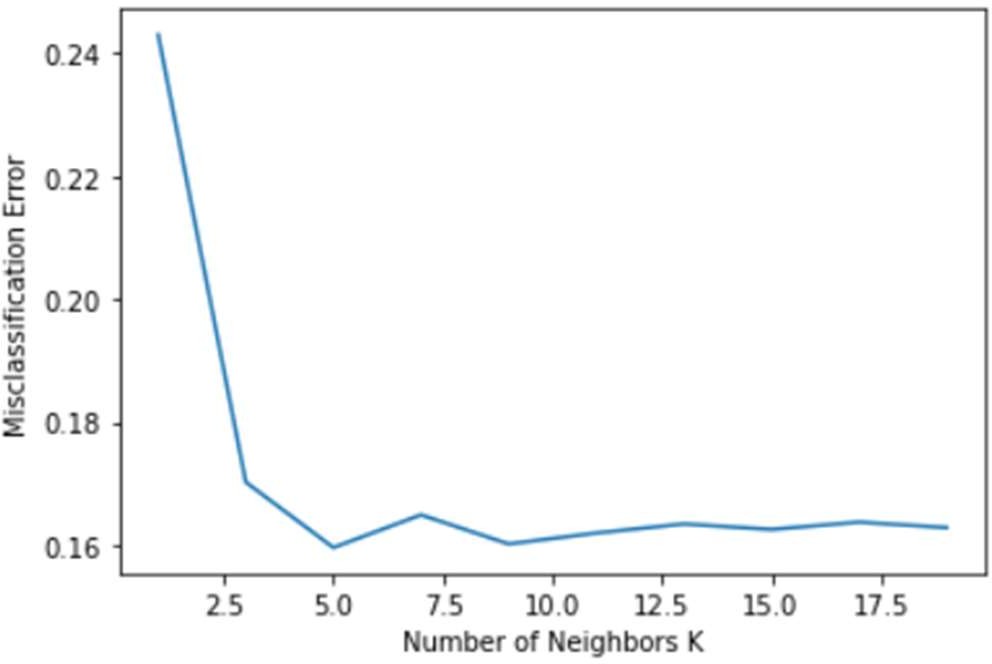
It’s very important to have the right value of n\_neighbors to fetch the best accuracy from the model. We can decide on the best value for n\_neighbors based on MSE (mean squared error) scores. The value with least score of MSE indicated least error and will fetch the best optimized n\_neighbors value.

**Below are the MSE scores: -**



***Fig 35: - MSE Scores***

Below is the graphical version of MSE scores across numerous values of n\_neighbors.



***Fig 36: - Graphical Version of MSE Score***

From the above plotted graph we can see the n\_neighbors with value “5” gives the least MSE score. With which we can proceed and build KNN model with n\_neighbor value as “5” which is also the default n\_neighbor. **Hence, different model building with correct number of n\_neighbor is not required as it’s the same as default value if n\_neighbor.**

**Building model using GridSearchCV and getting the best hyperparameters: -**

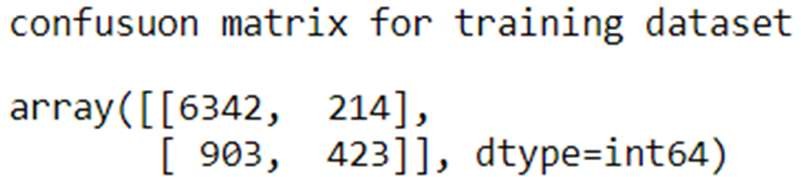
After building the model with its default values as shown above, we will try and find the best hyper parameters to check if we can outperform the accuracy achieved by the model built with default values of hyperparameter. From GridSearchCV we found that the best parameters are “ball-tree” as algorithm, “Manhattan” as metrics, “5” as n\_neighbors and “distance” as weights.

Below are the accuracy scores obtained from this model using GridSearchCV: -



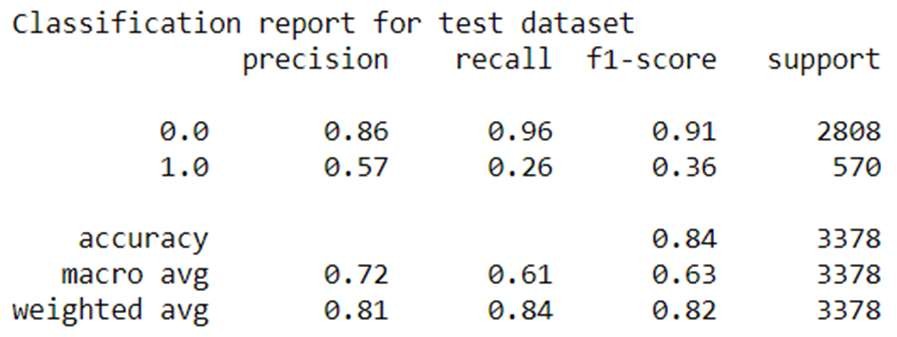
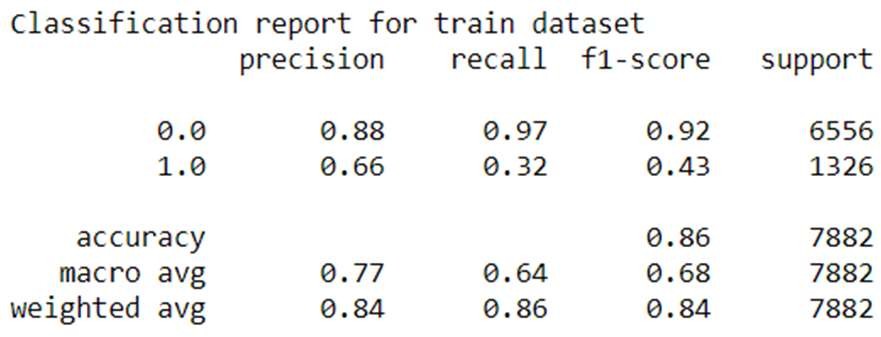
***Fig 37: - Accuracy from KNN with Hyperparamter Tuning***

Below are the confusion matrices obtained from this model using GridSearchCV: -



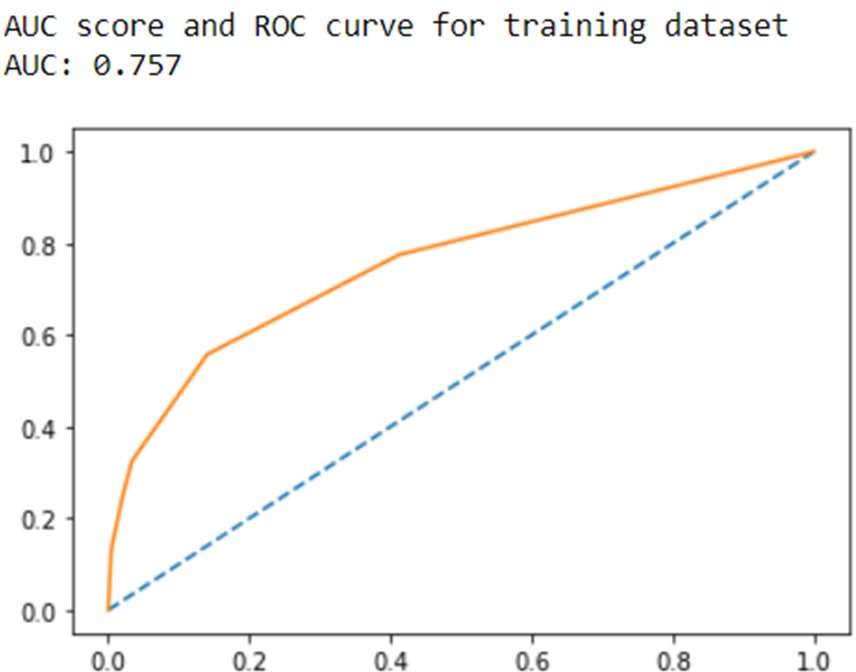
***Fig 38: - Confusion Matrix from KNN with Hyperparamter Tuning***

Below is the classification report obtained from this model using GridSearchCV: -



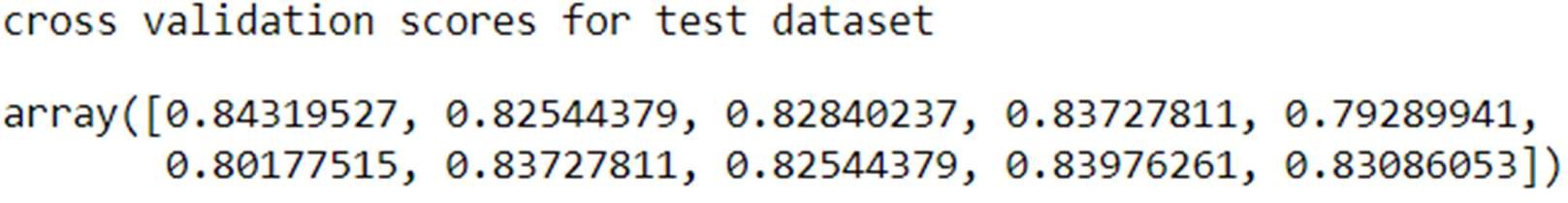
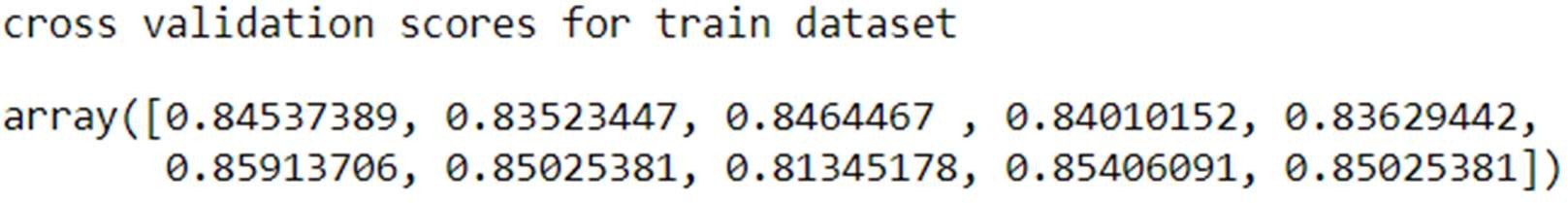
***Fig 39: - Classification Report from KNN with Hyperparamter Tuning***

Below are the AUC scores and ROC curves obtained from this model using GridSearchCV: -



***Fig 40: - ROC Curve and AUC Score From KNN with Hyperparamter Tuning***

Below are the 10-fold cross validation scores: -



***Fig 41: Cross Validation Scores from KNN with Hyperparamter Tuning***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

**Building model using SMOTE: -**

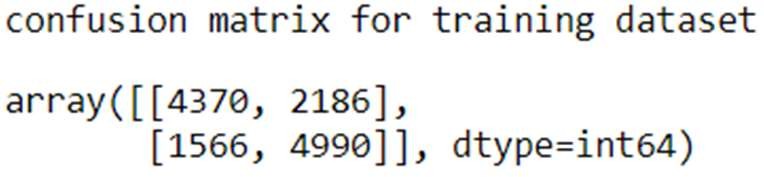
From above descriptive analysis we can conclude that the original data provided is imbalance in nature and by using SMOTE technique we can balance the data to check if the model can outperform when data is balanced. We have applied SMOTE technique to oversample the data and to obtain a balanced dataset.

Below are the accuracy scores obtained from balanced dataset: -



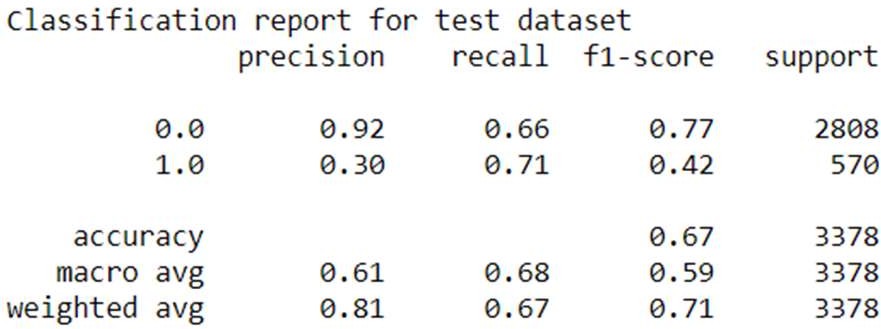
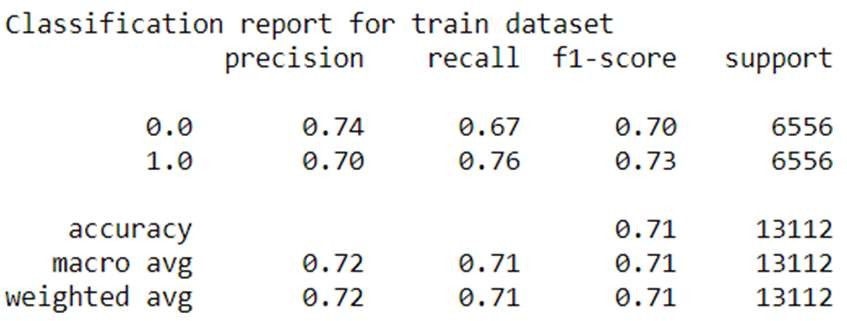
***Fig 42: Accuracy Score from KNN with SMOTE***

Below are the confusion matrices obtained from balanced dataset: -



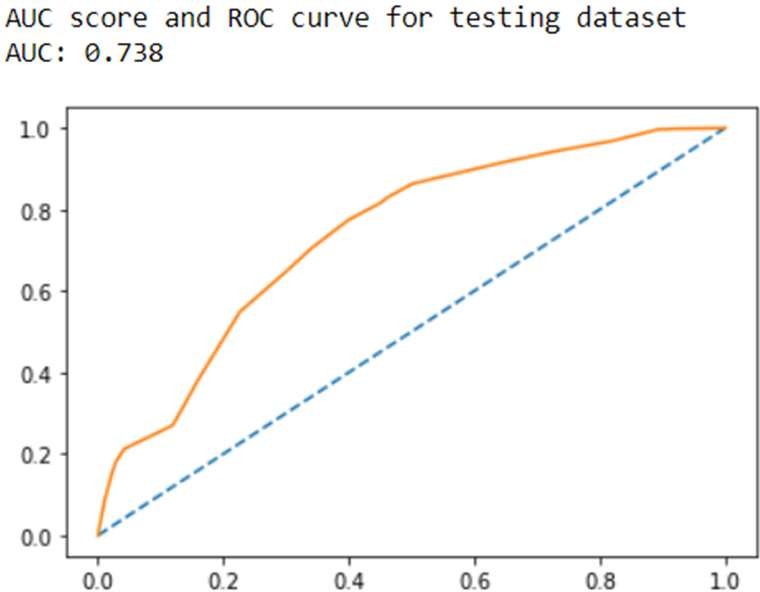
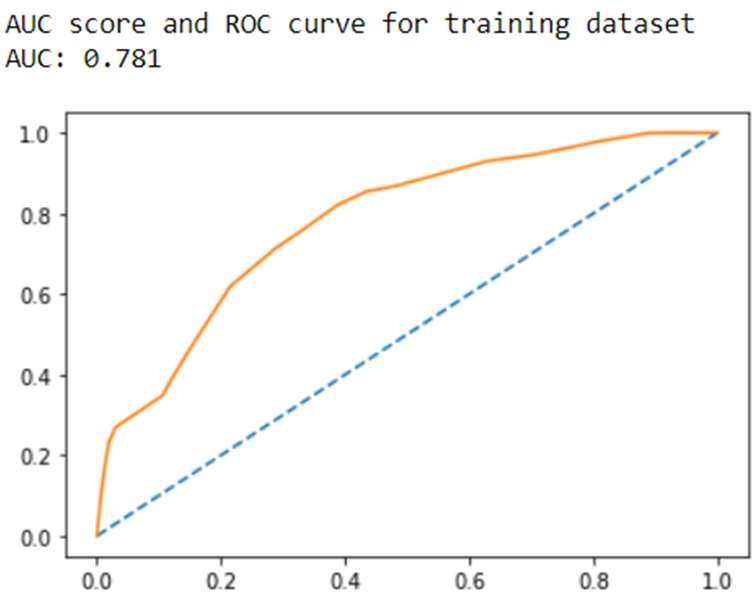
***Fig 43: Confusion Matrix from KNN with SMOTE***

Below are the classification reports obtained from balanced dataset: -



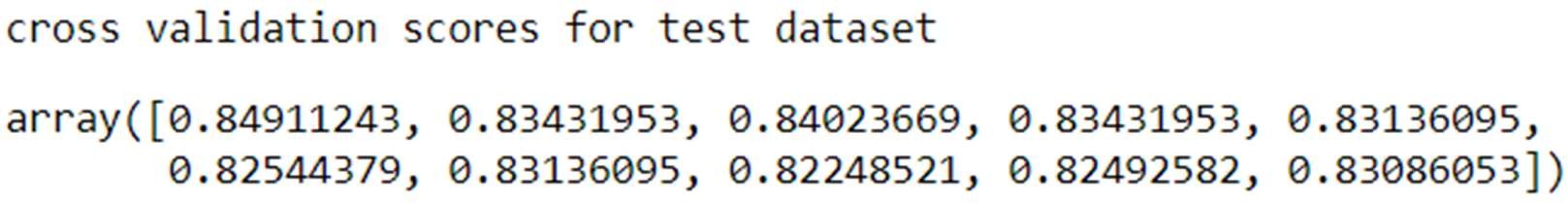
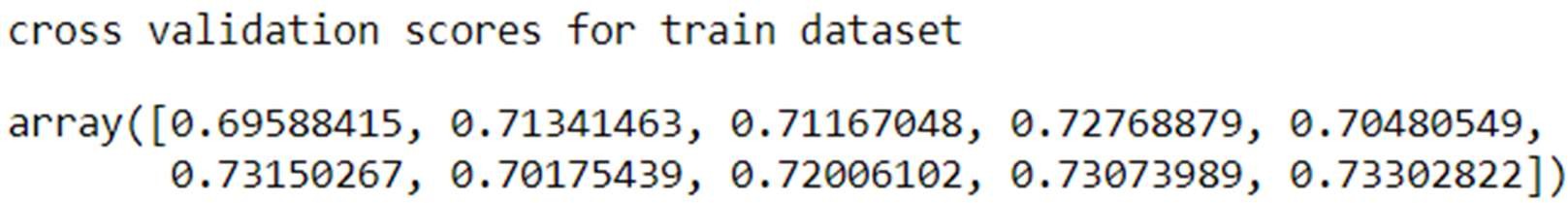
***Fig 44: Classification Reports from KNN with SMOTE***

Below are the AUC scores and ROC curves obtained from balanced dataset: -



***Fig 45: ROC Curve and AUC Scores from KNN with SMOTE***

Below are the 10-fold cross validation scores: -



***Fig 46: Cross Validation Scores From KNN with SMOTE***

We can observe that the cross validations scores are almost same for all the folds. Which indicates that the model built is correct.

**Inference/Conclusion from KNN Model: -**

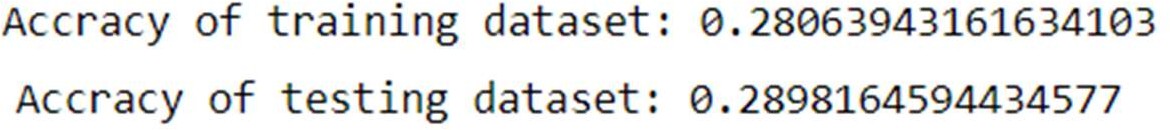
From the above we can conclude that the data is neither “Overfit” nor “Underfit” in nature. And we can also inference that the model built using grid search CV is best optimized model for prediction. However, we can see significant variations in accuracy score, F1 score, recall values, precision values, ROC curves and AUC scores when compared with default values of KNN and also with model built on balanced dataset. Model built on balance data set using SMOTE technique works well in training dataset however we can see a significant deplete in accuracy when it comes to testing dataset.

**BUILDING NAÏVE BAYES MODEL: -**

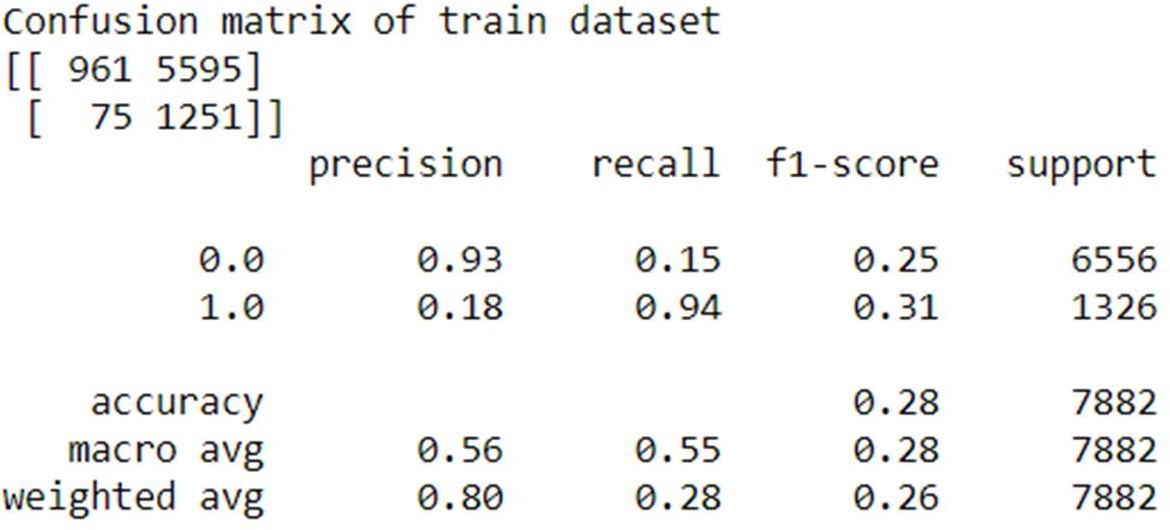
Post splitting data into training and testing we are now ready to build model using Naïve Bayes algorithm. Naïve Bayes algorithm is based out of Bayes theorem of conditional probability.

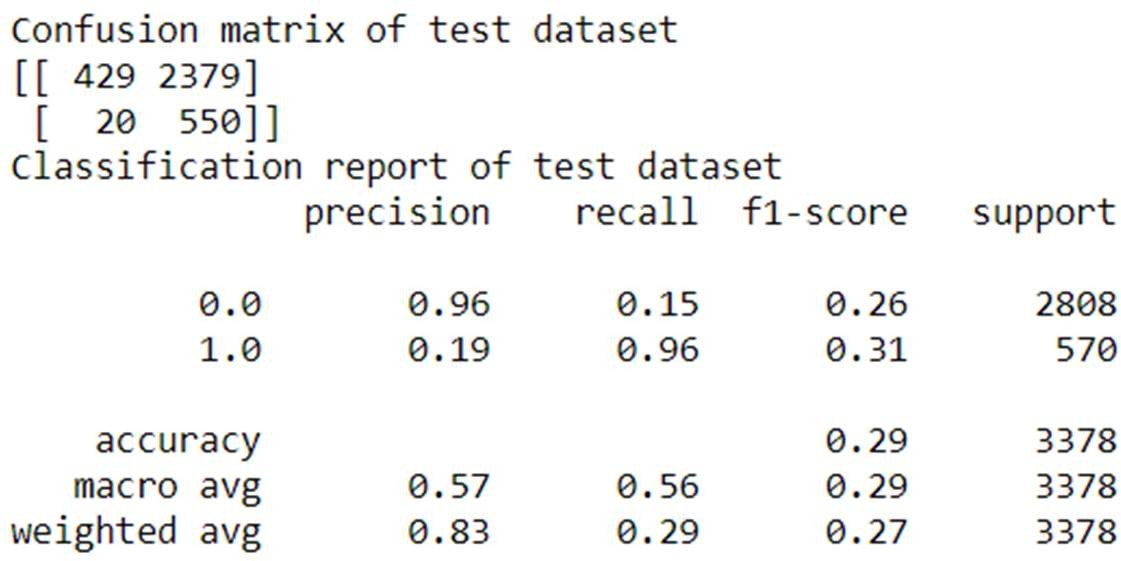
Considering that all events are independent of each other and then we find the probability of an event happening under the condition that one of the events has already occurred.

Below are accuracy scores using this model: -



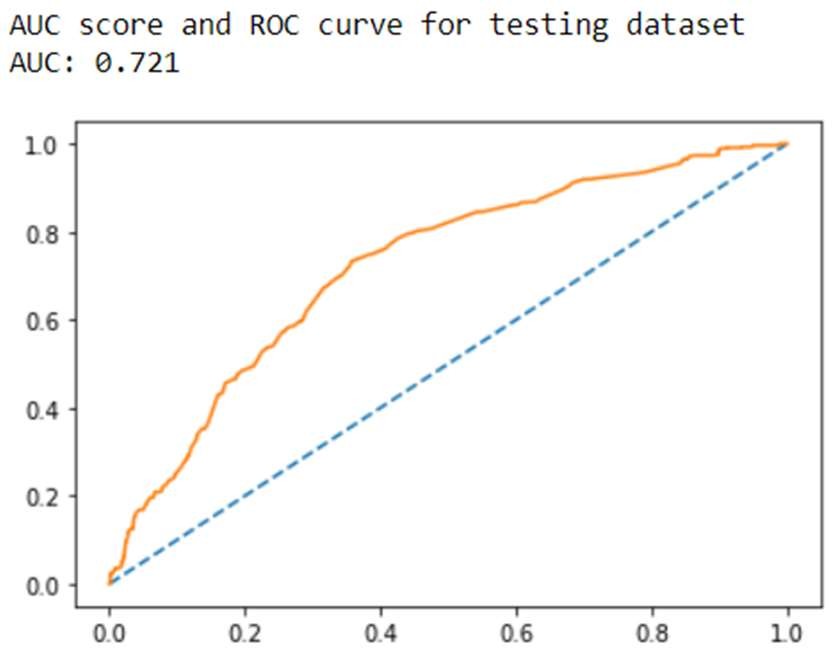
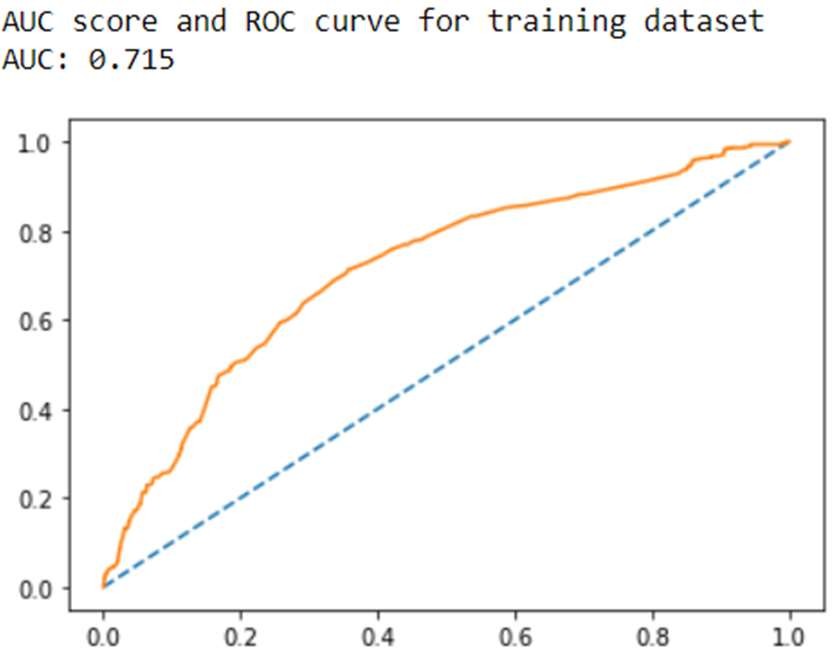
***Fig 47: Training Scores from Naïve Bayes with SMOTE***

Below are confusion matrices and classification reports using this model: -



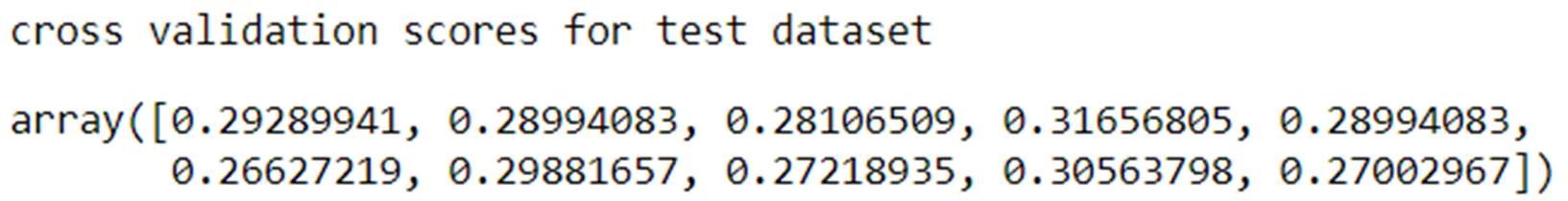
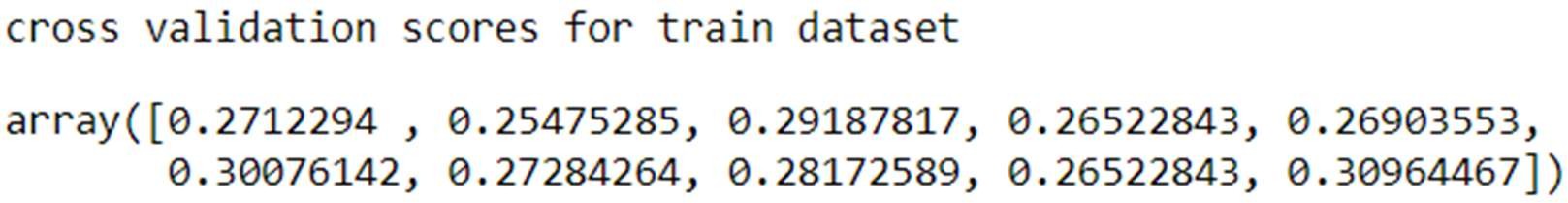
***Fig 48: Confusion Matrix and Classification Report from Naïve Bayes with SMOTE***

Below are AUC scores and ROC curves using this model: -



***Fig 49: ROC Curve and AUC Scores from Naïve Bayes with SMOTE***

Below are 10-fold cross validation scores using this model: -

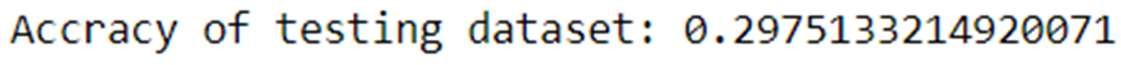


***Fig 50: Cross Validation Scores From Naïve Bayes with SMOTE***

**Building Gaussian Naive Bayes over balanced data using SMOTE**

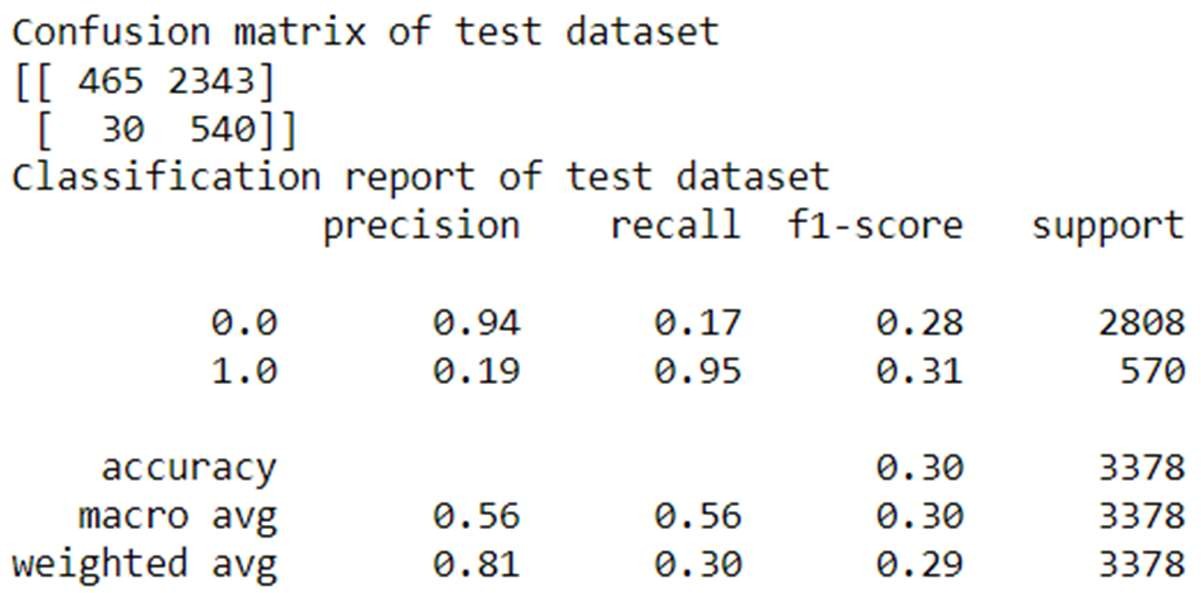
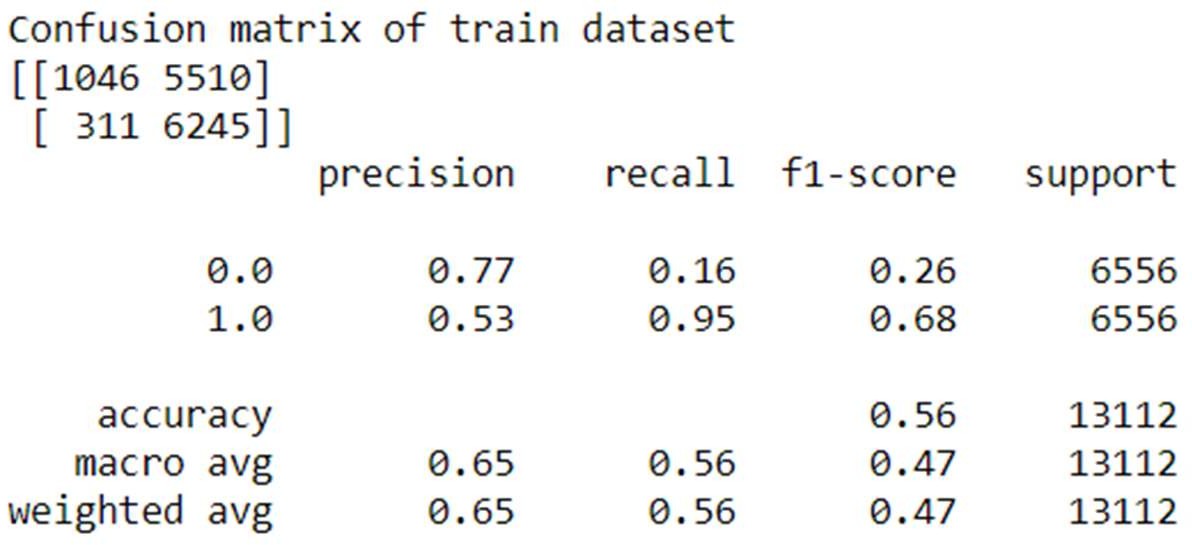
After building naïve Bayes model using original imbalanced data. Now, we can try and build the same model using balanced data to check if it outperforms in terms of accuracy and other measurement factors.

Below are the accuracy scores obtained using Naïve Bayes algorithm over balanced dataset: -



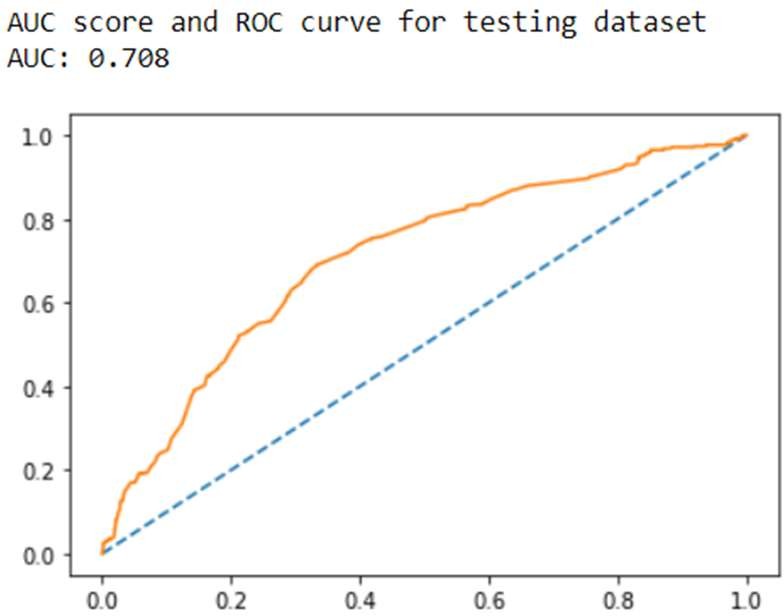
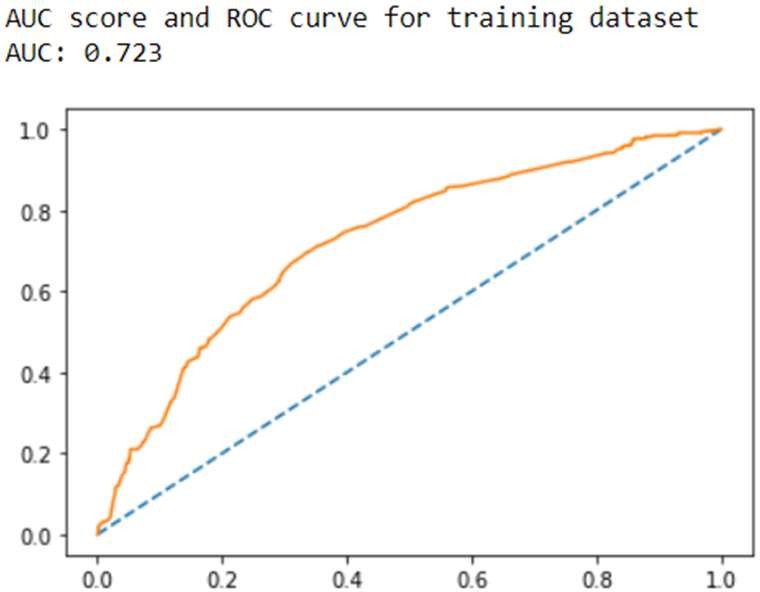
***Fig 51: Accuracy Scores from Naïve Bayes with SMOTE***

Below are the confusion matrices and classification reports obtained using Naïve Bayes algorithm over balanced dataset: -



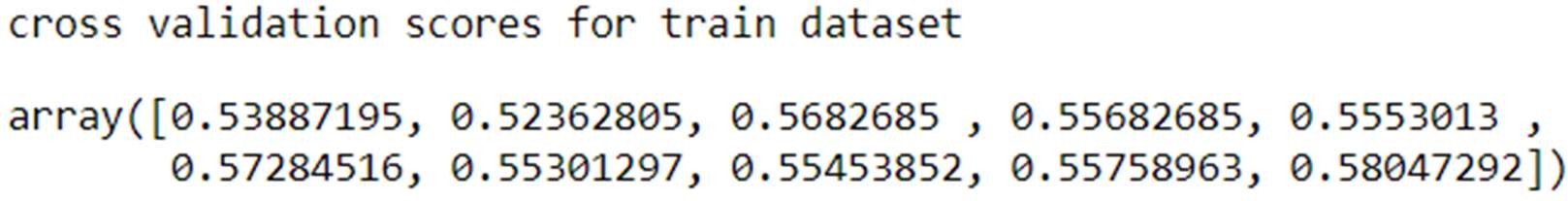
***Fig 52: Confusion Matrix and Classification Report From Naïve Bayes with SMOTE***

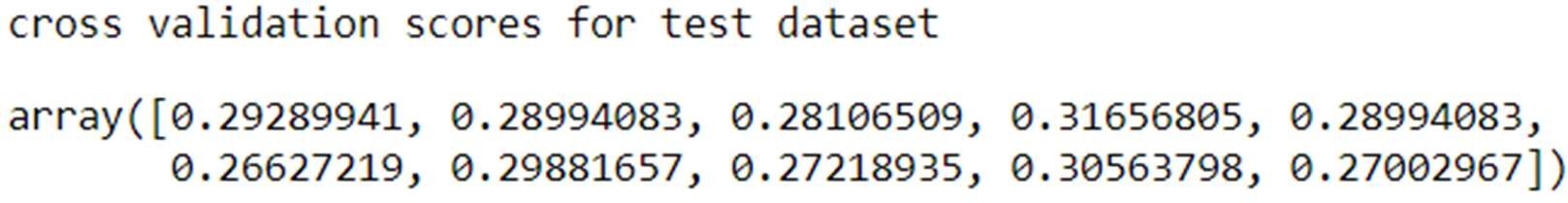
Below are the AUC scores and ROC curves obtained using Naïve Bayes algorithm over balanced dataset: -



***Fig 53: ROC Curve and AUC Scores from Naïve Bayes with SMOTE***

Below are the 10-fold cross validation scores obtained using Naïve Bayes algorithm over balanced dataset: -





***Fig 54: Cross Validation Scores From Naïve Bayes with SMOTE***

**Inference/Conclusion from Naïve Bayes Model: -**

From the above we can conclude that the data is neither “Overfit” nor “Underfit” in nature. And we can also inference that the model built using original imbalanced data is best optimized model for prediction. However, we can see significant variations in accuracy score, F1 score, recall values, precision values, ROC curves and AUC scores when compared with model built on balanced dataset. Model built on balance data set using SMOTE technique works well in training dataset however we can see a significant deplete in accuracy when it comes to training dataset.

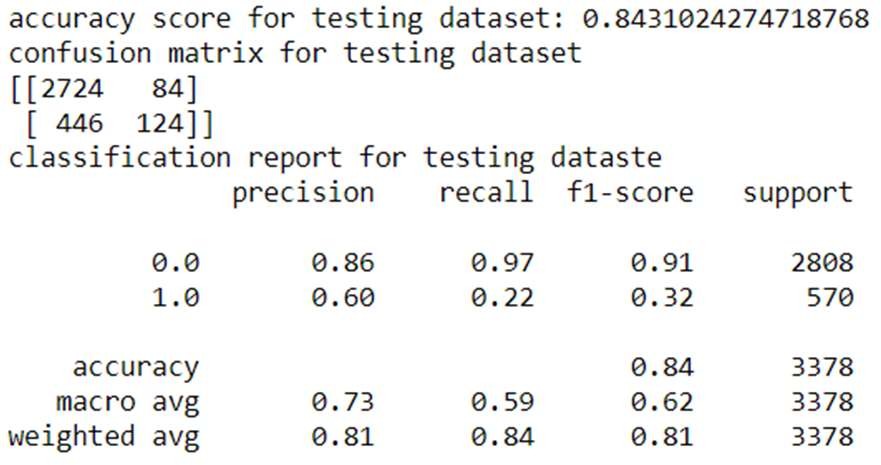
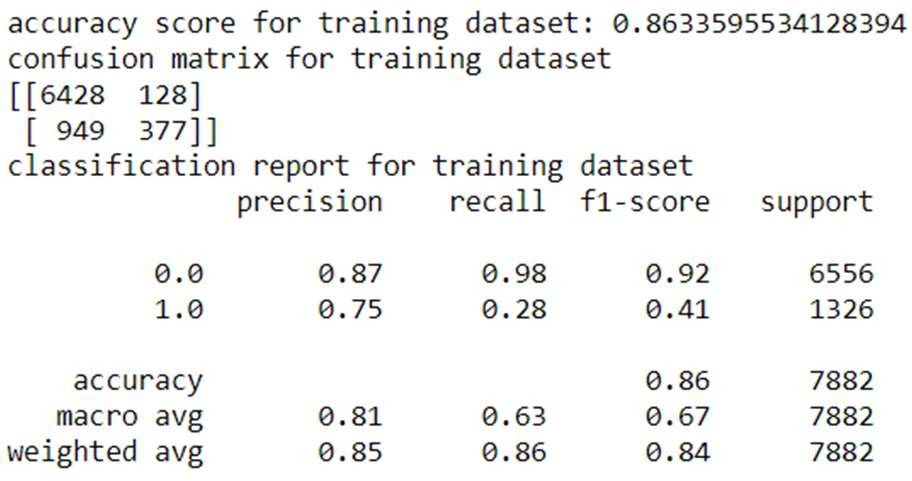
**BAGGING: -**

Building Random Forest model: -

Firstly, let’s build and random forest model for original dataset and balanced dataset and check the performance for the same.

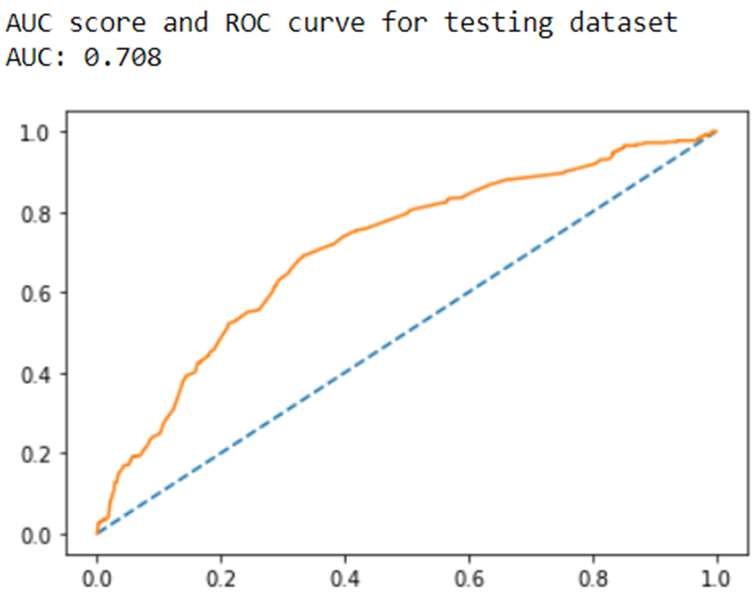
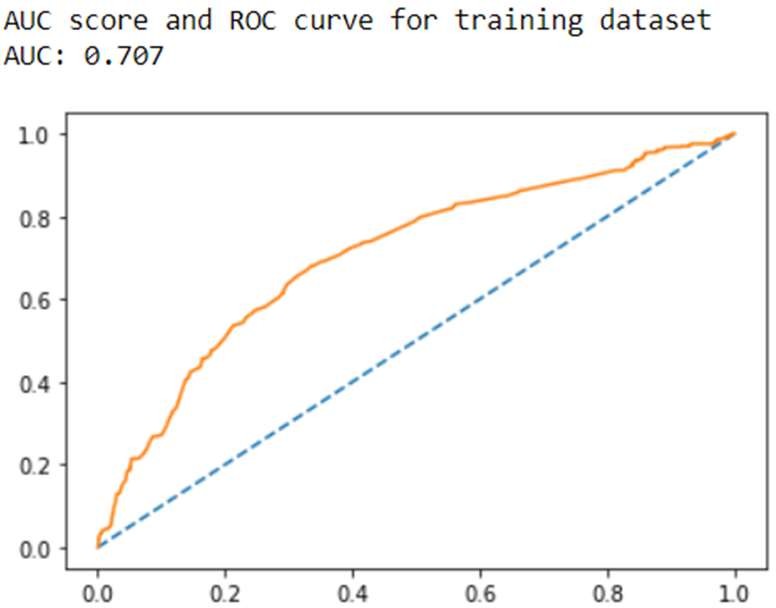
Random Forest Built in original dataset: -

Below are the accuracy scores, confusion matrix and classification report for training and testing dataset: -



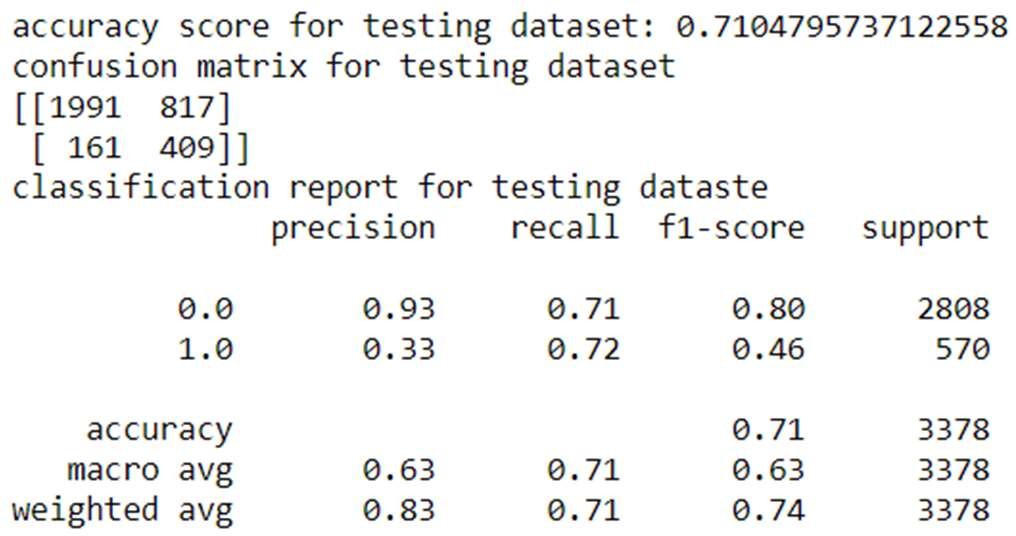
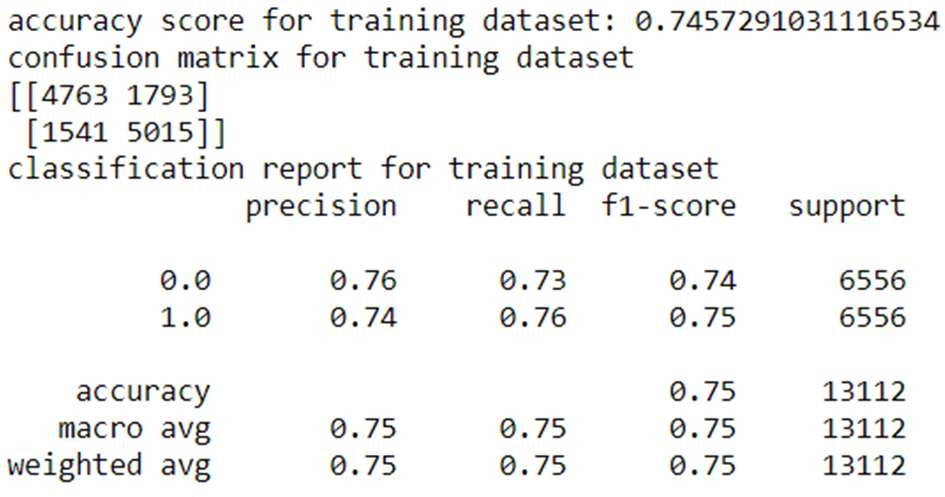
***Fig 55: Accuracy Parameters from Random Forrest***

Below are ROC Curve and AUC Score of train and test data set



***Fig 56: ROC Curve and AUC Scores from Random Forrest***

Random Forest Built in balanced dataset: -

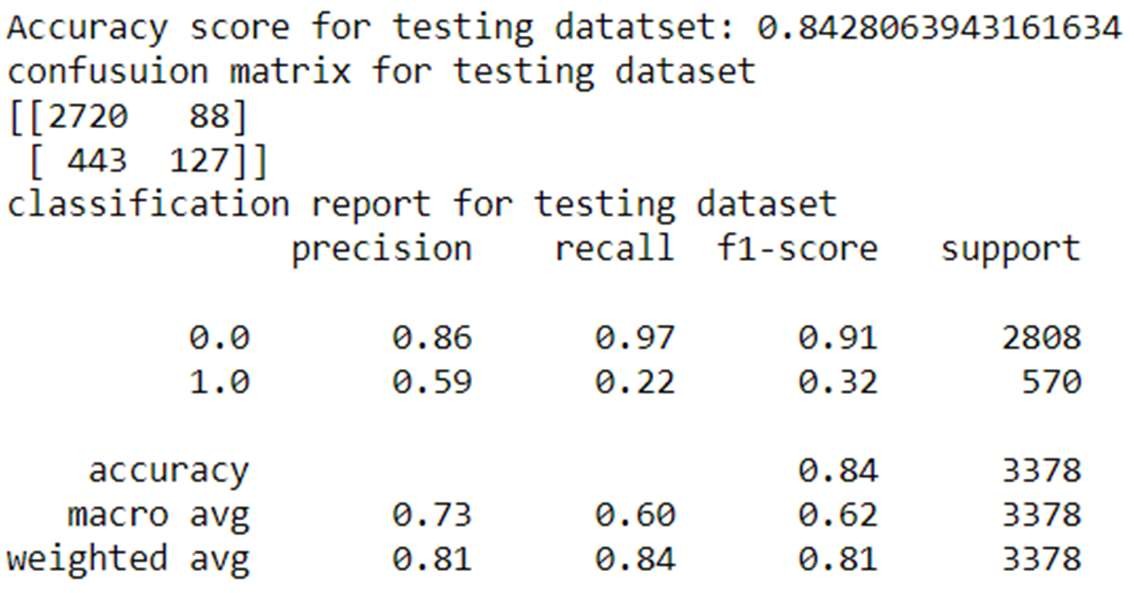
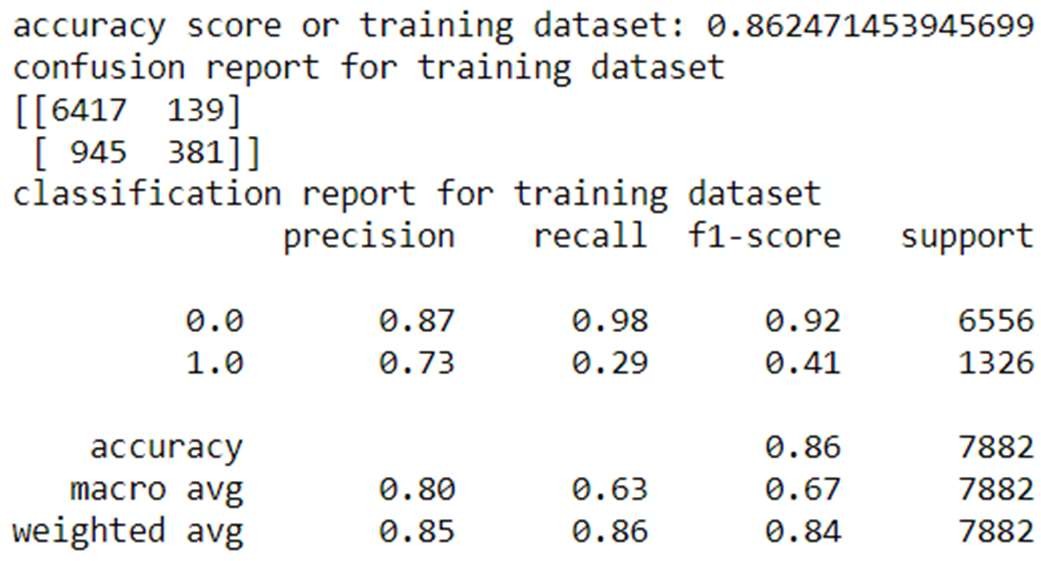
Below are the accuracy scores, confusion matrix and classification report for training and Tetsing dataset: -

***Fig 57: Accuracy Parameters from Random Forrest with SMOTE***

### Bagging on original dataset: -

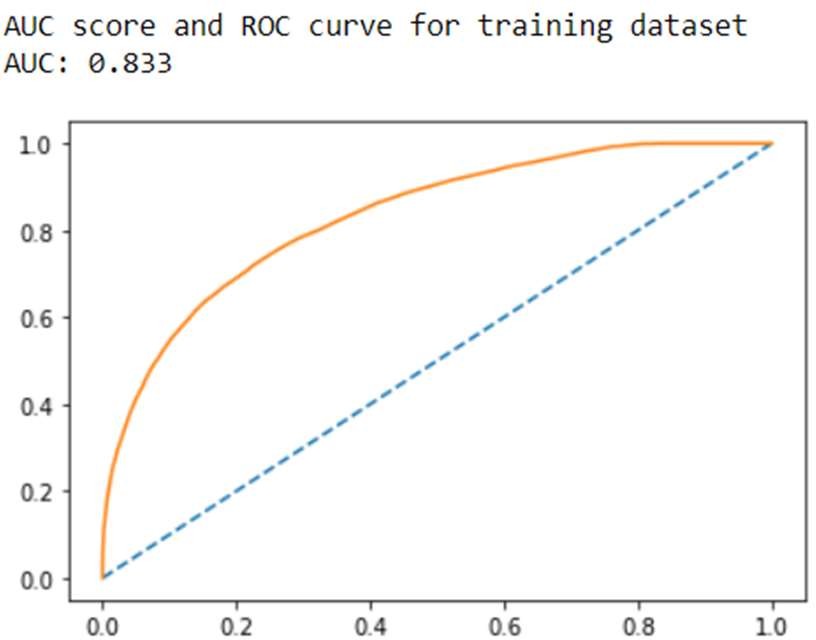
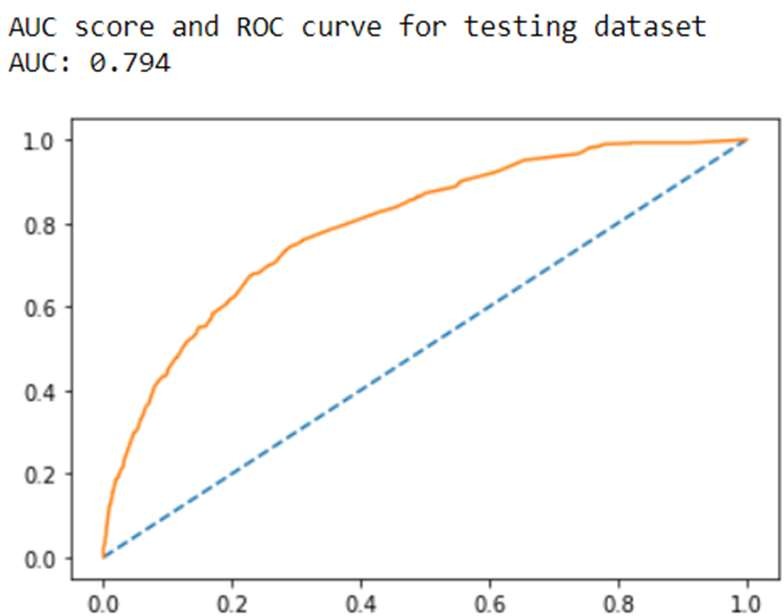
Let’s build bagging model using random forest and check if it can perform better than general random forest model.

Below are the accuracy scores, confusion matrix and classification report for training and Testing dataset: -



***Fig 58: Bagging On Original Dataset***

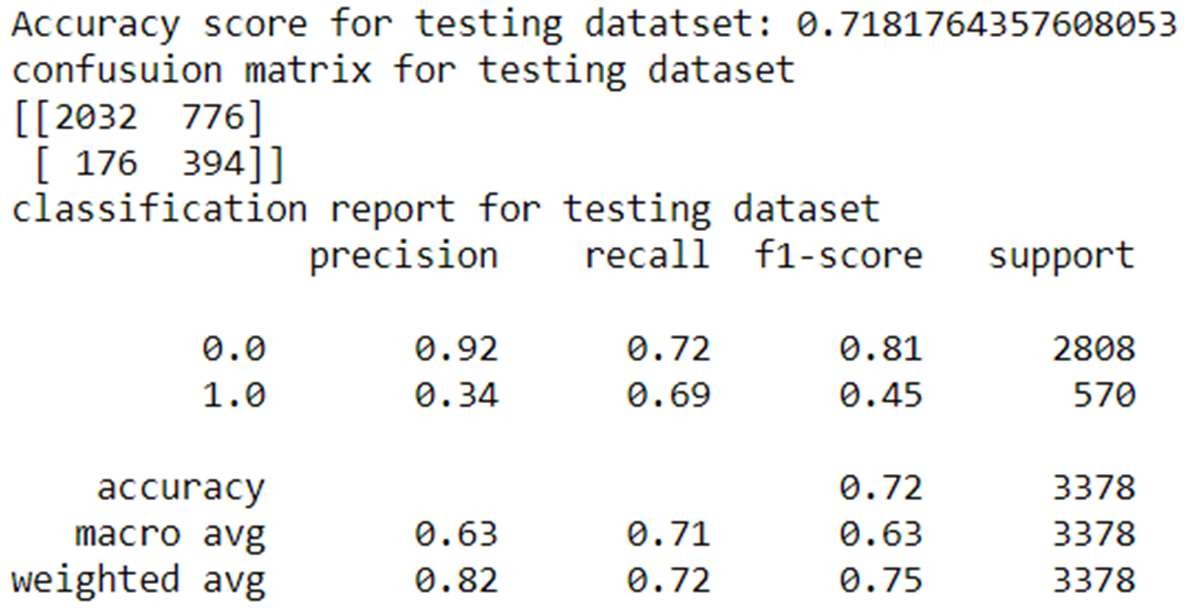
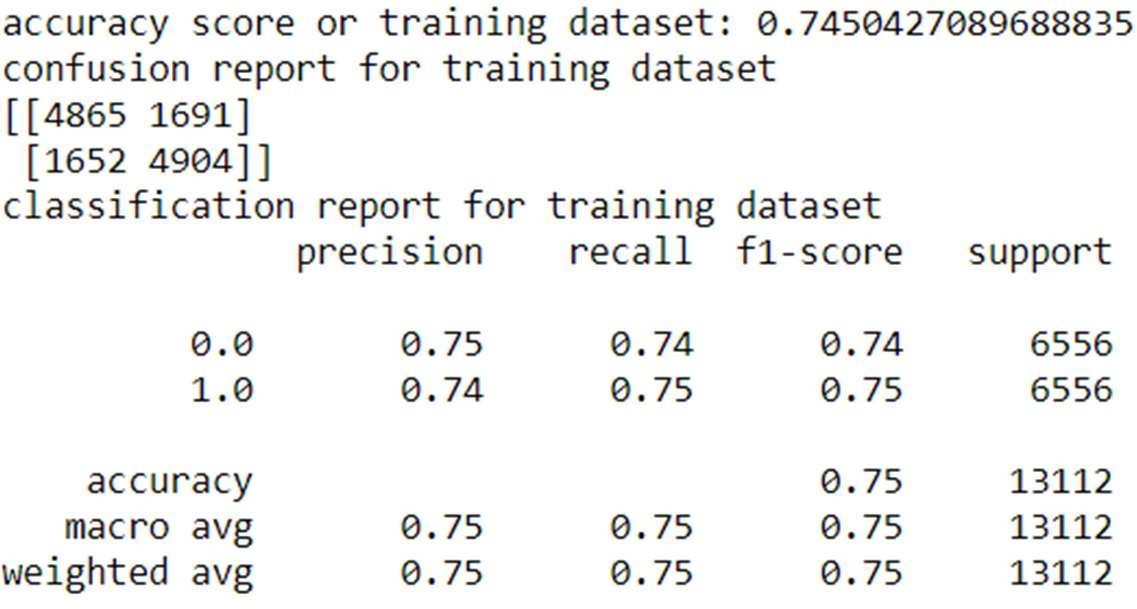
Below is the AUC score and ROC curve for training and testing dataset: -

***Fig 59: ROC Curve and AUC Score Bagging On Original Dataset***

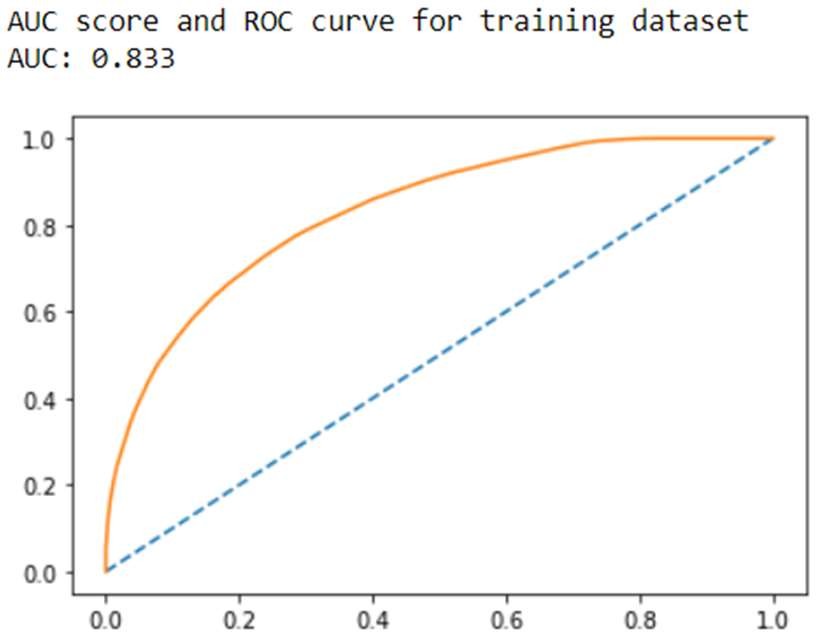
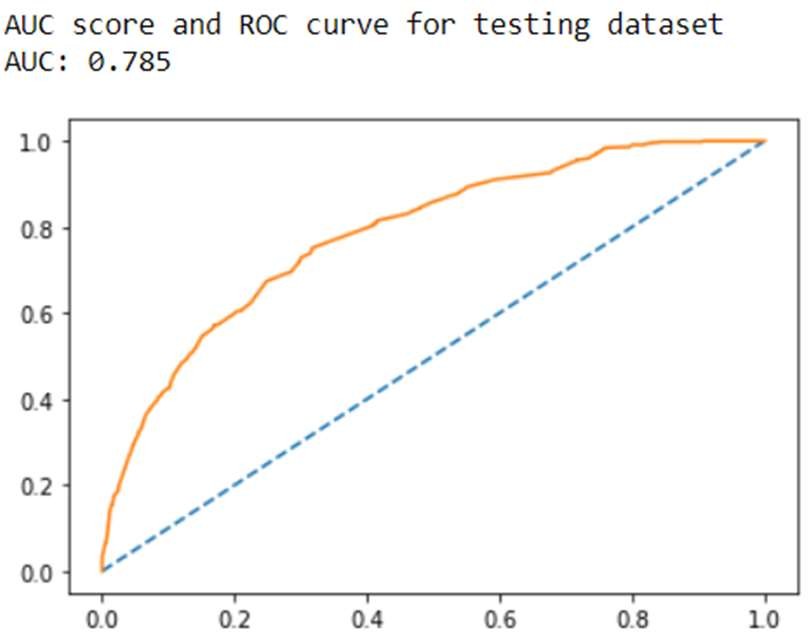
### Bagging on Balanced dataset: -

Below are the accuracy scores, confusion matrix and classification report for training and Testing dataset:-



***Fig 60: Accuracy Parameters from Bagging on Balanced Dataset***

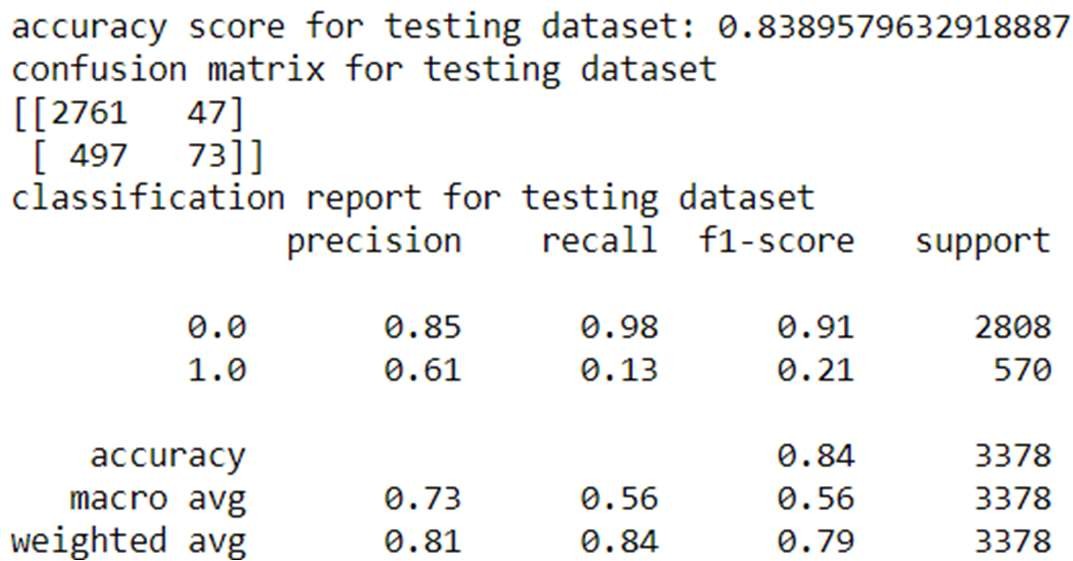
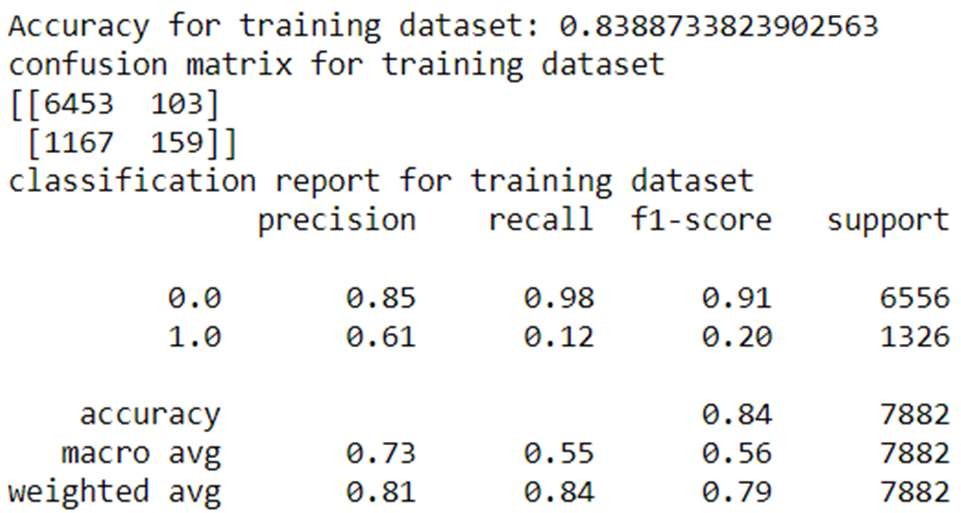
Below is the AUC score and ROC curve for training and testing dataset: -

***Fig 61: ROC Curve and AUC Scores from Bagging on Balanced Dataset***

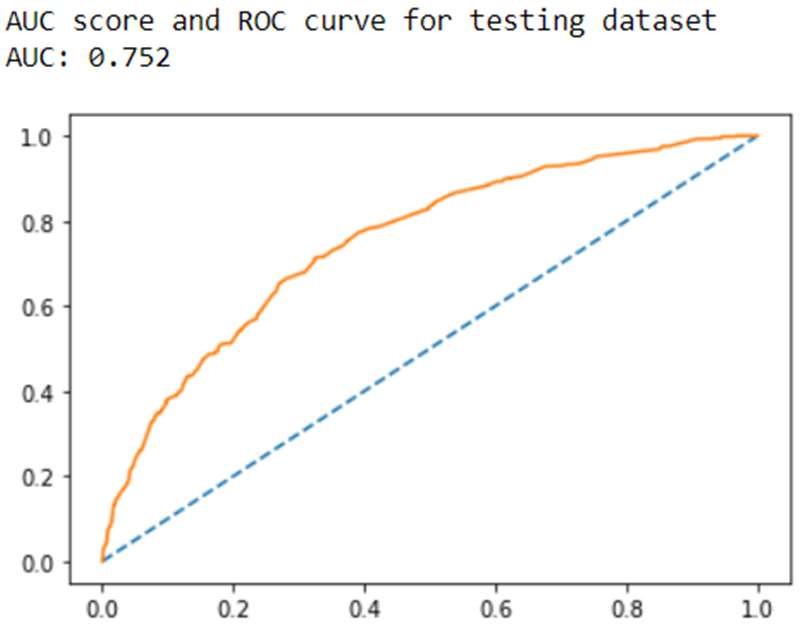
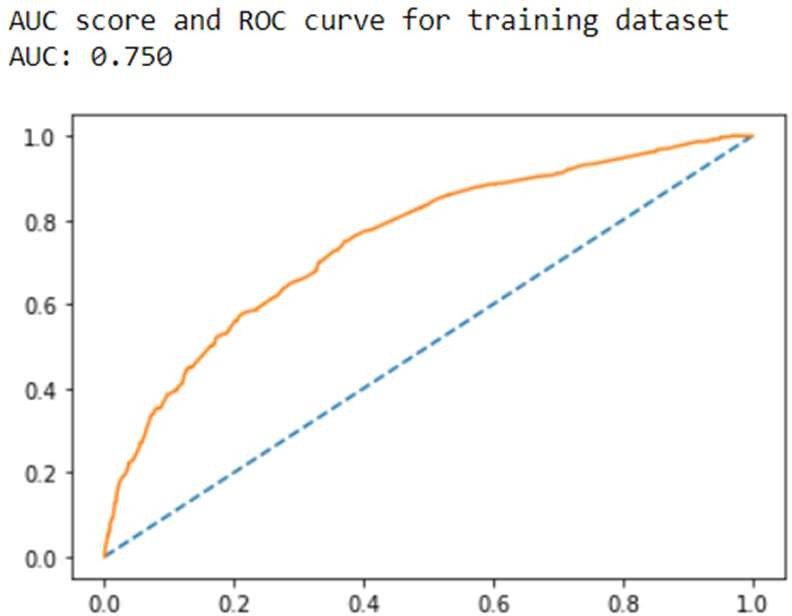
**Building Ada-Boost Model on original dataset: -**

Below are the accuracy scores, confusion matrix and classification reports for training and Tetsing dataset: -



***Fig 62: Accuracy Parameters from Ada-Boosting on Original Dataset***

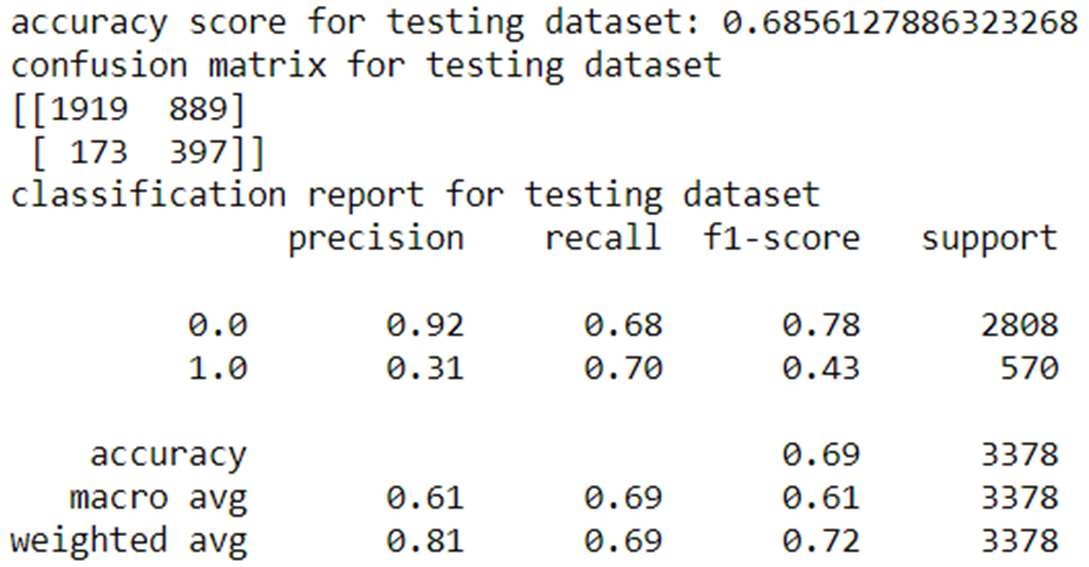
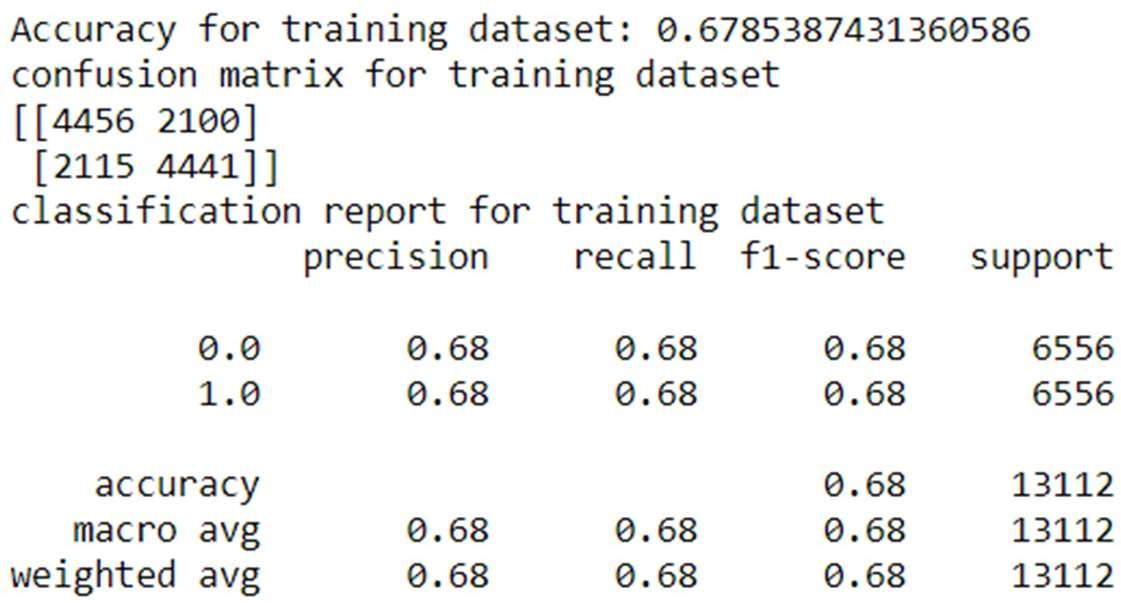
Below are the AUC scores and ROC curve for training and testing dataset: -



***Fig 63: ROC curve and AUC score from Ada-Boosting on Original Dataset***

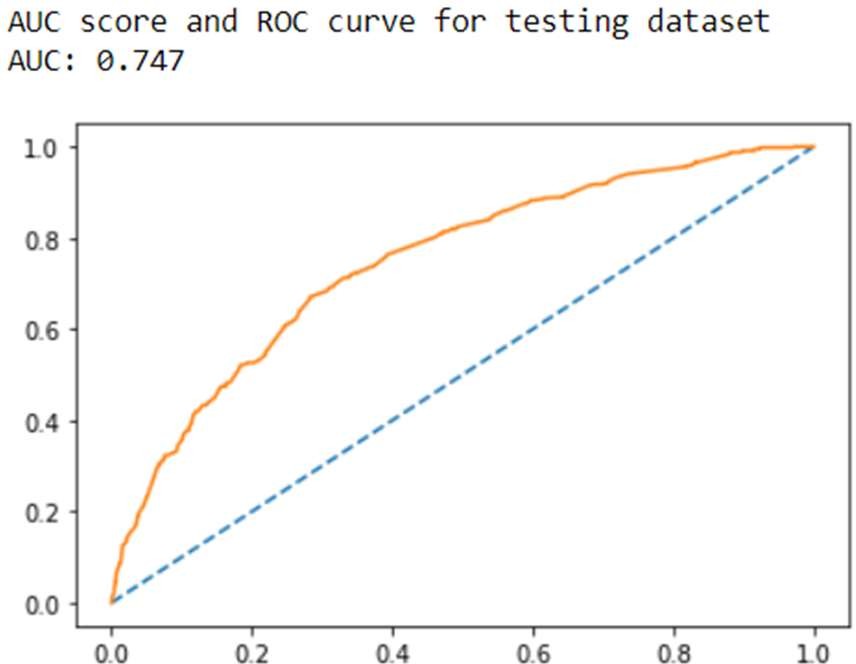
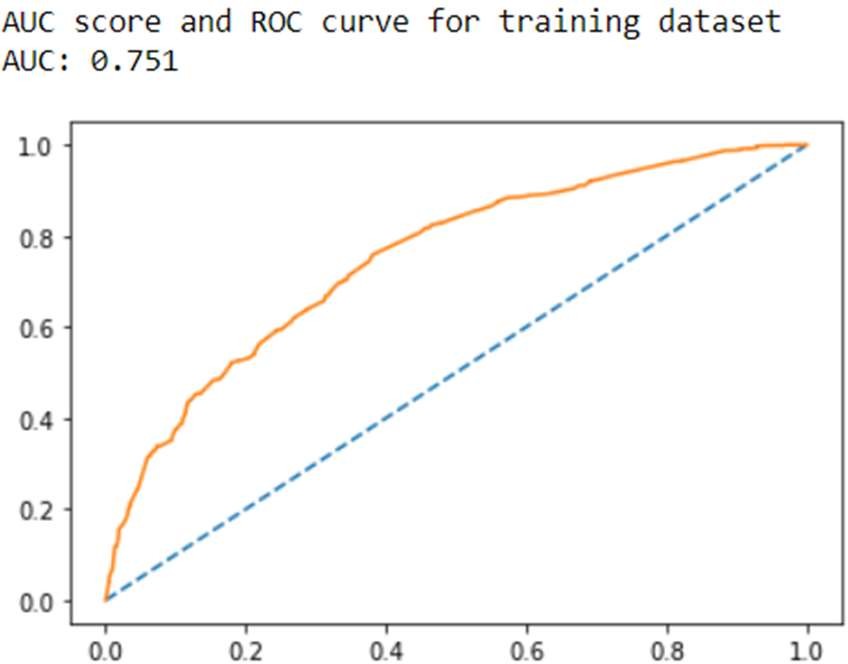
### Building Ada-Boost Model on balanced dataset: -

Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -



***Fig 64: Accuracy Parameter from Ada-Boosting on Balanced Dataset***

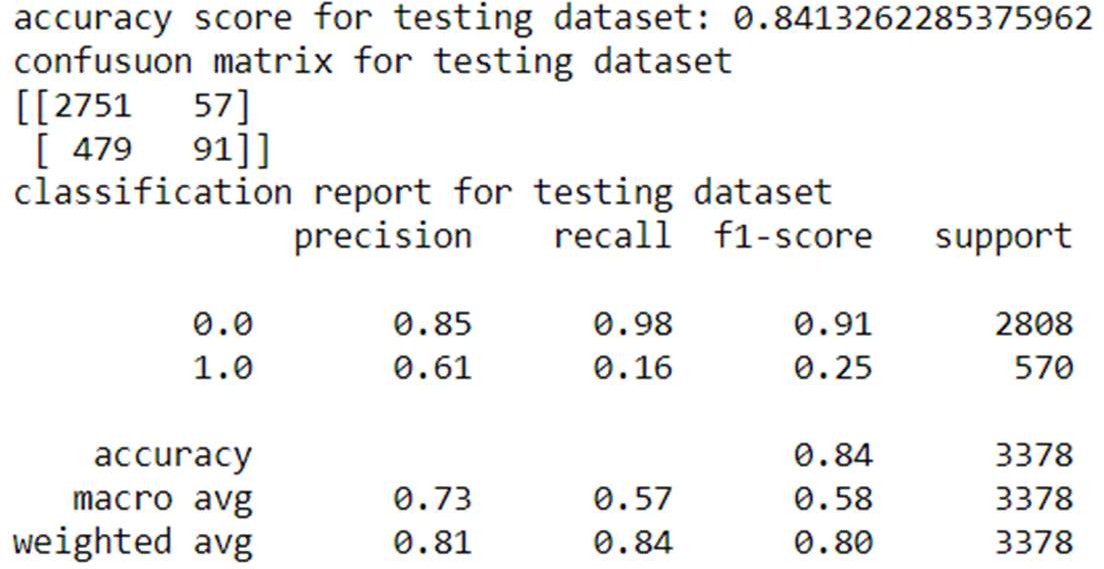
Below are the AUC scores and ROC curve for training and testing dataset: -



***Fig 65: ROC Curve and AUC Score from Ada-Boosting on Balanced Dataset***

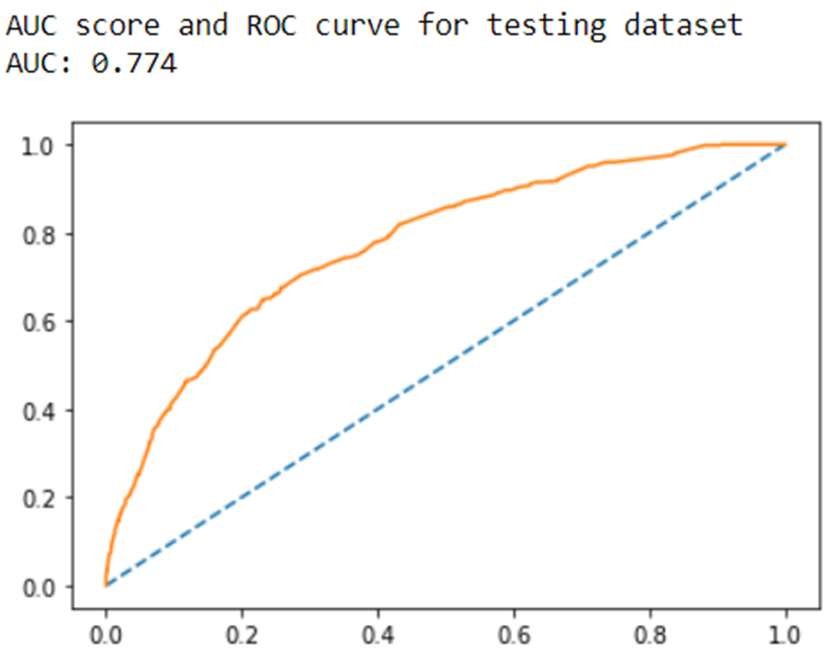
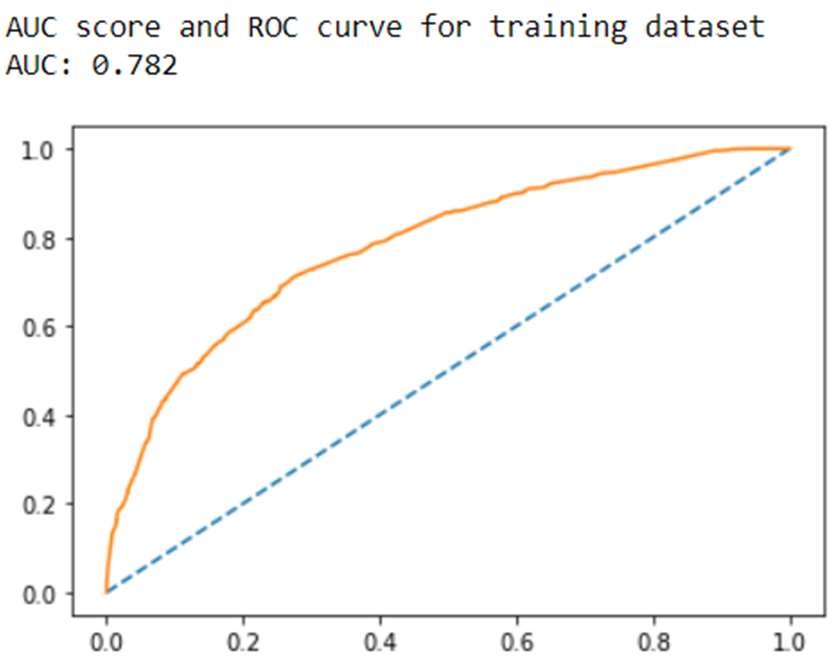
### Building Gradient Boosting Model on original dataset: -

Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -



***Fig 66: Accuracy Parameter from Gradient-Boosting on Original Dataset***

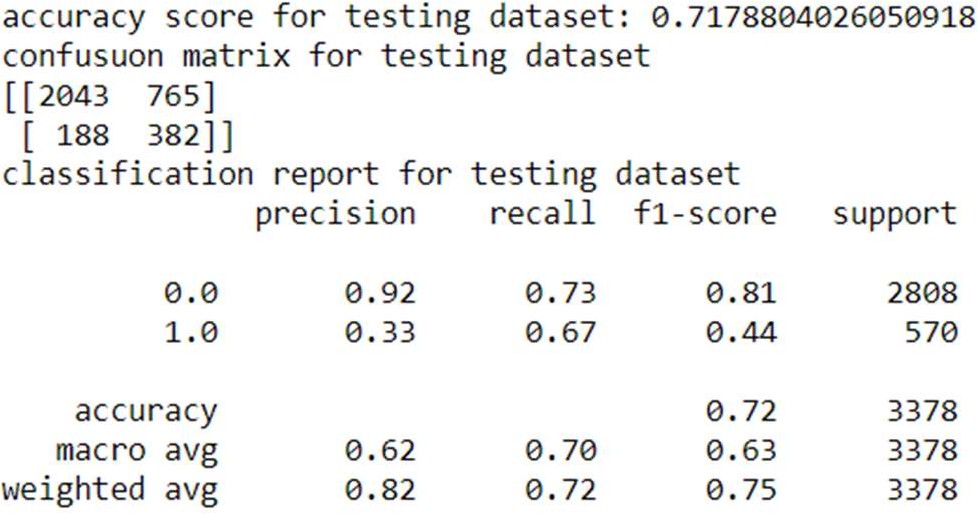
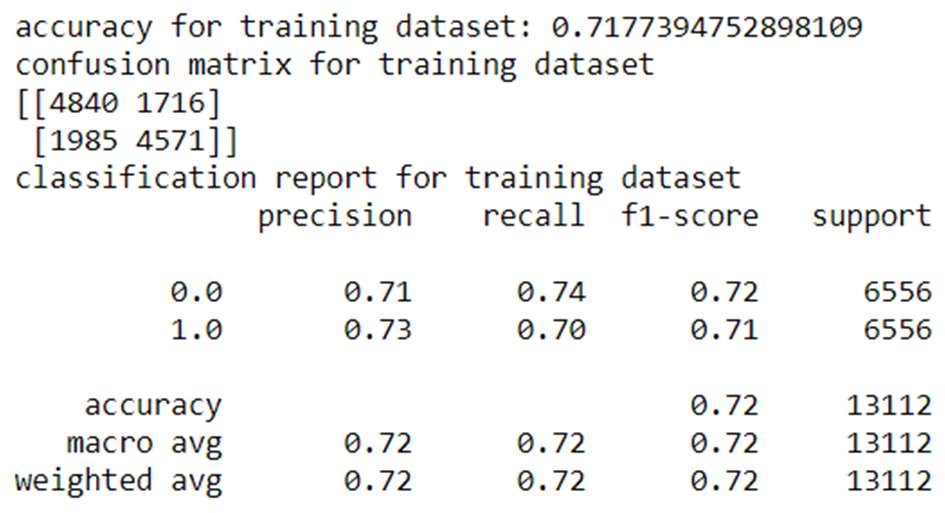
Below are the AUC scores and ROC curve for training and testing dataset: -



***Fig 67: ROC Curve and AUC Score from Gradient-Boosting on Original Dataset***

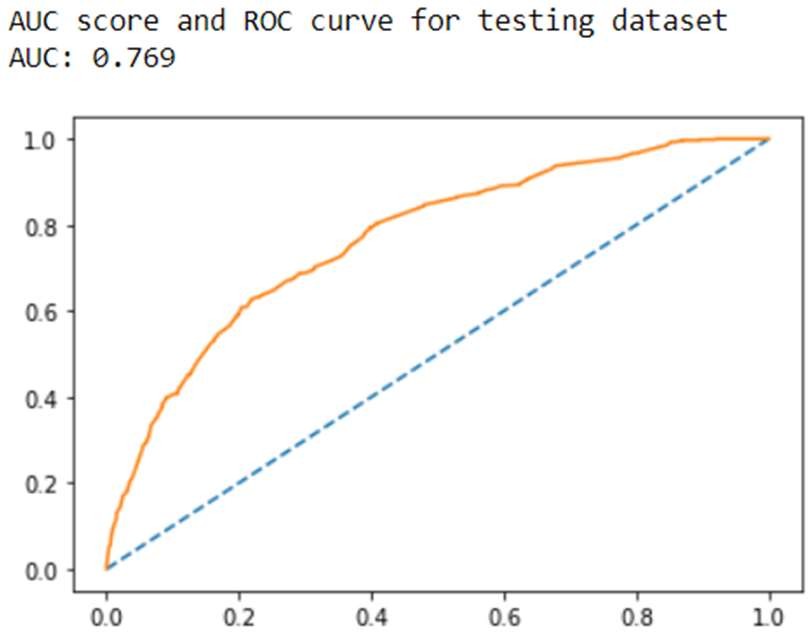
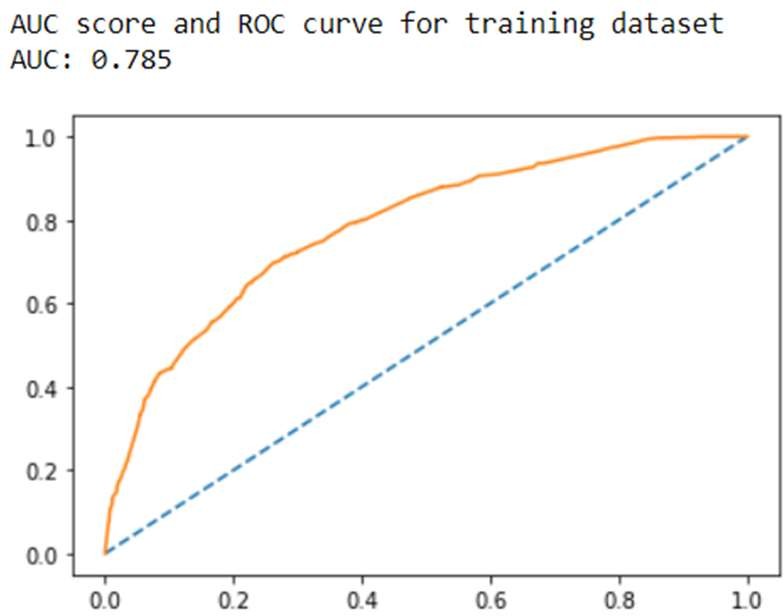
**Building Gradient Boosting Model on balanced dataset: -**

Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -



***Fig 68: Accuracy Parameter from Gradient-Boosting on Balanced Dataset***

Below are the AUC scores and ROC curve for training and testing dataset: -



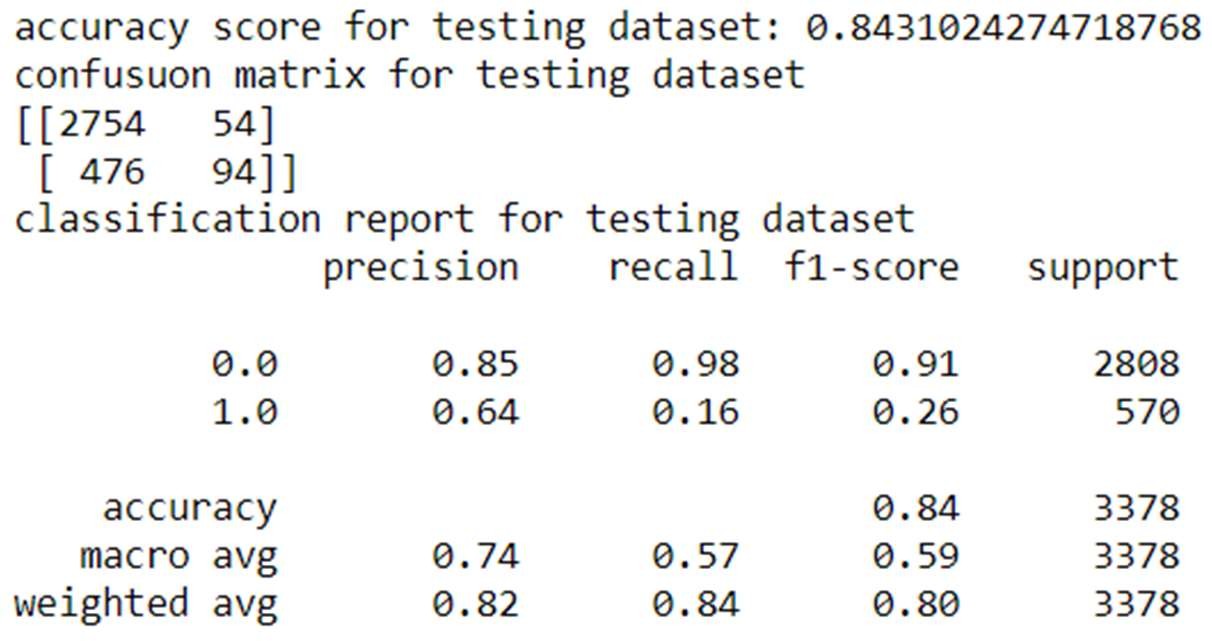
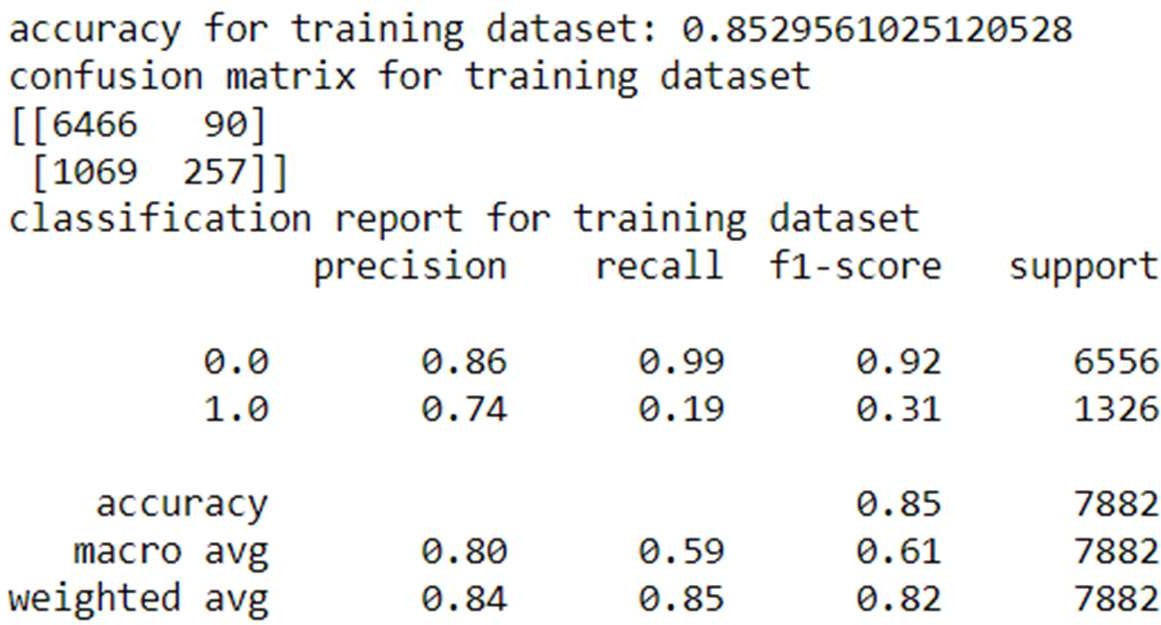
***Fig 69: ROC Curve and AUC Score from Gradient-Boosting on Balanced Dataset***

**Inferences from Bagging and Boosting model: -**

* Form both the model above we can notice that the model is still outfitted.
* When to come to model performance over balanced dataset it performs well in training dataset however the accuracy reduces when it comes to training dataset.

**Building Support Vector Machine (SVM) Model on Original dataset: -**

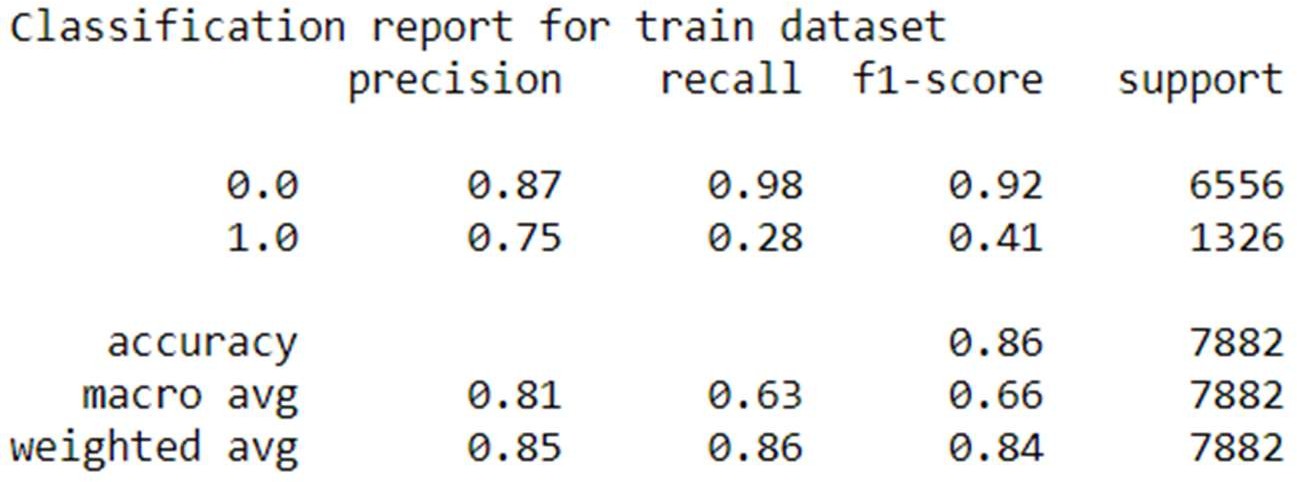
Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -

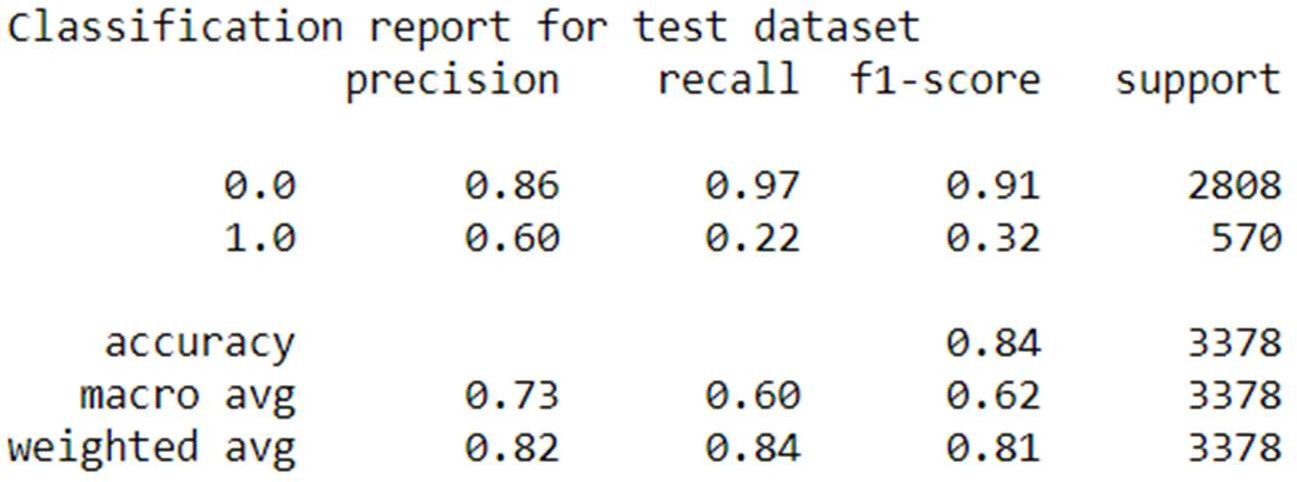


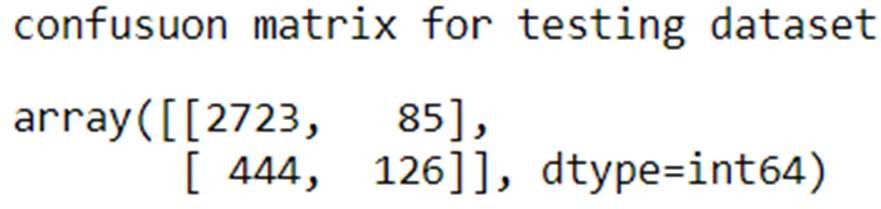
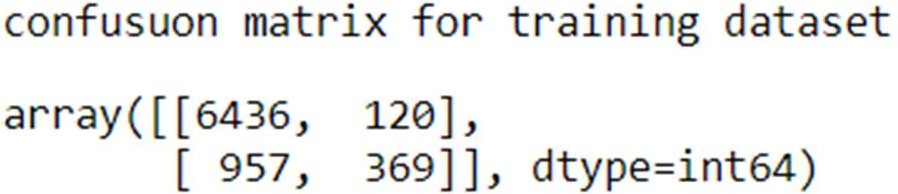
***Fig 70: Accuracy parameter Scores from SVM on Original Dataset***

**Building Support Vector Machine (SVM) Model on Original dataset Using Hyper-Parameter: -**

Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -

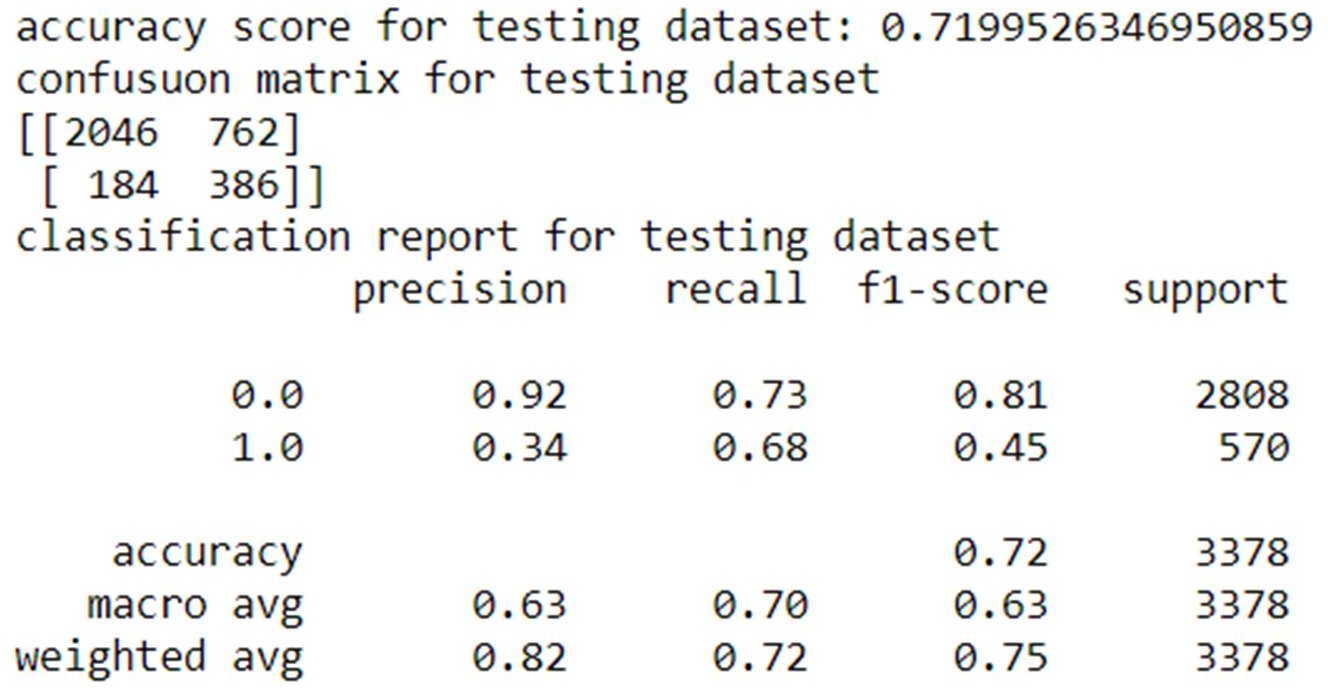






***Fig 71: Accuracy parameter Scores from SVM on Original Dataset Using Hyper-Parameter***

**Building Support Vector Machine (SVM) Model on Balanced Dataset: -**

Below are the accuracy scores, confusion matrix and classification reports for training and Testing dataset: -

***Fig 72: Accuracy parameter Scores from SVM on Balanced Dataset***

**Inferences from SVM model: -**

From the above we can conclude that the data is neither “Overfit” nor “Underfit” in nature. And we can also inference that the model built using balanced data is best optimized model for prediction. However, we can see significant variations in accuracy score, F1 score, recall values, precision values, ROC curves and AUC scores when compared with model built on imbalanced dataset. Model built on balance data set using SMOTE technique works well in training dataset and testing dataset.

### Overall Model Building Comparison across parameters: -



***Table 9: Comparison across Various Models***

### Indicators/symbols for above tabular data: -

* CV: - indicates scores for model built on best params obtained from GridSearchCV with model name as prefix.
* SM: - indicates scores for model built on balanced dataset with model name as prefix.
* KNN-5: - Indicates KNN model built with N\_neighbors as “5”.

### Inferences on final model: -

* From the above tabular representation of all the scores for training and testing dataset across various model we can conclude that the KNN model with default values of hyper-parameters is best optimized for the given dataset. (highlighted in BOLD)
* There is marginal difference in accuracy for Logistic regression and LDA, but comparatively LDA had a little better performance than logistic regression.
* Model with bagging and boosting is also well optimized but difference in accuracy for training and testing dataset is little on the higher side as compared to KNN.
* Other models namely Naïve Bayes, LDA and SVM worked well on training dataset but the accuracy came down when performed over testing dataset. Which indicates overfitting of data in that model.
* All models built on balances dataset showed overfitting.

### We also understand that the accuracy and other measuring parameter of a model can be improved by trying various other combinations of hyper-parameter. Model building is an iterative process. Model performance both on training and testing dataset can be improves

**Implication of final model on Business: -**

* Using the model built above business can plan various strategies to make customers stick with them.
* They can roll out different Offers and discounts as family floater.
* They can give regular discount coupons if paid by their e-wallet platform.
* Discount vouchers of other vendors or on next bill can be provided based in minimum bill criteria.
* This model gives business an idea where they stand currently and what best they can do to improve on the same.