CVPR
#

Risk Analytics in Banking and Financial Services

Abhinav Gudipati 2019227 | Nikhil 2019259 | Rohit Kumar Vishwas 2019269

1. Introduction

India has been ranked 5th in the list of countries with the highest Non - Performing Assets. It implies India possesses more defaulters each year regarding the subject's ability to repay their loans. Noticeably this is an alarming figure and with the rise of Bank frauds, figured at high as 100 Crore per day. With this as our motivation, we've decided to work on a problem that suitably allows us to identify whether or not a bank can engage in business with a client based on their previous history. Using ML techniques taught, our project aims at creating an efficient model where we can appropriately find the right set of metrics from a diverse group of features that helps us in identifying whether a subject can repay their loan once taken.

2. Related Work

From the Paper 1,Paper 2 and Paper 3, we've been motivated to work on the problem of how most Banks' digital adoption strategies have been injected into the most mundane tasks. With India's worsening of its ranking in the most number of Non-Performing Assets cases, the current problem we are working on has been largely unexplored as most data about individuals is kept private within Banks. The most demanding result we've observed is that our model accounts for an ID's previous application data and the subject's current application data when applying for a Loan. The subject has deemed a loan depending on the credit history scores and various other factors.

3. Data-set and Evaluation

3.1. Data-set Description

The Dataset has been taken from Kaggle. The dataset is composed of two CSV files, application_data.csv and previous_application.csv. The application_data.csv file has the Dimensionality of 307511 x 122 (rows x columns). It consists of 122 features that give the applicant's current status in need of a loan. It contains all the information of the client at the time of application. The data is also concerning whether a client has had payment difficulties. The 2nd file has the Dimensionality of 1670214 x 37 (rows x columns), containing information about the client's previous loan data. It includes whether the previous application had been approved, cancelled, refused, or unused. As a part of our interim goals, we built a united dataset that accounted for both the application_data and the previous_application data of the clients. We obtained the client IDs with the help of the SK_ID feature. We've merged both datasets into one. So our united dataset is only limited to those IDs that match both datasets.

3.2. Data Visualisation

From Fig 2 we infer that our data is extremely skewed. These 4 pie chart represents the distribution of 4 features column with binary values. we can see that almost all of them are biased towards one values. The corresponding column names are: client by Gender, Client by contact type, client owning the car and client by contract type, that is cash loans vs revolving loans. From Fig 3,we observe that our data is not much linearly separable. From fig 4, it represents the correlation between each and every features and how its related to each other. From fig 5, it represents the min and max values of some important feature, which would be used for standardization or normalisation purposes. From fig 6, we can see that how different perspective of amount variables related with each other for both class varibales. From fig 7, we see that almost everyday peak hours achieved maximum around 2'o clock.

3.3. Evaluation

We have merged both CSV files and appropriately split training and testing sets in a 4:1 ratio. Since our dataset's number of features has exceeded 100, we dropped some of the features that did not contribute much to the evaluation and have more null values than the data itself(see fig 8). Furthermore, we have applied dimensionality reduction and extracted only the necessary features using PCA (see fig 9 for best two principal components). We calculated the explained variance ratio for all features and extracted those that contribute more towards better model training(see fig 1). After PCA modelling of our united dataset, we are left

with a dataset having the final dimension of 291057 rows against 44 features. From fig 10, we can compare the distribution of class variable for unsampled and undersampled dataset. After undersampling our resultant data becomes 38040 x 44.

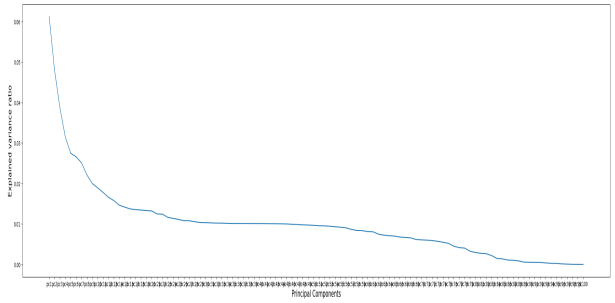

Fig.1 Principal Components vs Explained Variance Ratio

4. Methodology

Regarding the fig 2 and 3, the dataset obtained after dimensionality reduction remains skewed, and it is not much linearly separable. We've applied both undersampling and oversampling by resampling with sampling_strategy as 'majority' concerning undersampling and 'minority' concerning oversampling. Oversampling strategy creates duplicate or new synthetic examples in the minority class, whereas undersampling removes or merges examples from the majority class. We've correspondingly taken both the oversampling and undersampling cases respectively and tested the model's efficiency in each scenario. The models we have used are as follows:

4.1. Decision Tree

To get the initial evaluation of the decision tree, we applied it with a brute force implementation. We get the highest accuracy of 83 per cent on our unsampled data.

From table 1, we see that precision, recall and f1 score getting improved in case of undersampling and oversampling against unsampled data and along with decreasing accuracy score, which was earlier higher owing to biases in the unsampled data. Thus, our model getting improved in both the cases of undersampled and oversampled with the undersampled data performing better than the oversampled data.

Moving on, we used k- fold cross-validation to optimise the model and evaluated accuracy on each decision tree depth along with mean cross-validation accuracy. We can observe from fig 11 that depth-5 Tree achieves the best mean cross-validation accuracy at 67.Moreover, we evaluated unsampled and sampled data on the decision tree with the best depth (that is 5). We observed test accuracy as 91.70 and training accuracy as 91.83 on our unsampled data. Finally, we used Grid search on the decision tree for finding the best parameters with fitting undersampled data. And after getting the best parameter we applied a decision tree with the best parameters on each data and get the result as mentioned in table 2 . It improved all evaluation scores for every data. And a maximum accuracy score in case of unsampled data up to 91 per cent.

4.2. Stochastic Gradient Descent

We applied SGD with a brute force implementation. We get the highest accuracy of 91 per cent on our unsampled data as mentioned in table 3.To optimise our model, we used grid search for optimising and getting best parameters for it. We've implemented the SGD classifier with an L2 penalty and a Log loss function. Consequently, we evaluated our SGD classifier over learning rates ranging from 0.0001 to 1000. Drawing from those evaluations we inferred that with a 0.01 learning rate, we achieved the best AUC score, with a score of .6443 on our unsampled data observed in fig 12. With the above observations can infer that if the learning rate is too high, then the graph will oscillate too much, else, a too low learning rate will mean that the training is too slow as the weights are changing at a small pace.

Finally, we used the Grid search on the SGD for finding the best parameters with fitting undersampled data. And after getting the best parameter we applied SGD with the best parameters on each data and get the result as mentioned in table 4.

4.3. Random Forest

We first evaluated Random forest with default parameters on both sampled and unsampled data. We get a maximum accuracy score of 91.71 for unsampled data, whereas a more excellent f1 score for under-sampling n table 5. Furthermore, we used Grid cross-validation on our model with five-fold cross-validation. We get Table 6 as result after applying random forest with the best parameters obtained from grid search cross-validation.

4.4. Gaussian NB

Our intention behind implementing the Gaussian Naive Bayes model is that our data's dimensions are very high, i.e beyond 100 features. From Table 7 we infer that without applying any feature tuning we've run the model for each of the 3 cases.

After implementing Grid Search CV Further we applied Grid Search CV on our Gaussian NB model with KFold value = 10, our var_smoothing value varies from np.logspace(0,-9, num=100). Our Grid Search CV model fits 10 folds for each of 100 candidates totalling 100 fits. We obtain the best parameters for var_smoothing at a value of 0.6579. Further, beyond optimising our Gaussian NB

model with the best parameter we obtain a scoring matrix for each of our 3 cases.

4.5. Logistic Regression

We first implemented simple logistic regression on unsampled data, with and without penalty (L1 and L2). We get an accuracy score across every model same, which is 91.70 per cent (see Table 9). Further we applied Grid Search CV on our logistic regression model for optimising parameters. We used unsampled data with grid cross-search validation and the best value of c obtain is 0.1 as observed in fig 15. We have further used the best parameter to enhance the logistic regression with OVO and OVR models. We calculated the metrics score on each sampled and unsampled data with and without penalty and get the following tables(table 10 and table 11) as result.

4.6. K-Nearest Neighbours

Implemented KNN because it is a nonparametric algorithm, we don't exactly apply any assumption on our data. Since our training data is large, we can assume that KNN is effective which is reflected in the accuracy scores we've obtained for each of the 3 cases. In KNN model, we prioritise the n neighbours parameters, thats why we didn't implement simple model and directly implemented KNN with grid search cross -validation method. We are fitting 10-fold for each of 11 candidates totalling 110 fits. We get n_neighbours = 100 as a best parameters.

4.7. XGBoost Classifier

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems. We used it to further optimise the logistic regression model. We implemented it using grid search cross validation having parameters max-depth ranges between 2 to 10, and estimators ranging from 60 to 220 with steps of 40 and checking over learning rate 0.1 to 0.05. We used ROC_AUC for scoring purposes. We have fitted 10-fold for every 96 candidates totally of 960 fits.

4.8. Multi Layer Perceptron Classifier

As MLP is suitable for classification prediction problems where inputs are assigned a class or label. They are also suitable for regression prediction problems where a real-valued quantity is predicted given a set of inputs. From Table 13 we infer that without applying any feature tuning we've run the model for each of the 3 cases

After implementing Grid Search CV on MLP Further we applied grid search cross-validation to optimise the best parameters values to increase the metrics score. We get the table Table 14 as the result after applying the best parameters we get from the grid search.

5. Results & Analysis

5.1. Decision Tree

The values of all metrics have increased after applying grid search and using best parameters with it. we can see an increase in values of precision, F1 score, accuracy score and recall score compared to the model that we applied without using the best parameters. The Decision Tree achieves a maximum score as follows: precision score: 0.574 recall score: 0.573, F1 score: 0.572 and accuracy score: 0.917.

5.2. Stochastic Gradient Descent

The values of all metrics have increased after applying grid search and using best parameters with it . we can see an increase in values of precision, F1 score, accuracy score and recall score compared to the model that we applied without using the best parameters. The SGD Model achieves a maximum score as follows: precision score: 0.607, recall score: 0.607, F1 score: 0.607 and accuracy score: 0.917.

5.3. Random Fores

The values of all metrics have increased after applying grid search and using best parameters with it . we can see an increase in values of precision, F1 score, accuracy score and recall score compared to the model that we applied without using the best parameters(See fig 13 for Roc-curve for undersampled data). However, there is a greater increase in the accuracy score of undersampled and oversampled compared to the model without grid search. The Random Forest Model achieves a maximum score as follows: precision score: 0.811, recall score: 0.813, F1 score: 0.823 and accuracy score: 0.923.

5.4. Gaussian NB

From table Table 7, the precision score and F1 score are best for undersampled among all models (see fig 14 for ROC-curve), while recall score is best for the unsampled model. This result can be predicted as our unsampled data is skewed. All scores do not vary much in undersampled and oversampled data. The same scenario can be seen in Table 8. However, the values of all scores are higher than the previous respective table. The accuracy of unsampled is quite large compared to undersampled and oversampled accuracy. It is because unsampled data is very biased in comparison to undersampled and oversampled. The Gaussian Bayes Model score achieves a maximum score as follows: precision score: 0.599, recall score: 0.525, F1 score: 0.519 and accuracy score: 0.916.

5.5. Logistic Regression

The values of all metrics have increased after enhancement in the case of both OVO as well as OVR (see table 10 and table 11). we can see a large increase in values of precision, F1 score and recall score compared to the model that we applied without using the best parameters. The average score for these three evaluation metrics revolves around 0.49 while in the case of OVO and OVR with enhancing best parameters, these values get around 74-79. The logistic regression Model achieves a maximum score as follows: precision score: 0.749, recall score: 0.769, F1 score: 0.759 and accuracy score: 0.939.

5.6. K-Nearest Neighbours

The best parameter we calculated using grid search got as follows: n_neighbours = 100.We predict this model from undersampled data and get metrics scores as follows (see table 12): The KNN model achieves a maximum score as follows: precision score: 0.687, recall score: 0.686, F1 score: 0685. and accuracy score: 0.926.

5.7. XGBoost Classifier

The best parameters we calculated using grid search got as learning: 0.01, max-depth:5 and n_estimators:100. We predict this model from undersampled data and get metrics scores as follows: precision score Grid Search sampling on under data: 0.7693269023120815 recall score Grid Search sampling on under data: 0.7793150017794595 f1Score Grid Search sampling on under data: 0.7693043697072172 accuracy score Grid Search sampling on under data: 0.9393150017794594

5.8. Multi Layer Perceptron Classifier

The values of all metrics have increased after applying grid search and using best parameters with it . we can see in table 14 an increase in values of precision, F1 score, accuracy score and recall score compared to the model that we applied without using the best parameters. The MLP model achieves a maximum score as follows: precision score: 0.811, recall score: 0.813, F1 score: 0.823 and accuracy score: 0.923. The final metrics score describes it as one of the best models we have gotten till now among others.

5.9. Conclusion

Although XGBoost and MLP both resulting in the best models with highest scores with respect to other models with and without optimisation. However, the best model with maximum metric score was Multi Layer Perceptron with best metrics scores as :-.precision score: 0.811, recall score: 0.813, F1 score: 0.823 and accuracy score: 0.923.

6. Members Contribution

All Members contributed to each and every task and cross-validates them.

References

[1] A Formal Analysis of Fraud in Banking Sector

[2] Frauds in the Indian Banking Industry

[3] Banking Comprehensive Risk Management System Based on Big Data Architecture of Hybrid Processing Engines and Databases

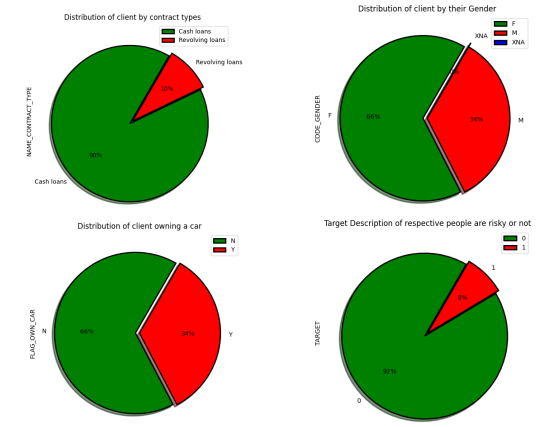

Fig.2 Data distribution of binary valued features

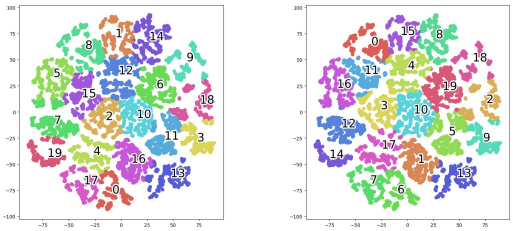

Fig.3 TSNE scatterplots of 20 features

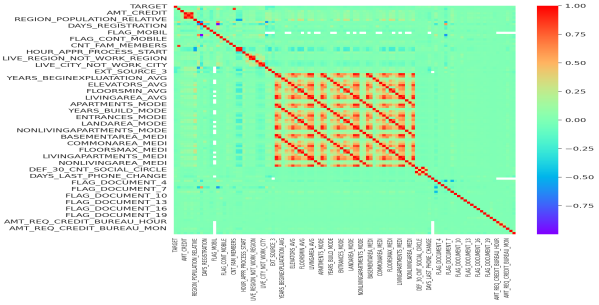

Fig.4 Correlation Matrix (Heat Map)

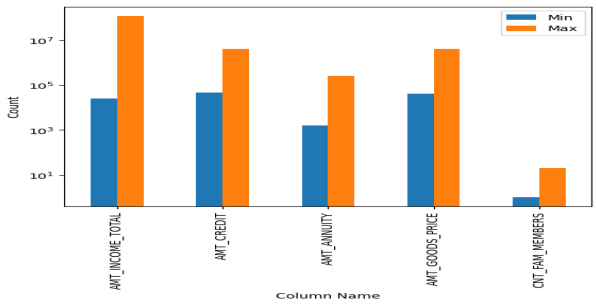

Fig.5 Min-Max values of important Features

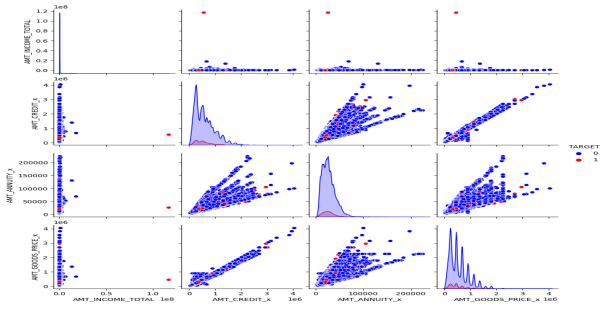

Fig.6 Pair plot between amount variables

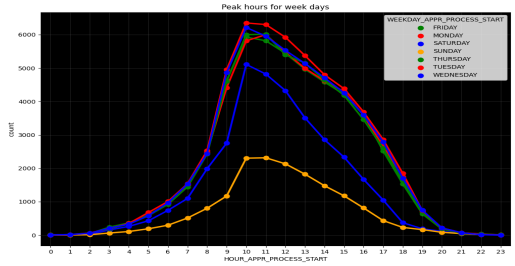

Fig.7 Peak hours of week days

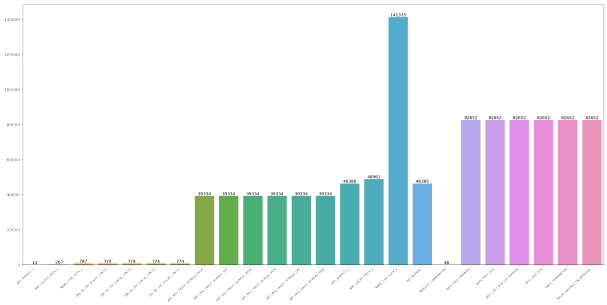

Fig.8 Features vs NaNcounts

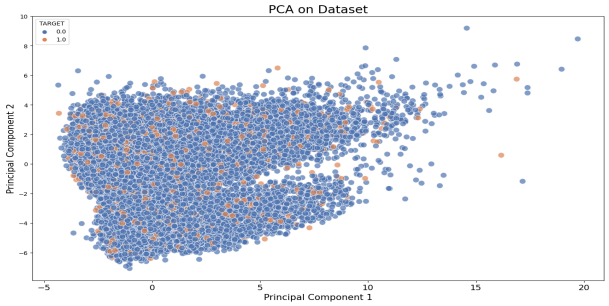

Fig.9 Principal Components vs Explained Variance Ratio

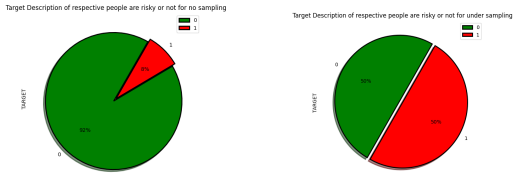

Fig.10 Unsampled vs Sampled Distribution

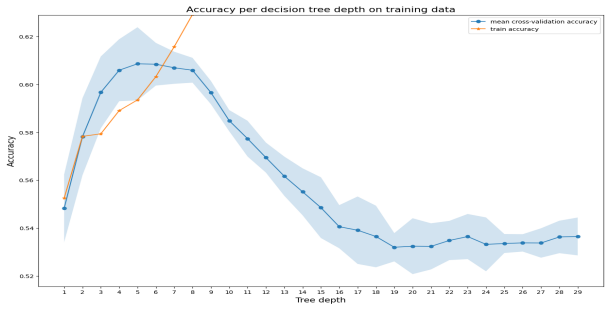

Fig.11 Decision Tree depth vs AUC score

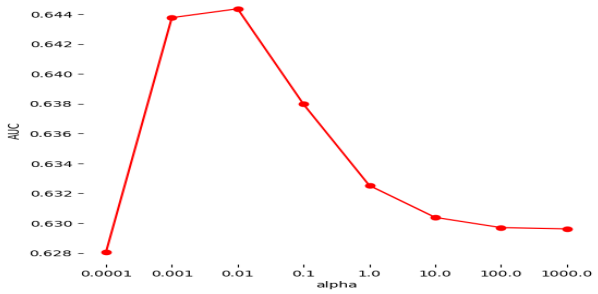

Fig.12 SGD Learning Rate vs AUC score

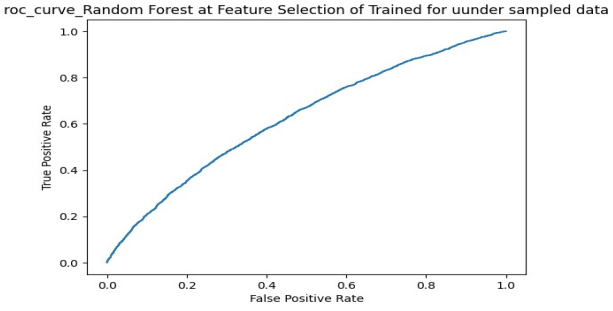

Fig.13 Random Forest undersampled ROC curve

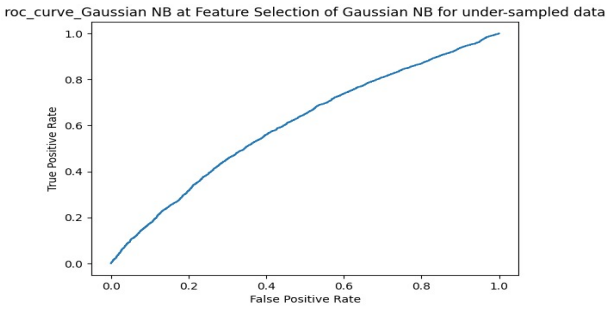

Fig.14 Guassian NB ROC-curve

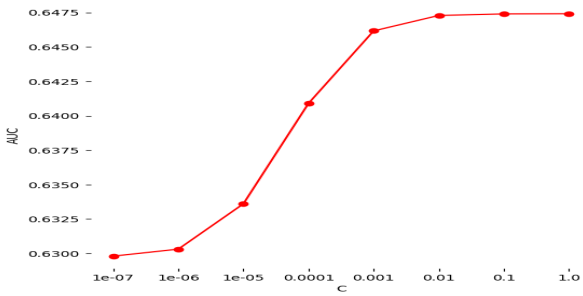

Fig.15 Logisitic Regression c vs AUC score

Decision Tree	Precision score	Recall score	F1 score	Accuracy score
unsampled	0.51187159372 96814	0.513973363525 7502	0.512417717667 174	0.838486909915 4814
undersampled	0.529845124935 4387	0.529844559585 4921	0.529842333060 9561	0.529844559585 4923
oversampled	0.543565491123 8585	0.515921479011 7445	0.424651124461 7922	0.515921479011 7445

Table.1 Metrics score on Decision Tree on without tuning

Decision Tree	Precision score	Recall score	F1 score	Accuracy score
unsampled	0.458556654985 2264	0.5	0.478382422781 5661	0.917113309970 4528
undersampled	0.574375692490 7081	0.573471502590 6736	0.572171217338 6244	0.573471502590 6736
oversampled	0.570325031941 7514	0.567244834884 8971	0.562453764804 0309	0.567244834884 8971

Table.2 Metrics score on Decision Tree with best parameters

	Precision score	Recall score	F1 score	Accuracy score
SGD without Tuning on unsampled	0.458556654985 2264	0.5	0.478382422781 5661	0.917113309970 4528
SGD without Tuning on undersampled	0.583600914157 3315	0.583523316062 1762	0.583426650556 975	0.583523316062 1761
SGDwithout Tuning on oversampled	0.604910806848 115	0.604735235169 6106	0.604569793812 5247	0.604735235169 6106

Table.3 Metrics score on SGD on without tuning

	Precision score	Recall score	F1 score	Accuracy score
SGD with best parameters on unsampled	0.458555943034 8216	0.499990634424 1107	0.478377748705 17397	0.917096131381 8456
SGD with best parameters on undersampled	0.595532743933 5716	0.595440414507 7721	0.595342642320 0858	0.595440414507 7721
SGD with best parameters on oversampled	0.607610727067 8319	0.607610466967 6139	0.607610229861 4545	0.607610466967 6139

Table.4 Metrics score on SGD with best parameters

RANDOM FOREST	Precision score	F1 Score	Recall Score	Accuracy score
Unsampled	0.458556654985 2264	0.478382422781 5661	0.5	0.917113309970 4528
Undersampled	0.586336433580 7972	0.586303047032 4642	0.586321243523 316	0.586321243523 316
Oversampled	0.580643731235 4431	0.580521923156 0215	0.580590780527 0947	0.580590780527 0946

Table.5 Metrics score on Random Forest without tuning

RANDOM FOREST after CV	Precision score	Recall score	F1 score	Accuracy score
Unsampled	0.771709406315 277	0.783730569948 1865	0.773185390896 9007	0.923730569948 1865
Undersampled	0.811709406315 277	0.803185390896 9007	0.823730569948 1865	0.923730569948 1865
Oversampled	0.811709406315 277	0.813185390896 9007	0.813730569948 1865	0.783730569948 1865

Table.6 Metrics score on Random Forest with best parameters

Gaussian Naive Bayes	Precision	F1 Score	Recall Score	Accuracy
Unsampled	0.5207174498152 82	0.4898401665473 1596	0.5669270594391 854	0.9148182505325 362
Undersampled	0.5614903912541 241	0.5077318559898 053	0.5163730569948 187	0.5163730569948 186
Oversampled	0.5614903912541 241	0.4976213966264 835	0.5134115046734 224	0.5134115046734 223

Table.7 Metrics score on Guassian NB without tuning

Gaussian Naive Bayes after CV	Precision	F1 Score	Recall Score	Accuracy
Unsampled	0.5233470419296 219	0.5196309726759 783	0.5247027807802 078	0.9163986806849 497
Undersampled	0.5956703467529 584	0.4137925623312 283	0.5238341968911 917	0.5238341968911 917
Oversampled	0.5995915655104 094	0.4155874634156 158	0.5250060876243 28	0.5250060876243 28

Table.8 Metrics score on Guassian NB with best parameters

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.4585559430348216	0.4999906344241 1074	0.4783777487051 7397	0.9170961313818456
L1	0.4585559430348216	0.4999906344241 1074	0.4783777487051 7397	0.9170961313818456
L2	0.4585559430348216	0.4999906344241 1074	0.4783777487051 7397	0.9170961313818456

Table.9 Metrics score on Logistic Regression without tuning

Logistic Regression enhancing with OVO on unsampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.5585559430348216	0.5999906344241 1074	0.5783777487051 7397	0.9270961313818456
L2	0.5885559430348216	0.6199906344241 1074	0.6083777487051 7397	0.9170961313818456

Logistic Regression enhancing with OVO on undersampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.7384989193223 92	0.7584974093264248	0.7484960471714063	0.9384974093264248
L2	0.7479800584662087	0.7679792746113 99	0.7579785631654775	0.9379792746113 99

Logistic Regression enhancing with OVO on oversampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.7393455501709768	0.7693337329312379	0.7493231775824749	0.9393337329312379
L2	0.7393269023120815	0.7693150017794595	0.7593043697072172	0.9293150017794594

Table.10 Metrics score on Logistic Regression enhancing OVO

Logistic Regression enhancing with OVR on unsampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.5685559430348216	0.5999906344241 1074	0.5883777487051 7397	0.9170961313818456
L2	0.5785559430348216	0.5999906344241 1074	0.5823777487051 7397	0.9270961313818456

Logistic Regression enhancing with OVR on undersampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.7484989193223 92	0.7584974093264248	0.7484960471714063	0.9384974093264248
L2	0.7379800584662087	0.7679792746113 99	0.7579785631654775	0.9379792746113 99

Logistic Regression enhancing with OVR on oversampled data

	Precision score	Recall score	F1 score	Accuracy score
Without penalty	0.7393455501709768	0.7693337329312379	0.7593231775824749	0.9293337329312379
L2	0.7493269023120815	0.7693150017794595	0.7593043697072172	0.9293150017794594

Table.11 Metrics score on Logistic Regression enhancing OVR

KNN using Grid Search CV	Precision score	Recall score	F1 score	Accuracy score
unsampled	0.5585559430348216	0.5999906344241074	0.5783777487051739 7	0.9170961313818456
Undersampled	0.6878791649460997	0.6869430051813471	0.6858400136028733	0.9269430051813471
oversampled	0.6868791649460997	0.6839430051813471	0.6828400136028733	0.9239430051813471

Table.12 Metrics score on KNN with best parameters

MLP without tuning	Precision	F1 Score	Recall Score	Accuracy
Unsampled	0.5841709406315277	0.5831853908969007	0.5837305699481865	0.9137305699481865
Undersampled	0.6417094063152 77	0.6431853908969007	0.6537305699481865	0.7837305699481865
Oversampled	0.6597305699481865	0.6491853908969007	0.6597305699481865	0.7957305699481865

Table.13 Metrics score on MLP without tuning

MLP after CV	Precision	F1 Score	Recall Score	Accuracy
Unsampled	0.7717094063152 77	0.7731853908969007	0.7731853908969007	0.9237305699481865
Undersampled	0.8117094063152 77	0.8031853908969007	0.8237305699481865	0.8837305699481865
Oversampled	0.8117094063152 77	0.8137305699481865	0.8031853908969007	0.8837305699481865

Table.14 Metrics score on MLP with best parameters