Abstract

Having superior customer satisfaction helps to build a sustainable business and strength customer relationships with the company. What variables measure customer satisfaction? This project uses an anonymized dataset consisting of a large number of numerical values derived from the Kaggle Santander Customer Satisfaction competition. The “target” column is used as a predictive variable which consists of 0s and 1s indicating 0 as a satisfied customer and 1 as a satisfied.

In the working process, we found that data is imbalanced where 96% of the data consist of 0, there is a high collinearity between features, and large numbers of features are constant and quasi constant. Thus, we utilized SMOTE to deal with data imbalance, and lasso path to eliminate the constant or quasi-constant variables. Overall we went from 369 to 70 variables.

Our first Random Forest model with all features results in AUC of 0.78 and F1 score of 0.24. Second, we model Random Forest with lasso features which results in AUC of 0.80 and F1 of 0.2.

Then, we perform other models which results in;

- Very naive model with accuracy score of 96% and F1 score of 0.00.
- Decision tree with accuracy score of 80% and F1 score of 0.28.
- Random forest with accuracy score of 86% and F1 score of 0.23.
- AdaBoost with accuracy score of 85% and F1 score of 0.23.

Table of contents

- Data Pre-processing
    - Eliminating Collinearities
- Data Cleaning
- Exploratory Data Analysis
- Modeling
    - Base Model
    - Ensemble Model
    - Experimentation
- Interpretation
    - Model Comparison Table
    - Default Feature Importance
    - Permutation Importance
    - Partial Dependence Plots
- Conclusion
- References

Project details
Data Pre-processing

The data consists of 76,020 rows and 370 columns. Since, some portion of the data includes constant values, we removed them to obtain clean data.

Afterwards, we split the data with a train size of 0.8. After splitting,

- Xtrain size is 60,616 rows and 369 columns, and ytrain is 60816.
- Xval size is 15,204 rows and 369 columns, and yval is 15204.

For upsampling we utilize SMOTE to deal with data imbalance.

SMOTE is an improvement over minority oversampling. SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line.

Before SMOTE:

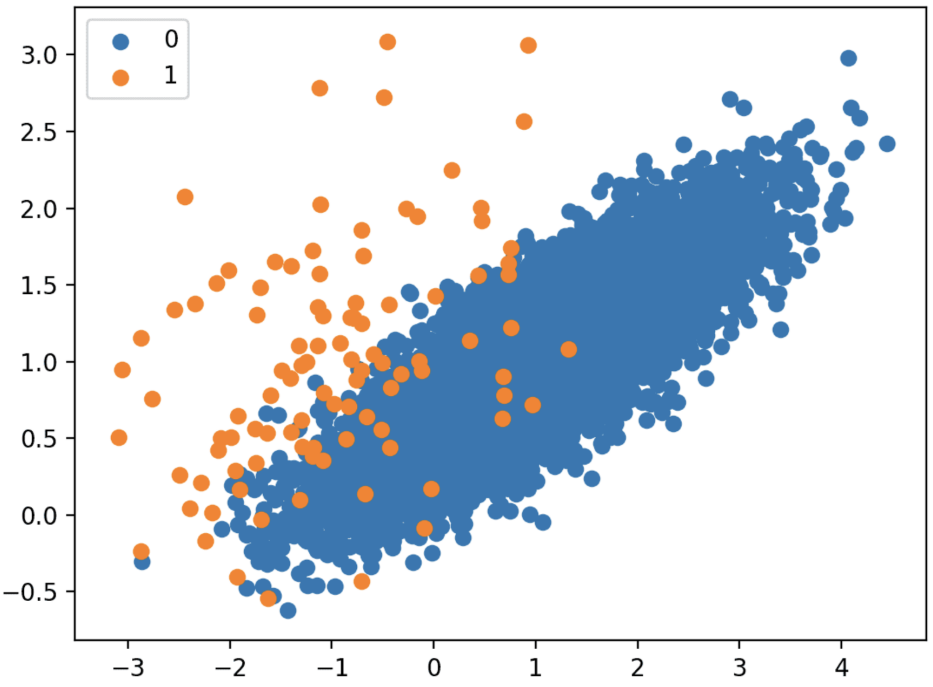

Figure 1: Distribution before SMOTE

After SMOTE:

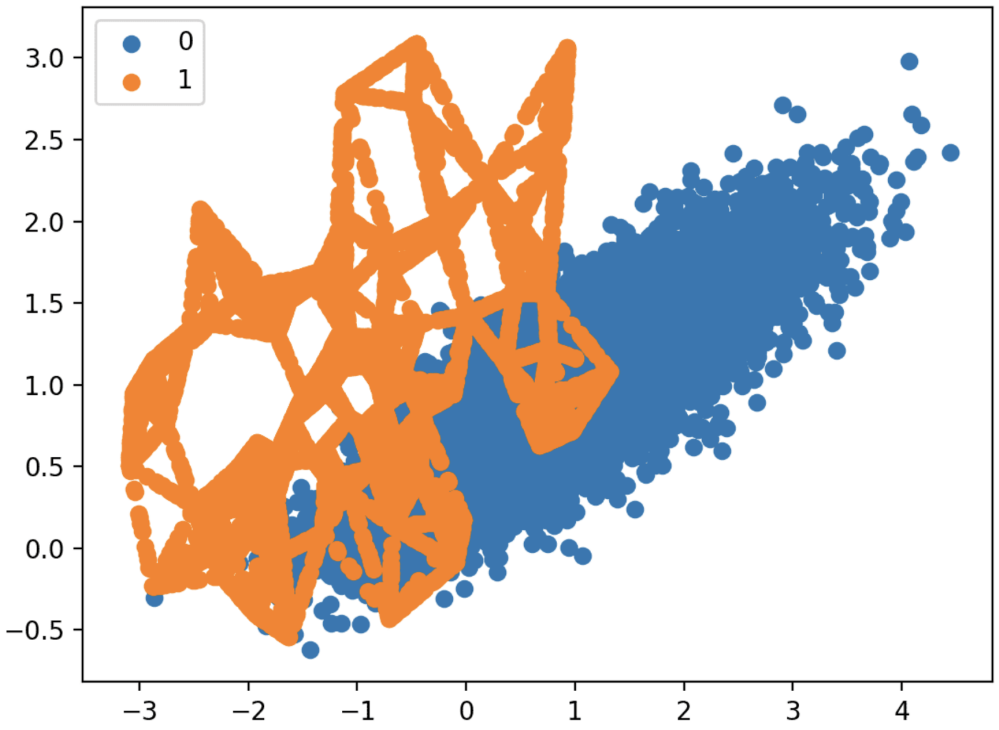

Figure 2: Distribution after SMOTE

By computing the SMOTE, we get the following result:

```
Before SMOTE
Counter({0: 58410, 1: 2406})

After SMOTE
Counter({0: 58410, 1: 58410})
```

Figure 3: Class distribution summary

Eliminating Collinearities

As we have a lot of features we will have to eliminate those with high collinearities. We have 369 features, around 100 of those (by inspection) are constant or quasi-constant. We will use the lasso path to eliminate variables for us.

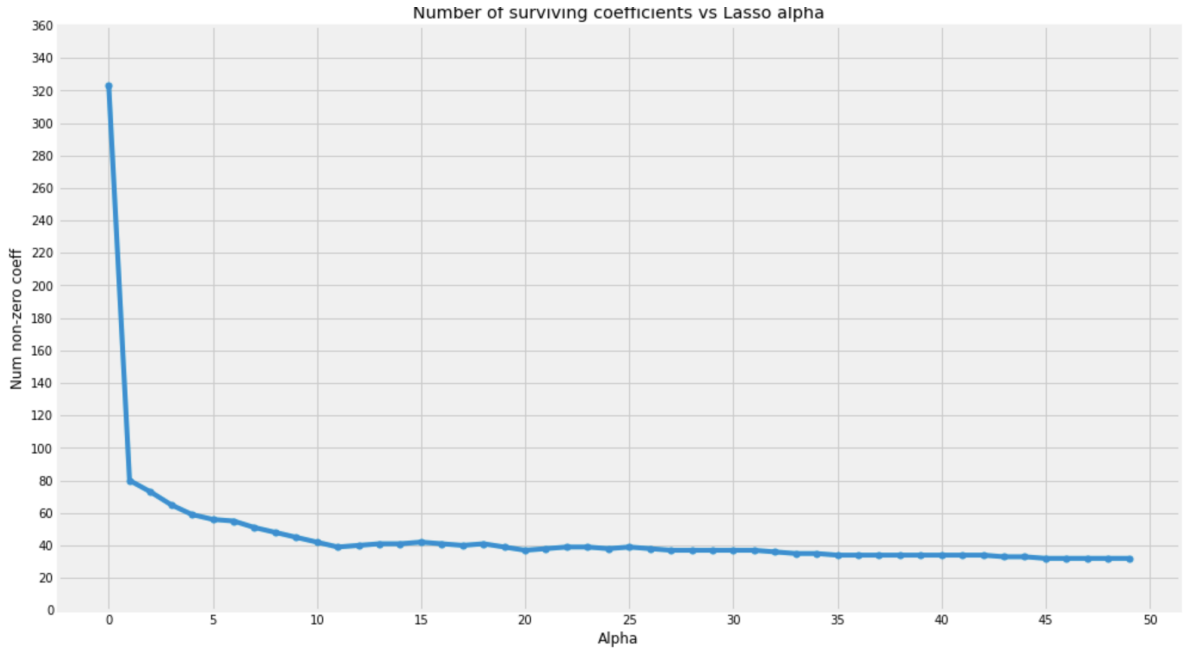

Figure 4: Number of surviving coefficients vs Lasso alpha

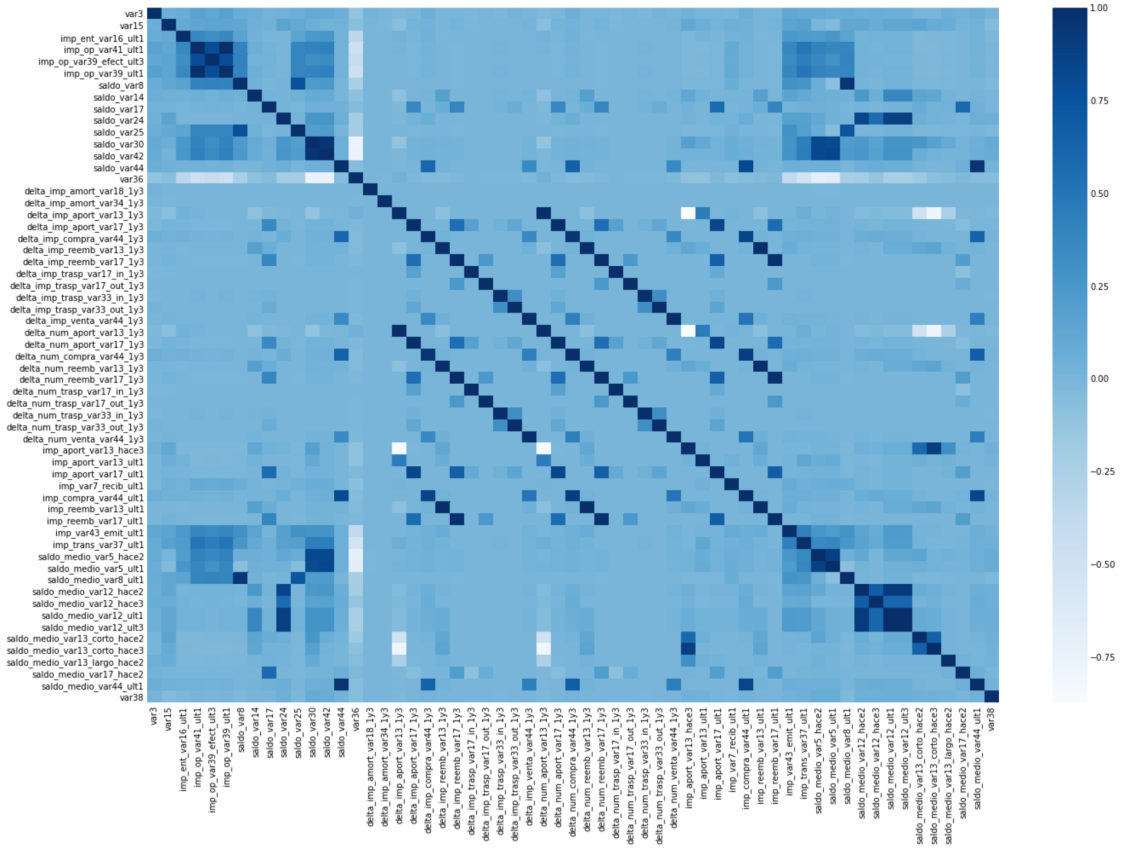

Figure 5: Correlation Heatmap with Lasso

Even though lasso has eliminated most of the collinear features, we still see some high correlations in the heatmap. We are going to keep them for now as we cannot always eliminate all correlations without losing a significant amount of information.
In order to verify that we are not losing a lot of information we fit non-trivial models. First on all data and second on lasso selected columns.

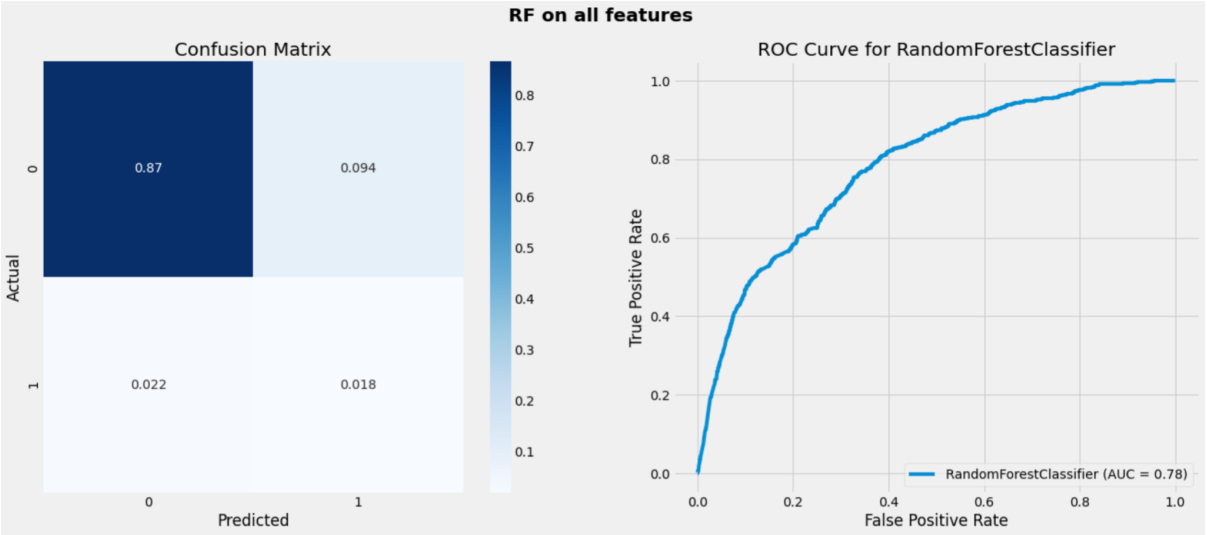

Figure 6: Confusion Matrix and ROC Curve for Random Forest Classifier when trained in all features

```
Train Accuracy: 0.919
  Val Accuracy: 0.885

=============== Classification Report ===============
              precision    recall  f1-score   support

           0       0.98      0.90      0.94     14602
           1       0.16      0.45      0.24       602

    accuracy                           0.88     15204
   macro avg       0.57      0.68      0.59     15204
weighted avg       0.94      0.88      0.91     15204
```

Figure 7: Classification Report for Random Forest Classifier when trained in all features

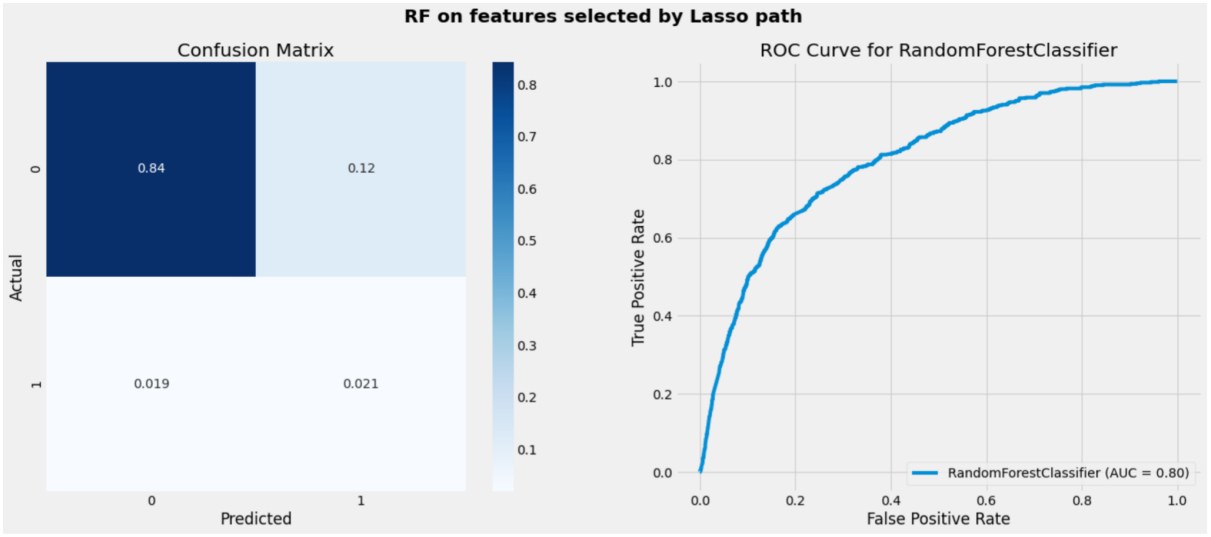

Figure 8: Confusion Matrix and ROC Curve for Random Forest Classifier when trained on features selected by lasso path

Model trained on Lasso features has accuracy a little lower than model trained on all features. However, the Lasso model outperforms other models in F1 score and AUC score. Also, models trained on Lasso features are definitely the better model.

Exploratory Data Analysis

Since we have a lot of variables, only the most important 8 variables are featured in the plot using SelectKBest.

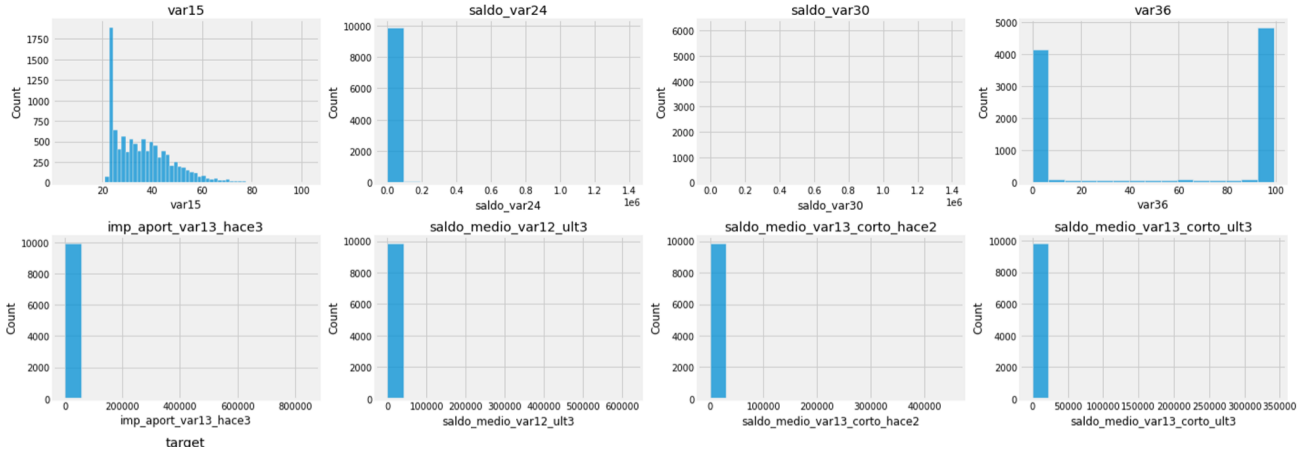

Figure 9: Distribution of important features

The subplot shows the distribution of most correlated variables. Majority of the variables consist of two unique values, and a high count of single values in most plots.

Base Model

While dealing with the base data, we found that we have a data imbalance: 96% of them 0s and only 4% is 1s.

```
Train Accuracy: 0.96
  Val Accuracy: 0.96
              precision    recall  f1-score   support

           0       0.96      1.00      0.98     14602
           1       0.00      0.00      0.00       602

    accuracy                           0.96     15204
   macro avg       0.48      0.50      0.49     15204
weighted avg       0.92      0.96      0.94     15204
```

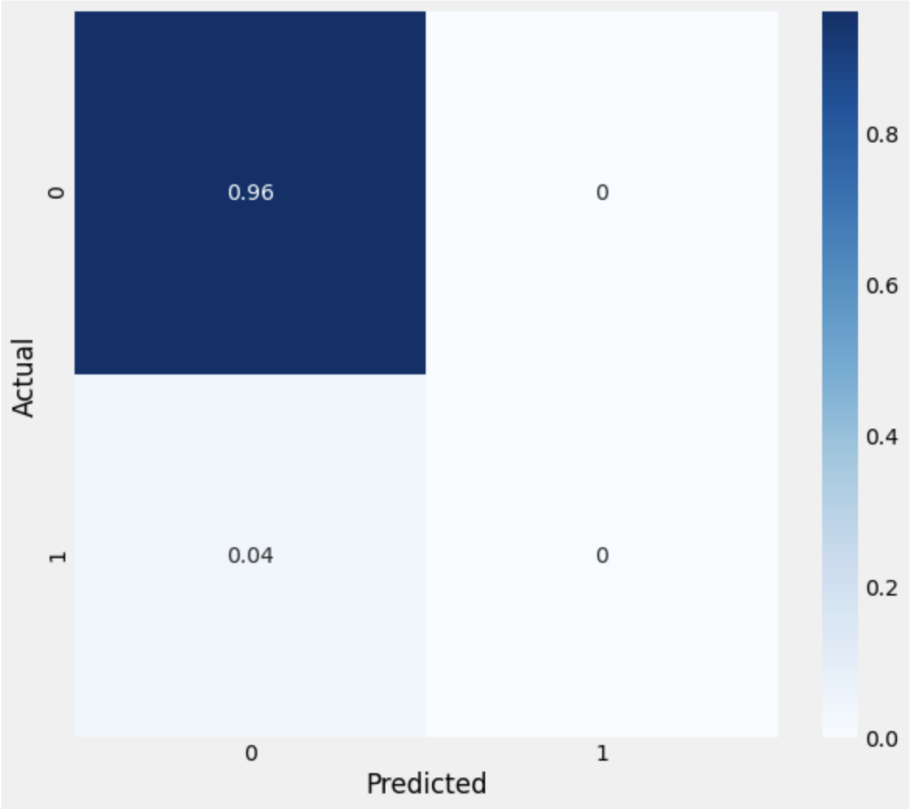

Figure 10: Correlation Matrix of Very Naive Model

```
Train Accuracy: 0.96
  Val Accuracy: 0.96
              precision    recall  f1-score   support

           0       0.96      1.00      0.98     14602
           1       0.00      0.00      0.00       602

    accuracy                           0.96     15204
   macro avg       0.48      0.50      0.49     15204
weighted avg       0.92      0.96      0.94     15204
```

Figure 11: Classification Report of Very Naive Model

Decision Tree

We computed a decision tree classifier with a max depth of 5, criterion of entropy and min samples leaf of 50.

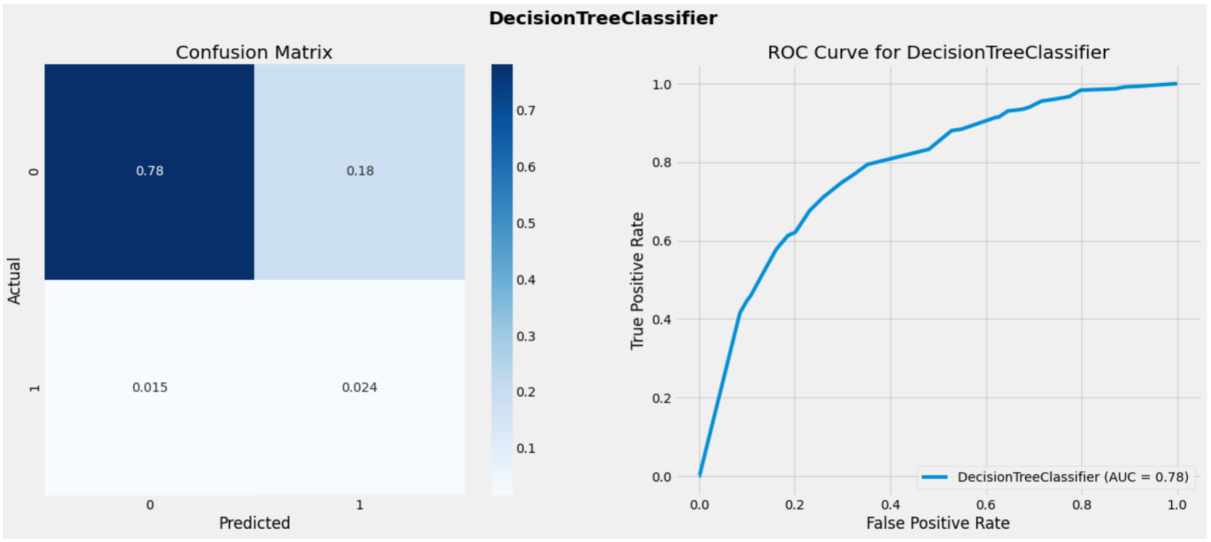

Figure 12: Correlation Matrix of Decision Tree Model

	precision	recall	f1-score	support
0	0.98	0.92	0.95	14602
1	0.19	0.45	0.27	602
accuracy			0.90	15204
macro avg	0.58	0.68	0.61	15204
weighted avg	0.94	0.90	0.92	15204

Figure 13: Classification Report of Decision Tree Classifier

Logistic Regression

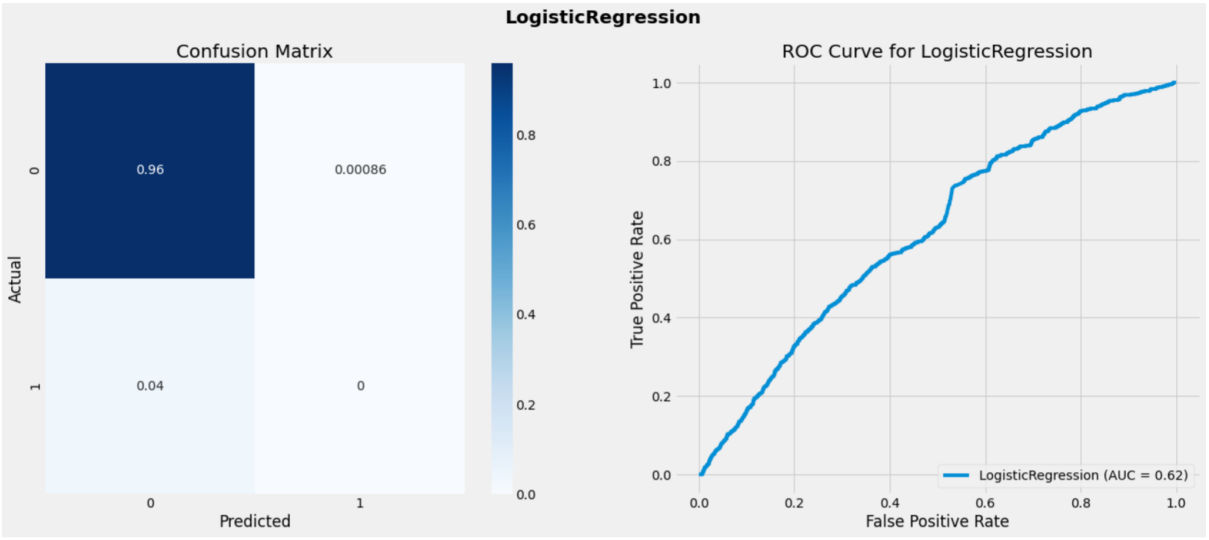

Figure 14: Confusion Matrix of Logistic Regression

```
Train Accuracy: 0.500
  Val Accuracy: 0.960

=============== Classification Report ===============
              precision    recall  f1-score   support

           0       0.96      1.00      0.98     14602
           1       0.00      0.00      0.00       602

    accuracy                           0.96     15204
   macro avg       0.48      0.50      0.49     15204
weighted avg       0.92      0.96      0.94     15204
```

Figure 15: Classification Report of Logistic Regression

From figure 12 until figure 15, we can derive that the Very Naive Model has accuracy of 96% and F1 score of 0.00. Decision Tree has accuracy of 82% and ROC-AUC score of 0.78. Logistic Regression has accuracy of 96% and ROC-AUC score of 0.61. And it performs unusually worse than DecisionTree. Very Naive outperforms DecisionTree in terms of accuracy. But as we have high data imbalance, accuracy is not the best metric to use. In addition, Decision Tree outperforms Naive Model in terms of F1 score.

Ensemble Models

Random Forest

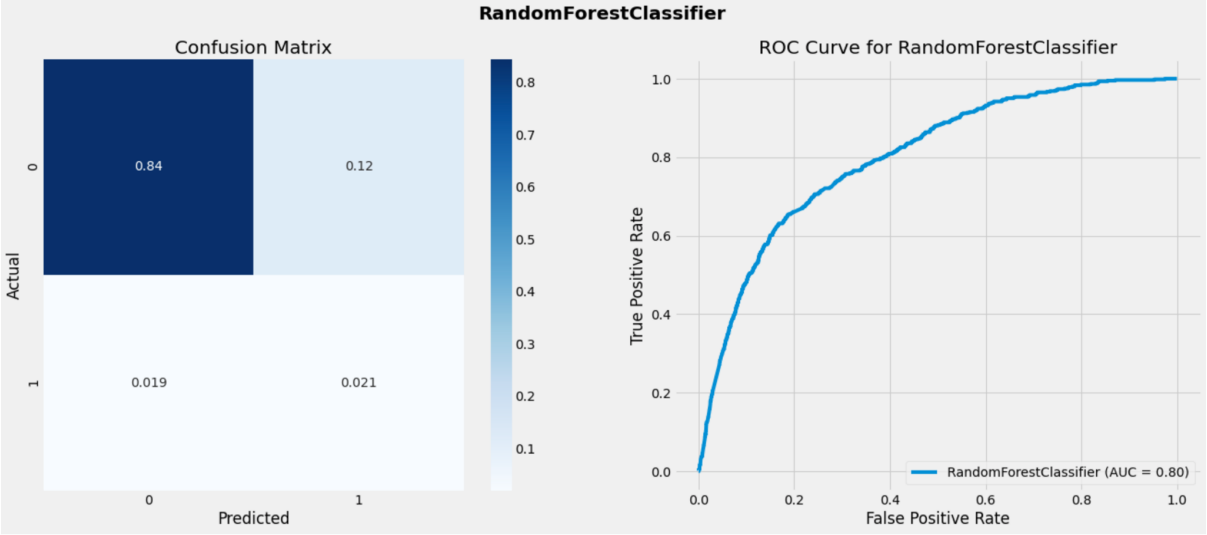

Figure 16: Confusion Matrix for Random Forest Classifier

```
Train Accuracy: 0.896
  Val Accuracy: 0.865

=============== Classification Report ===============
              precision    recall  f1-score   support

           0       0.98      0.88      0.93     14602
           1       0.15      0.53      0.24       602

    accuracy                           0.86     15204
   macro avg       0.57      0.70      0.58     15204
weighted avg       0.95      0.86      0.90     15204
```

Figure 17: Classification Report of Random Forest

Adaboost Classifier

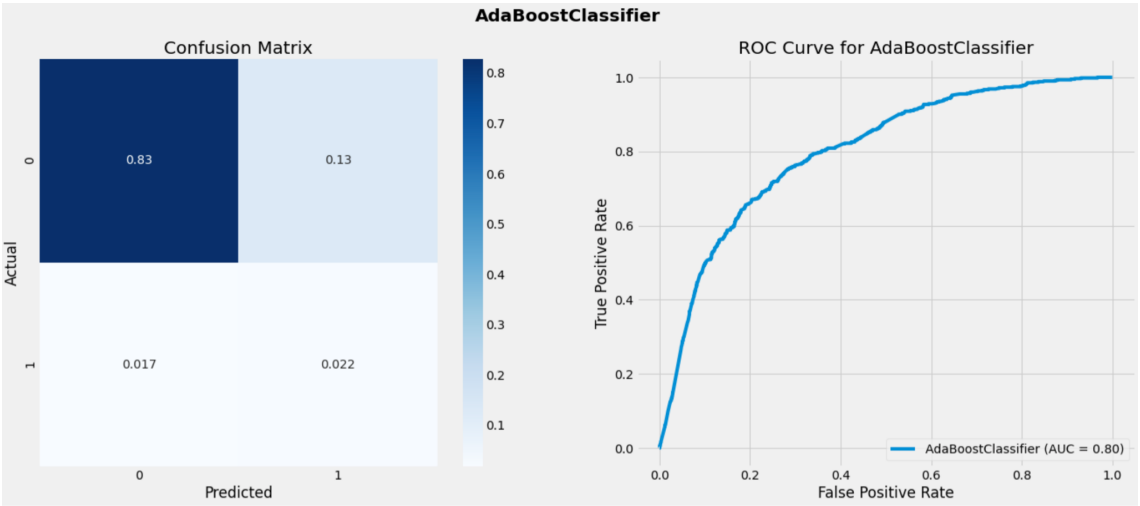

Figure 18: Confusion Matrix of AdaBoost Classifier

```
Train Accuracy: 0.876
  Val Accuracy: 0.850

=============== Classification Report ===============
              precision    recall  f1-score   support

           0       0.98      0.86      0.92     14602
           1       0.14      0.56      0.23       602

    accuracy                           0.85     15204
   macro avg       0.56      0.71      0.57     15204
weighted avg       0.95      0.85      0.89     15204
```

Figure 19: Classification Report of AdaBoost Classifier

Interpretation

Model Comparison

	Very Naive Model	Decision Tree	Random Forest	Adaboost	Logistic Regression	Best Score
Accuracy	0.960405	0.806301	0.864838	0.850434	0.95955	Very Naive Model
Precision	0.000000	0.119766	0.152226	0.144255	0.00000	Random Forest
Recall	0.000000	0.612957	0.528239	0.563123	0.00000	Decision Tree
F1 Score	0.000000	0.200380	0.236343	0.229675	0.00000	Random Forest

Figure 20: Model Comparison Table

In terms of the accuracy, the very naive model outperforms the other model. However, random forest is taking the lead precision and F1 score. In addition, the decision tree has the highest score in recall.

Feature Importance

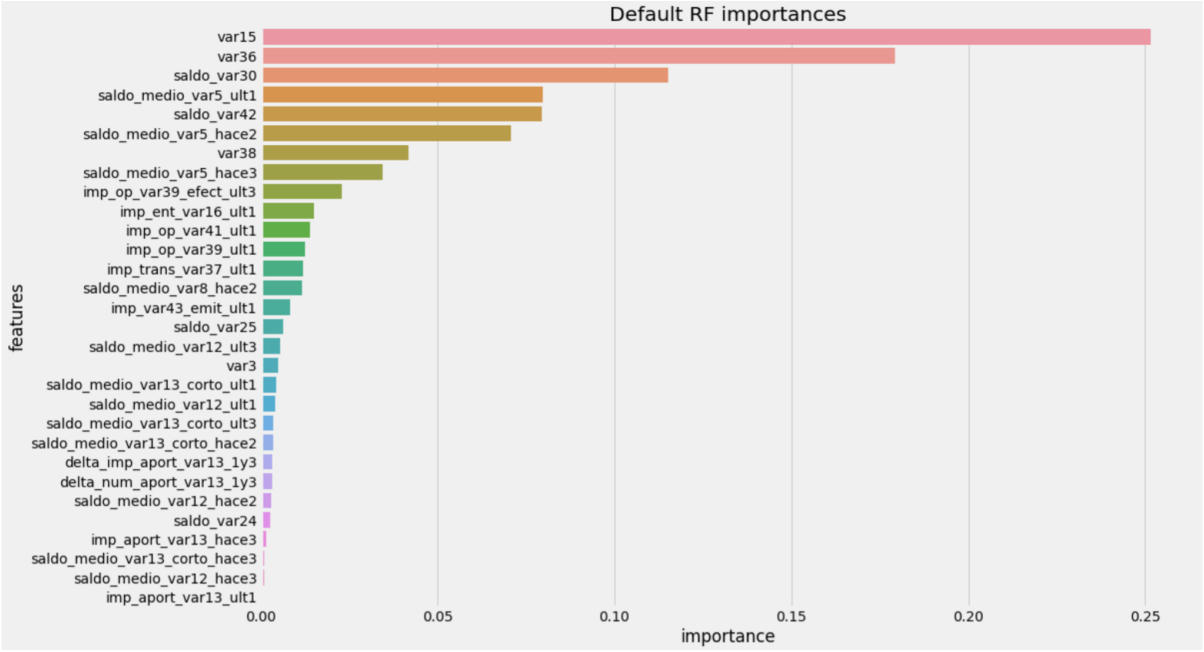

Figure 21: Feature importance in Random Forest Classifier

The graph indicated that the var15, var36 and saldo_var30 are the three most important features that are contributing the most to the random forest model.

Permutation Importance

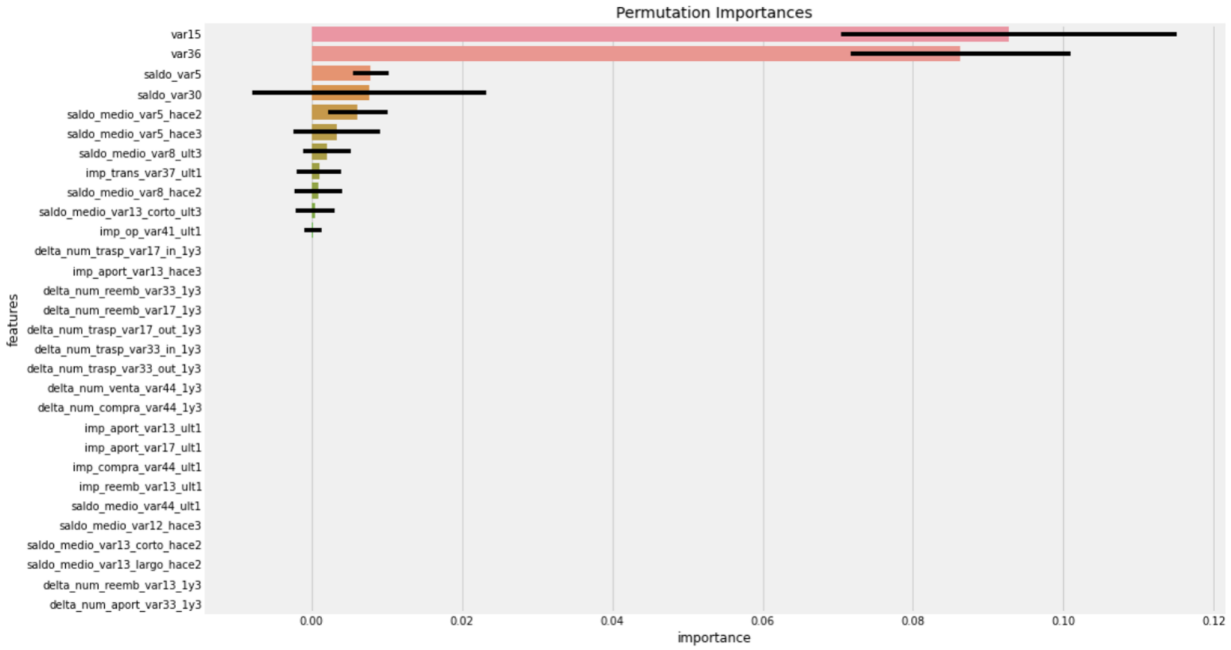

Figure 22: Permutation Importance in Random Forest Classifier

Similar trend can be observed in the graph. When variables are permuted to compute the permutation importance, var15, and var36 is providing the most value to the model. However, differences in the figure 21 and 22 can be seen in differences in position between the saldo_var5 and saldo_var30. After permuting, saldo_var5 is taking the lead than saldo_var30.

Partial Dependence Plots

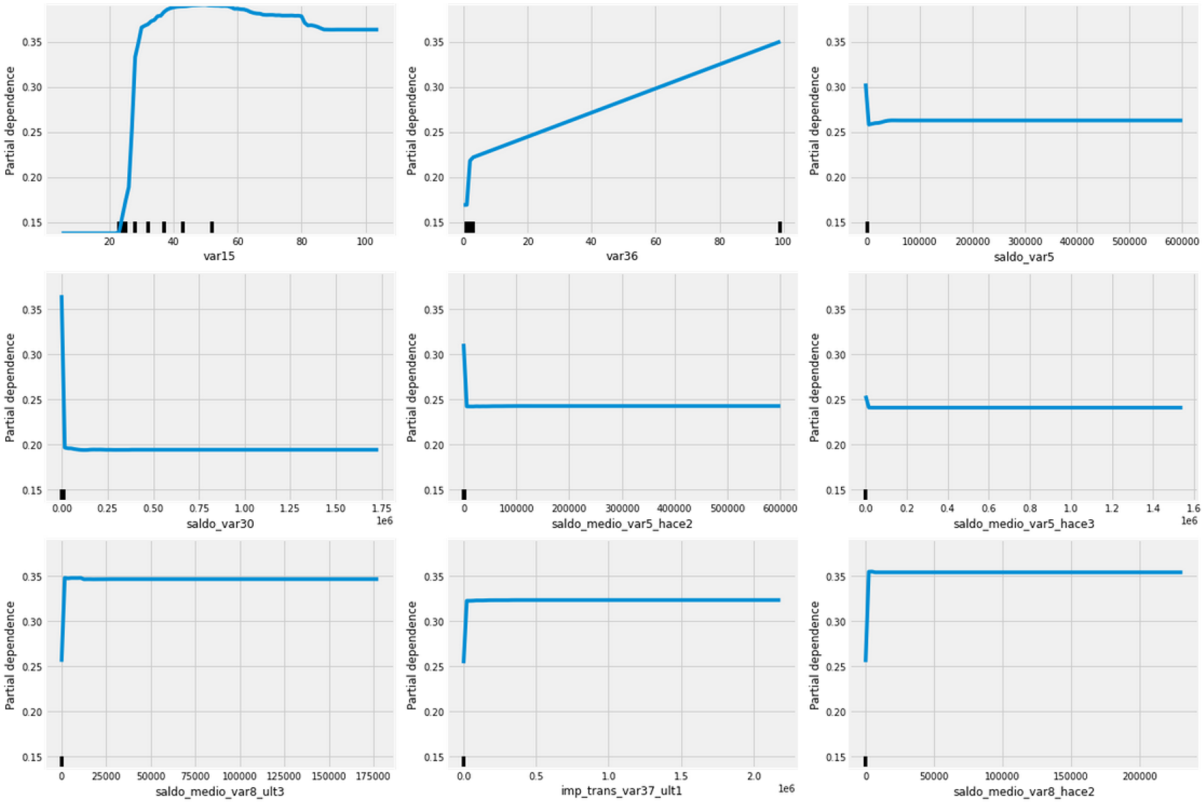

Figure 23: Partial Dependence Plot

Partial Dependence Plots help us to observe how predicted probability depends on individual variables. As seen in figure 23, the majority of the curves became flat after a certain value after 0. Also, we can see an increase in var36. However, for other variables, such as or var15 graph rises for a while and then plateaus.

Conclusion

In this project we studied the Santander Customer Satisfaction dataset to predict the unsatisfied customer. During the project, we observed a lot of features. We found that some of the features were either constant or quasi-constant.

To deal with data imbalance, we dealt with correlations using Lasso Path. When a non-trivial model was trained on around 70 features selected by Lasso path it outperformed other models trained on all 369 features.

During the modeling process, we beat the Baseline F-1 score by a small margin. In addition, we plotted various ROC curves and our best AUC score was 0.82.

To better interpret the results, we plotted features importances using RandomForest's default importances and Permutation Importances. We found that a lot of features which were tagged as important by RFI were found not very important in permutation importance's graph. Due to very high 0 inflation in almost all features Partial Dependence Plots may not be very accurate.

References

Jason Brownlee. (2021). SMOTE for Imbalanced Classification with Python. https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/