

PROJECT ON SANTANDER MISSION CUSTOMER PREDICTION

-By Yamini Peddireddi

Index

Chapter 1. Introduction.....	4-5
1.1. Data science	
1.2. Problem solving	
Chapter 2. Methodology.....	6-7
2.1. Problem Statement	
2.2. Steps to problem solving	
2.3. Challenges	
Chapter 3. Data understanding and analysis.....	8-9
3.1. Description of the data	
3.2. Features in the data	
3.3. Data cleaning	
3.4. Feature engineering and selection	
Chapter 4. Data visualization.....	10-12
4.1. Plots and insights	
Chapter 5. Handling imbalanced data.....	13-14
5.1. Techniques of handling data	
5.2. Applied method and balanced	
Chapter 6. Modelling.....	15-22
6.1. Splitting the data	
6.2. Logistic Regression	
6.3. Decision Tree Classifier	

6.4. Gaussian Naïve Bayes	
6.5. Random Forest Classifier	
6.6. XG Boost Classifier	
6.7. Stacking classifier	
Chapter 7. Model evaluation and selection.....	23-27
7.1. Model evaluation metrics	
7.2. Hyper parameter tuning	
7.3. Rebuilding the model and	
7.4. Model selection	
Chapter 8. Conclusion.....	28-29
8.1. Predictions on test data	
8.2. Deployment and feedback	
References.....	30-31

Chapter 1: Introduction

1.1. Data Science

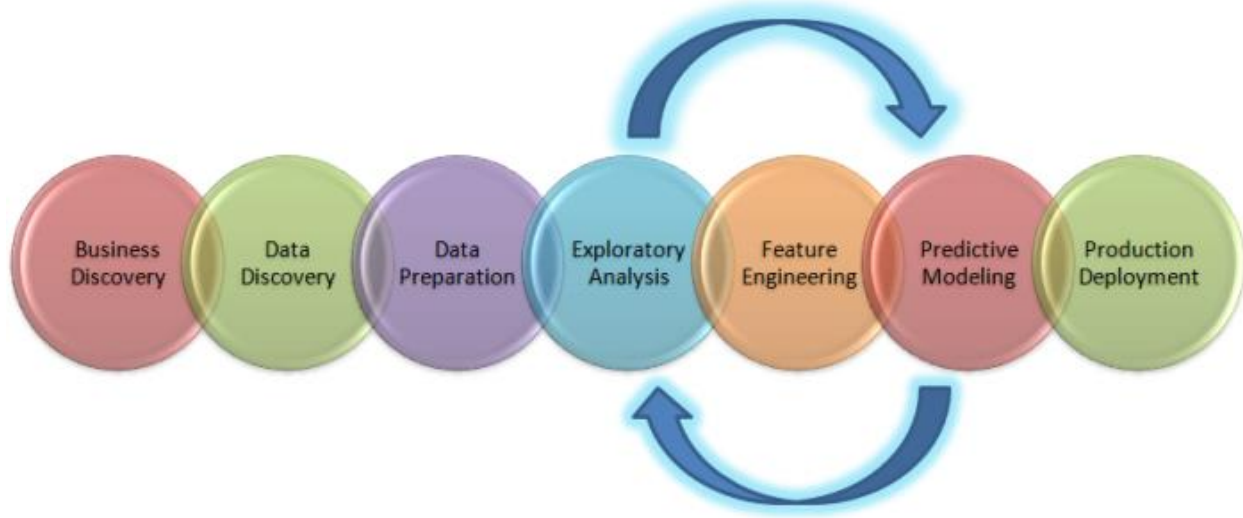
Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. Data science is related to data mining, machine learning and big data.

Data science is a "concept to unify statistics, data analysis and their related methods" in order to "understand and analyze actual phenomena" with data. It uses techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, domain knowledge and information science. Turing award winner Jim Gray imagined data science as a "fourth paradigm" of science (empirical, theoretical, computational and now data-driven) and asserted that "everything about science is changing because of the impact of information technology" and the data deluge.

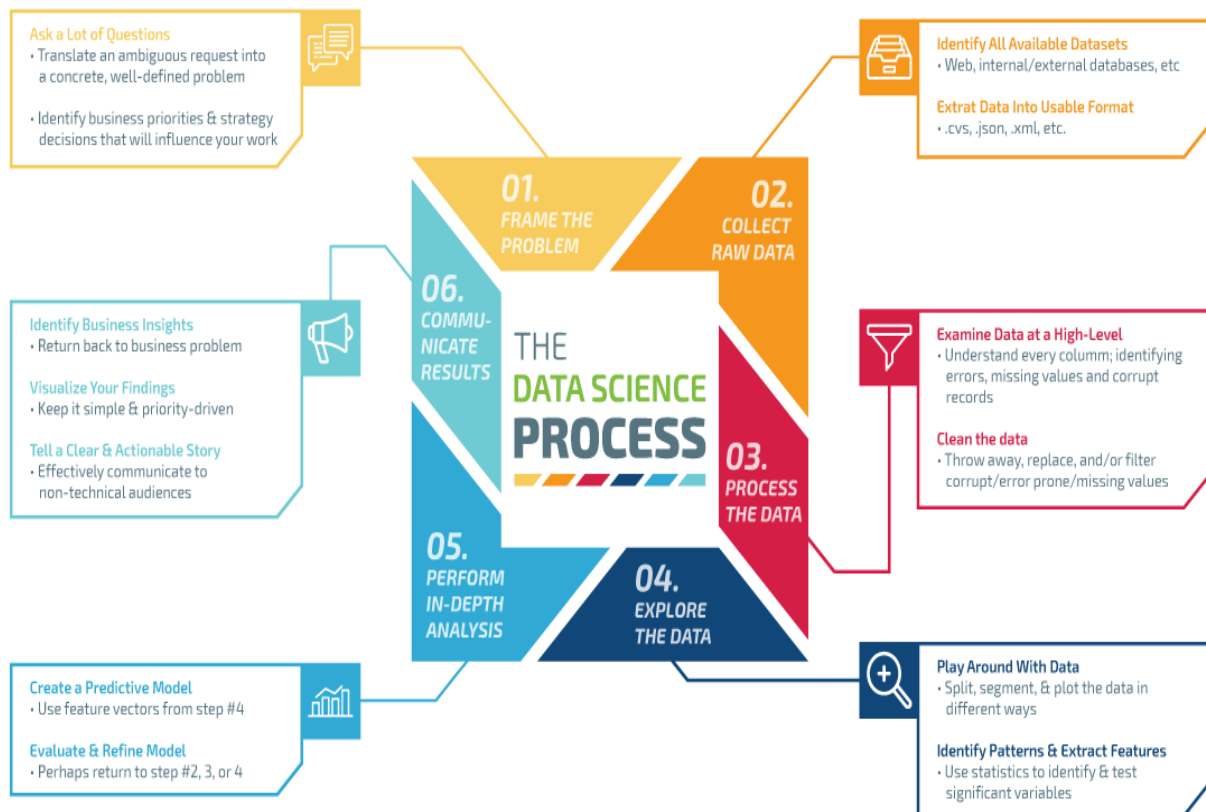
Big data is very quickly becoming a vital tool for businesses and companies of all sizes. The availability and interpretation of big data has altered the business models of old industries and enabled the creation of new ones. Data-driven businesses are worth \$1.2 trillion collectively in 2020, an increase from \$333 billion in the year 2015. Data scientists are responsible for breaking down big data into usable information and creating software and algorithms that help companies and organizations determine optimal operations. As big data continues to have a major impact on the world, data science does as well due to the close relationship between the two.



1.2. Problem solving



DATA SCIENCE DECONSTRUCTED



Chapter 2: Methodology

2.1. Problem Statement

At Santander, mission is to help people and businesses prosper. We are always looking for ways to help our customers understand their financial health and identify which products and services might help them achieve their monetary goals. Our data science team is continually challenging our machine learning algorithms, working with the global data science community to make sure we can more accurately identify new ways to solve our most common challenge, binary classification problems such as: is a customer satisfied? Will a customer buy this product? Can a customer pay this loan?

In this challenge, we need to identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

According to Epsilon research, 80% of customers are more likely to do business with you if you provide personalized service. Banking is no exception. Santander Group aims to go a step beyond recognizing that there is a need to provide a customer a financial service and intends to determine the amount or value of the customer's transaction. This means anticipating customer needs in a more concrete, but also simple and personal way. Santander Bank was founded in 1902 as Sovereign Bank, savings and loan in Wyomissing, Pennsylvania. The company's earliest customers were largely textile workers. Sovereign expanded rapidly during the savings and loan crisis of 1980s and 1990s, acquiring numerous other banks. It is based in Boston and its principal market is the north eastern United States. This paper focuses on Santander Bank, a large corporation focusing principally on the market in the northeast United States. The objective is to find an appropriate model to predict whether a client will make a transaction in the future. **Having this model in place can ensure that Santander can take proactive steps to improve a customer's happiness before they would take their business elsewhere.**

2.2. Steps to problem solving

According to the problem statement, it is clear that Santander bank's mission is help the people and business prosper with their products and services. So as there are many client data given from which there is a need to create or design a system through which the bank uses it for improving their service and business by serving the clients in a better way. Here the goal is to create a model to predict whether the customer is satisfied or not, whether they can repay the loan and whether the customers would make the future transaction or not so that they don't lose their clients.

The steps in create such a model or the steps in solving this problems are the following:

- Data collection and understanding
- Data cleaning
- Data visualization
- Feature engineering and selection
- Splitting the data
- Handling imbalanced dataset
- Modelling
- Hyper parameter tuning
- Rebuilding the model best parameters
- Model selection
- Prediction on the test set
- Deployment and feedback

2.3. Challenges

The challenges in creating the model are

- Data collection
- Data limitation
- Privacy and security
- Data manipulation
- Missing values or vague data
- Chance of misinterpretations

Chapter 3: Data understanding and analysis

3.1. Description of the data

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...	var_190	var_191	var_192	var_19
0	train_0	0	8.9255	-6.7863	11.9081	5.0930	11.4607	-9.2834	5.1187	18.6266	...	4.4354	3.9642	3.1364	1.691
1	train_1	0	11.5006	-4.1473	13.8588	5.3890	12.3622	7.0433	5.6208	16.5338	...	7.6421	7.7214	2.5837	10.951
2	train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	14.6155	...	2.9057	9.7905	1.6704	1.685
3	train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	14.9250	...	4.4666	4.7433	0.7178	1.421
4	train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	19.2514	...	-1.4905	9.5214	-0.1508	9.194

5 rows × 202 columns

3.2. Features in the data

The training data contains 200000 rows and 202 columns in which first feature is ID code containing only as reference to client names then next feature is target column which contains some of the client details of Yes and No kind represented as 0's and 1's. The other 200 columns are the feature names or independent features by which the target column is predicted.

3.3. Data cleaning

The data is further explored to remove if there are any missing values as those rows may not be useful in the prediction. So in the given dataset there are no such values exist. There are some outliers then there is not much deviation because of them so they are kept as it is and also the data is normally distributed.

3.4. Feature engineering and selection

As there are many features so there is a need to use the necessary features for model building instead of taking all the 200 features. So to achieve this dimensionality reduction techniques like PCA and SVD comes to place through which most useful features can be selected in the prediction.

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

From a simplified perspective, PCA transforms data linearly into new properties that are not correlated with each other. For ML, positioning PCA as feature extraction may allow us to explore its potential better than dimension reduction.

Geometrically speaking, principal components represent the directions of the data that explain a **maximal amount of variance**, that is to say, the lines that capture most information of the data. The relationship between variance and information here, is that, the larger the variance carried by a line, the larger the dispersion of the data points along it, and the larger the dispersion along a line, the more the information it has. To put all this simply, just think of principal components as new axes that provide the best angle to see and evaluate the data, so that the differences between the observations are better visible.

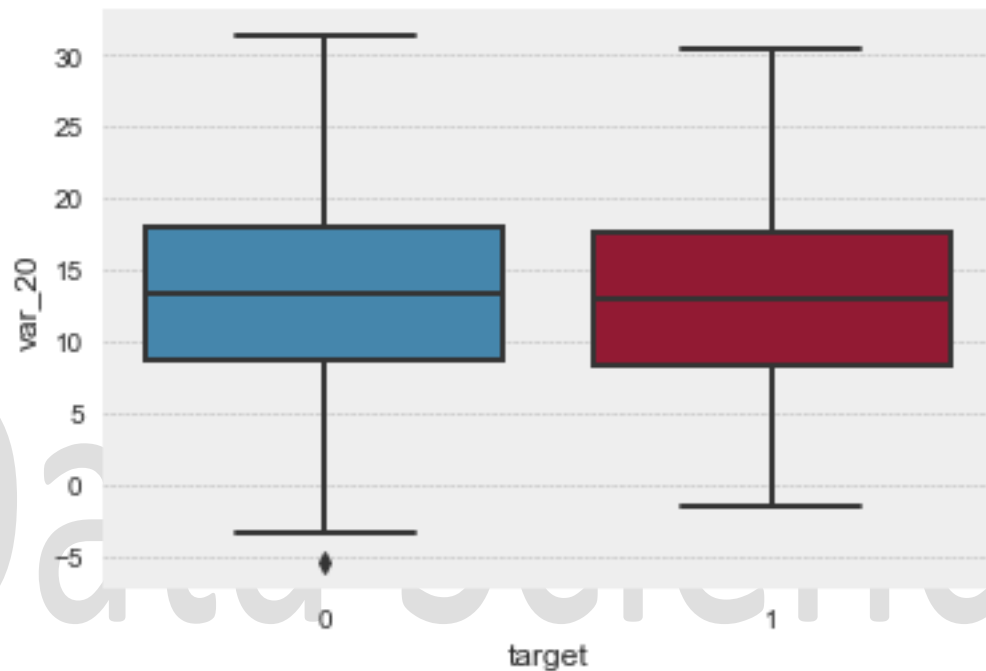
Singular value decomposition takes a rectangular matrix of gene expression data (defined as A , where A is a $n \times p$ matrix) in which the n rows represents the genes, and the p columns represents the experimental conditions. The SVD theorem states:

$$A_{n \times p} = U_{n \times n} S_{n \times p} V_{p \times p}^T$$

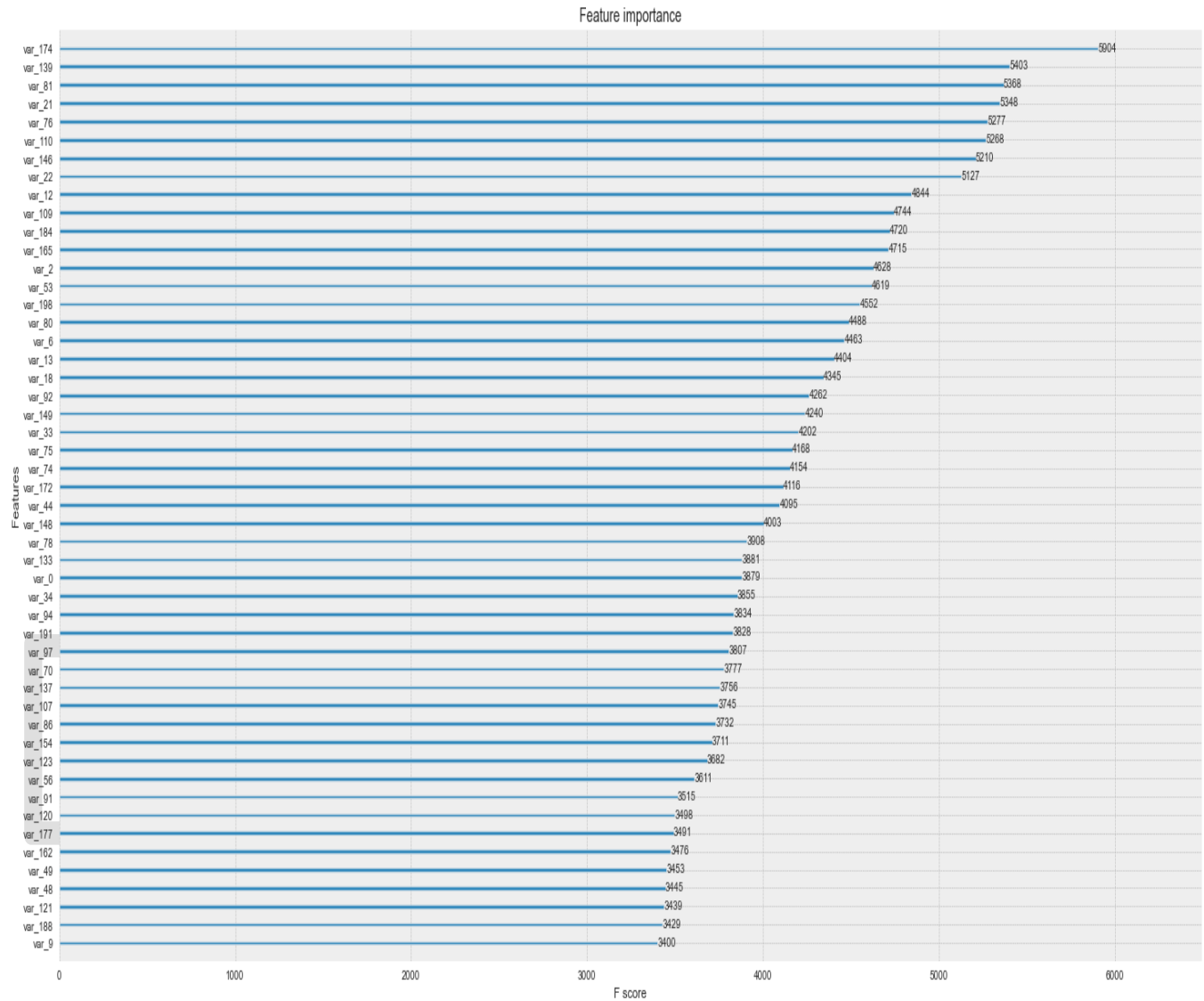
The difference between SVD and PCA is SVD gives you the whole nine-yard of diagonalizing a matrix into special matrices that are easy to manipulate and to analyze. It lay down the foundation to untangle data into independent components. PCA skips less significant components. Obviously, we can use SVD to find PCA by truncating the less important basis vectors in the original SVD matrix.

Chapter 4: Data visualization

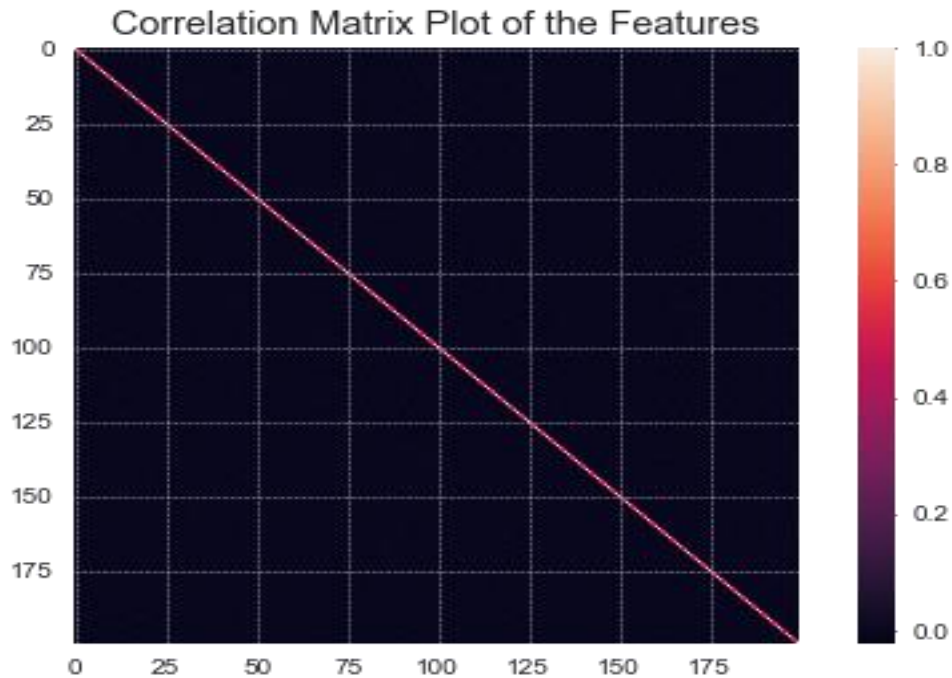
4.1. Plots and insights



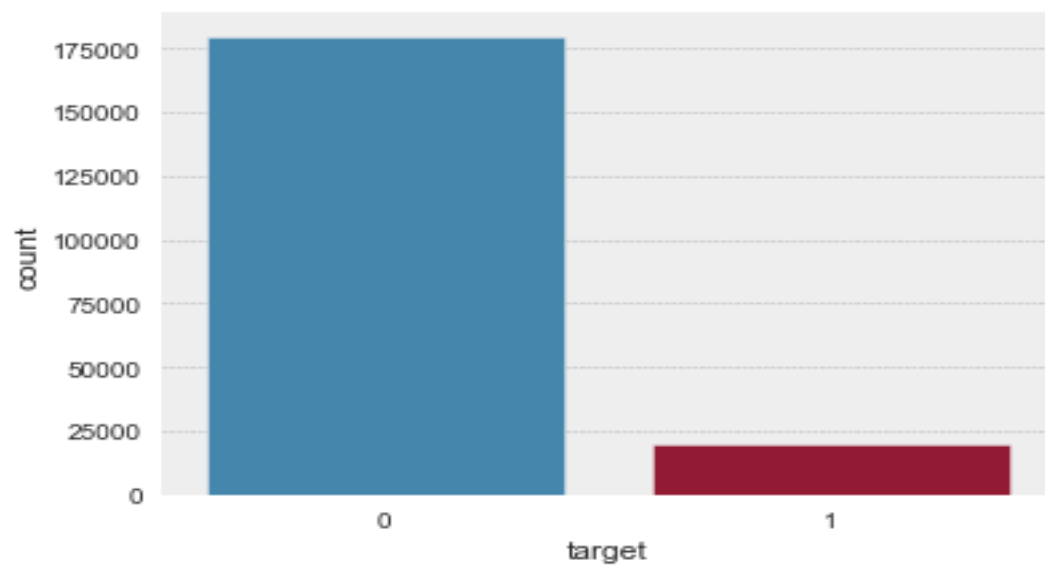
- ✚ From the above plot it can be seen that there are more Yes or 0 values which says the data is imbalanced and the data as it is cannot be used because there is a chance of the predictions more on 0 itself and also overfitting and under fitting can happen.
- ✚ From the above one it can also be seen that there are not many outliers and also the within the IQR (Inter-Quartile Range). This is the same for almost all the 200 features in the dataset. So there is very less standard deviation and mostly achieves normal distribution effortlessly.



- ✚ The above plot can be used for knowing the top 50 features out of the 130 features obtained after dimensionality reduction technique (SVD)
- ✚ This plot is obtained using the XGBoost Classifier which is the best performing model for this dataset in making the accurate predictions.



✚ From the above it can be seen that there is very less correlation or no correlation between the independent features.



- ✚ This Count plot above describes the data in the target column or the values of the dependent feature in which there are many features in the Yes or 0's values than the No or 1's values in the target column.
- ✚ It also says that almost 90% of the target columns contain 0 values than 1 value. This type of data in target column is called imbalanced dataset which can be further processed to balanced dataset for accurate predictions otherwise there is a chance of misclassification.

Chapter 5: Handling imbalanced data

5.1. Techniques of handling data

Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of misclassification of the minority class as compared to the majority class.

Evaluation of a classification algorithm performance is measured by the Confusion Matrix which contains information about the actual and the predicted class.

However, while working in an imbalanced domain accuracy is not an appropriate measure to evaluate model performance. For eg: A classifier which achieves an accuracy of 98 % with an event rate of 2 % is not accurate, if it classifies all instances as the majority class. And eliminates the 2 % minority class observations as noise.

Examples of imbalanced data

Thus, to sum it up, while trying to resolve specific business challenges with imbalanced data sets, the classifiers produced by standard machine learning algorithms might not give accurate results. Apart from fraudulent transactions, other examples of a common business problem with imbalanced dataset are:

- Datasets to identify customer churn where a vast majority of customers will continue using the service. Specifically, Telecommunication companies where Churn Rate is lower than 2 %.
- Data sets to identify rare diseases in medical diagnostics etc.
- Natural Disaster like Earthquakes

Resampling Techniques

Dealing with imbalanced datasets entails strategies such as improving classification algorithms or balancing classes in the training data (data preprocessing) before providing the data as input to the machine learning algorithm. The later technique is preferred as it has wider application.

The main objective of balancing classes is to either increasing the frequency of the minority class or decreasing the frequency of the majority class. This is done in order to obtain approximately the same number of instances for both the classes. Let us look at a few resampling techniques:

- Random Under Sampling
- Random Over Sampling
- Cluster-based Over Sampling
- Informed over sampling: SMOTE
- MSMOTE

5.2. Applied method and balanced

For the Santander dataset I have applied SMOTE (Synthetic minority oversampling technique). This technique is followed to avoid overfitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models.

- **Advantages**
 - Mitigates the problem of overfitting caused by random oversampling as synthetic examples are generated rather than replication of instances
 - No loss of useful information
- **Disadvantages**
 - While generating synthetic examples SMOTE does not take into consideration neighboring examples from other classes. This can result in increase in overlapping of classes and can introduce additional noise
 - SMOTE is not very effective for high dimensional data

Chapter 6: Modelling

6.1. Splitting the data

Now the data is clean, explored, engineered, selected and also balanced the dataset. Normally there are many methods to split the data and commonly used is train test split method. Here I used StratifiedKFold method for sampling for splitting the dataset into train and validation sets for better results.

For this I used the sklearn package from which Stratified K Fold is used.

The command used is the following:

```
skf = StratifiedKFold(n_splits=5,random_state=42,shuffle=True)
```

```
for t,v in skf.split(x,y):
```

```
    x_train,x_valid=x.iloc[t],x.iloc[v]
```

```
    y_train,y_valid=y.iloc[t],y.iloc[v]
```

Now the dataset is divided into train and test and also the data is balanced. So the data is ready for model building and below are the algorithms used in model building stage.

6.2. Logistic Regression


Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.


In simple words, the dependent variable is binary in nature having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no).

Mathematically, a logistic regression model predicts $P(Y=1)$ as a function of X . It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.

Logistic Regression Assumptions:

- 1) Before diving into the implementation of logistic regression, we must be aware of the following assumptions about the same –
- 2) In case of binary logistic regression, the target variables must be binary always and the desired outcome is represented by the factor level 1.
- 3) There should not be any multi-collinearity in the model, which means the independent variables must be independent of each other.
- 4) We must include meaningful variables in our model.
- 5) We should choose a large sample size for logistic regression

 The cross validation score on this dataset when Logistic model is applied is 79.87%

 From this it can be said that Logistic is performing good then not so high so let's check the score on other models by which it is possible to select the best model so that Accuracy, AUC score, TPR, TNR all tend to show good results.

6.3. Decision Tree Classifier

In this method a set of training examples is broken down into smaller and smaller subsets while at the same time an associated decision tree get incrementally

developed. At the end of the learning process, a decision tree covering the training set is returned.


- Using the decision algorithm, we start at the tree root and split the data on the feature that results in the largest information gain (IG) (reduction in uncertainty towards the final decision).
- In an iterative process, we can then repeat this splitting procedure at each child node until the leaves are pure. This means that the samples at each leaf node all belong to the same class.
- In practice, we may set a limit on the depth of the tree to prevent overfitting. We compromise on purity here somewhat as the final leaves may still have some impurity.


Advantages of Classification with Decision Trees:

- Inexpensive to construct.
- Extremely fast at classifying unknown records.
- Easy to interpret for small-sized trees
- Accuracy comparable to other classification techniques for many simple data sets.
- Excludes unimportant features.

Disadvantages of Classification with Decision Trees:

- Easy to overfit.
- Decision Boundary restricted to being parallel to attribute axes.
- Decision tree models are often biased toward splits on features having a large number of levels.
- Small changes in the training data can result in large changes to decision logic.
- Large trees can be difficult to interpret and the decisions they make may seem counter intuitive.

 The cross-validation score on the dataset with Decision Tree algorithm is 60.57%.

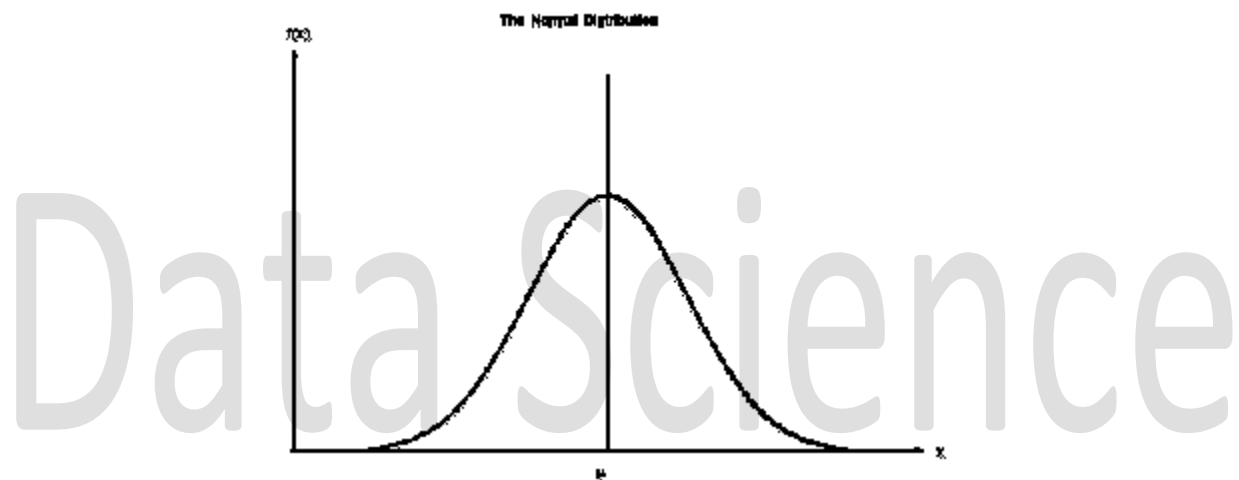
 This model gives very poor results so this model cannot be used for predictions for this dataset.

6.4. Gaussian Naïve Bayes

When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a gaussian distribution.

Since the way the values are present in the dataset changes, the formula for conditional probability changes to,

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp \left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2} \right)$$

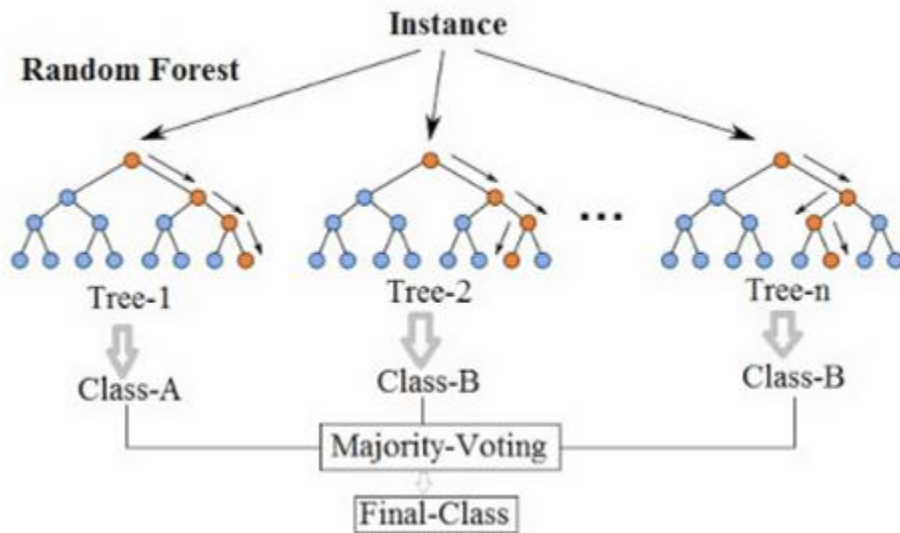


Naive Bayes algorithms are mostly used in sentiment analysis, spam filtering, and recommendation systems etc. They are fast and easy to implement but their biggest disadvantage is that the requirement of predictors to be independent. In most of the real life cases, the predictors are dependent, this hinders the performance of the classifier.

- ✚ The cross validation score with Gaussian Naïve Bayes is 86.55% with all 200 features and 84.32% on best features after Feature selection.
- ✚ As expected even though this is simple model Gaussian Naïve Bayes performs better on the dataset compared to Decision Tree and Logistic Regression.

6.5. Random Forest Classifier

It is an ensemble tree-based learning algorithm. The Random Forest Classifier is a set of decision trees from randomly selected subset of training set. It **aggregates the votes from different decision trees** to decide the final class of the test object.



Random forest is a great algorithm to train early in the model development process, to see how it performs. Its simplicity makes building a “bad” random forest a tough proposition.

The algorithm is also a great choice for anyone who needs to develop a model quickly. On top of that, it provides a pretty good indicator of the importance it assigns to your features.

Random forests are also very hard to beat performance wise. Of course, you can probably always find a model that can perform better, like a neural network for example, but these usually take more time to develop, though they can handle a lot of different feature types, like binary, categorical and numerical.

Overall, random forest is a (mostly) fast, simple and flexible tool, but not without some limitations.

Advantages and Disadvantages:

While it can be used for both classification and regression, Random Forest has an edge over other algorithms in the following ways,

- It is a robust and versatile algorithm.
- It can be used to handle missing values in the given data.
- It can be used to solve unsupervised ML problems.
- Understanding the algorithm is a cakewalk.
- Default hyperparameters used to give a good prediction.
- Solves the overfitting problem.
- It can be used as a feature selection tool.
- It handles high dimensional data well.

A few downsides go hand in hand with the advantages,

- It is computationally expensive.
- It is difficult to interpret.
- A large number of trees takes an ample of time.
- Creating predictions is often quite slow.

Random Forest's simplicity, diversity, robustness, and reliability trump the alternative algorithms available. It offers a lot of scope in improving the model's precision by tuning the hyperparameters and choosing the crucial features.

This article starts off by describing how a Decision Tree often acts as a stumbling block, and how a Random Forest classifier comes to the rescue.

- ✚ The cross validation score for this dataset on using Random Forest classifier is 90.80% on complete dataset with 200 features and 90.30% after feature selection using SVD.
- ✚ In either way the Random Forest works best and solves the problem with 90% accuracy.

6.6. XG Boost Classifier

Xgboost stands for eXtreme Gradient Boosting and is developed on the framework of gradient boosting.

“XGBoost used a more regularized model formalization to control over-fitting, which gives it better performance”.

The sequential ensemble methods, also known as “boosting”, creates a sequence of models that attempt to correct the mistakes of the models before them in the sequence. The first model is built on training data, the second model improves the first model, the third model improves the second, and so on.

The classifier models can be added until all the items in the training dataset is predicted correctly or a maximum number of classifier models are added. The optimal maximum number of classifier models to train can be determined using hyperparameter tuning.

eXtreme Gradient Boosting or XGBoost is a library of gradient boosting algorithms optimized for modern data science problems and tools. It leverages the techniques mentioned with boosting and comes wrapped in an easy to use library. Some of the major benefits of XGBoost are that it's highly scalable/parallelizable, quick to execute, and typically outperforms other algorithms.

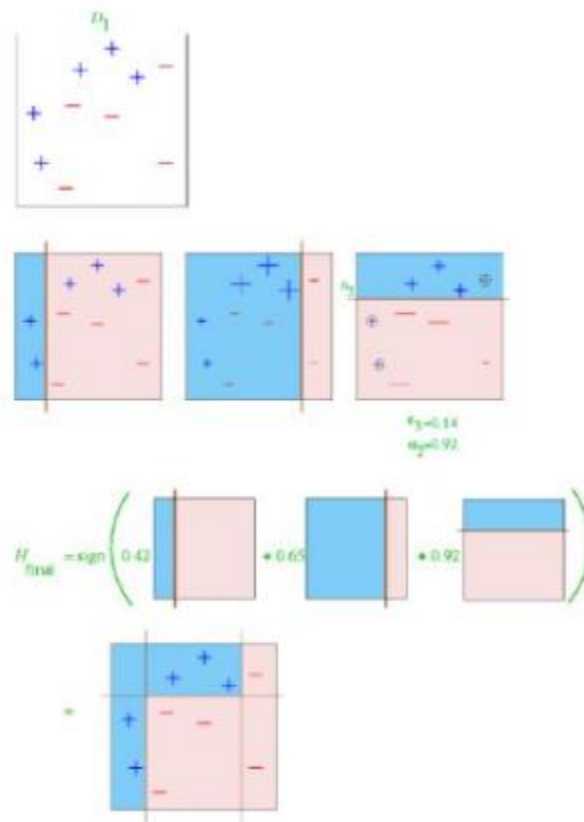


Figure 1: XG Boost in a picture

Pros:

- Extremely fast (parallel computation).
- Highly efficient.
- Versatile (Can be used for classification, regression or ranking).
- Can be used to extract variable importance.
- Do not require feature engineering (missing values imputation, scaling and normalization)

Cons:

- Only work with numeric features.
- Leads to overfitting if hyperparameters are not tuned properly.

✚ The cross-validation score for this dataset using XGBoosting Classifier is 86.09% for all the 200 features and 83.94% for best features obtained after feature selection using SVD.

✚ XGBoost also works well on this dataset and outperforms many algorithms in many datasets and as the accuracy seems to be 84% it can be considered a good model

6.7. Stacking classifier

Stacking is an ensemble learning technique to combine multiple classification models via a meta-classifier. The individual classification models are trained based on the complete training set; then, the meta-classifier is fitted based on the outputs -- meta-features -- of the individual classification models in the ensemble. The meta-classifier can either be trained on the predicted class labels or probabilities from the ensemble. Stacking is another ensemble model, where a new model is trained from the combined predictions of two (or more) previous model. The predictions from the models are used as inputs for each sequential layer, and combined to form a new set of predictions. These can be used on additional layers, or the process can stop here with a final result. In the example below I'm only going to use one layer for simplicity.

Ensemble stacking can be referred to as blending, because all the numbers are blended to produce a prediction or classification.

Keep in mind just by adding layers and more models to your stacking algorithm, does not mean you'll get a better predictor.

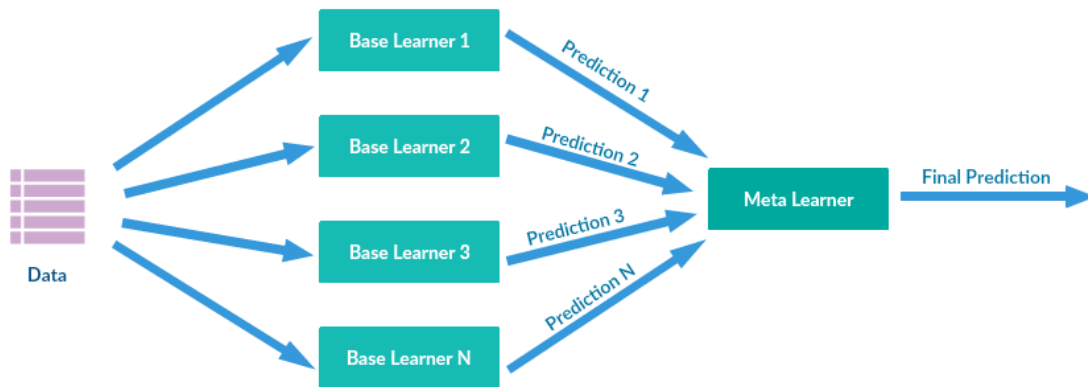


Figure 2: Stacking classifier in a picture

✚ The cross validation on using Stacking Classifier with models Gaussian Naïve bayes, Stochastic gradient descent(SGD)classifier and Random Forest is 75.76%

Chapter 7: Model evaluation and selection

7.1. Model evaluation metrics

The evaluation metrics for evaluating the performance of a machine learning model, which is an integral component of any data science project. It aims to estimate the generalization accuracy of a model on the future (unseen/out-of-sample) data.

A confusion matrix is a matrix representation of the prediction results of any binary testing that is often used to describe the performance of the classification model (or “classifier”) on a set of test data for which the true values are known.

A confusion matrix is a performance measurement method for Machine learning classification. It helps you to know the performance of the classification model on

a set of test data for that the true values and false are known. It helps us find out, how many times our model has given correct or wrong output and of what type. Hence, it is a very important tool for evaluating classification models.

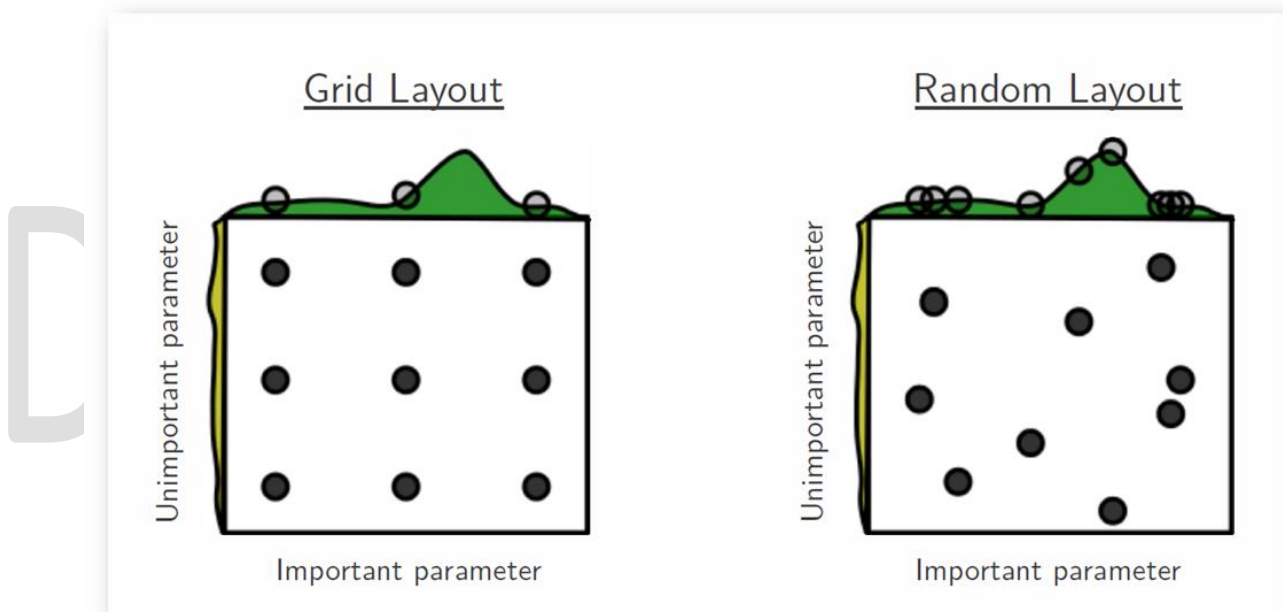
		Predicted Values	
		Negative	Positive
Actual Values	Negative	TN True Negative	FP False positive
	Positive	FN False Negative	TP True Positive

Understanding how well a machine learning model is going to perform on unseen data is the ultimate purpose behind working with these evaluation metrics. Metrics like accuracy, precision, recall are good ways to evaluate classification models for balanced datasets, but if the data is imbalanced and there's a class disparity, then other methods like ROC/AUC, Gini coefficient perform better in evaluating the model performance.

- ✚ As described evaluation of classification problems can be made based on accuracy score and from confusion matrix many of the metrics can be obtained like accuracy, precision, recall, sensitivity, specificity, f1 score etc. Classification report describes well about all these metrics.

7.2. Hyper parameter tuning

- ✚ There are many techniques through which hyperparameter tuning can be performed like Grid Search CV, Randomized Search CV and Bayesian Optimization etc
- ✚ Now I have used Grid Search CV for tuning the parameters and getting the best parameters and reconstructed the model with those parameters and have check the grid best score and cross validation score which gives accuracy score using cross validation. This score is the average of the accuracy scores made after n number of splits and ensures more accurate predictions.



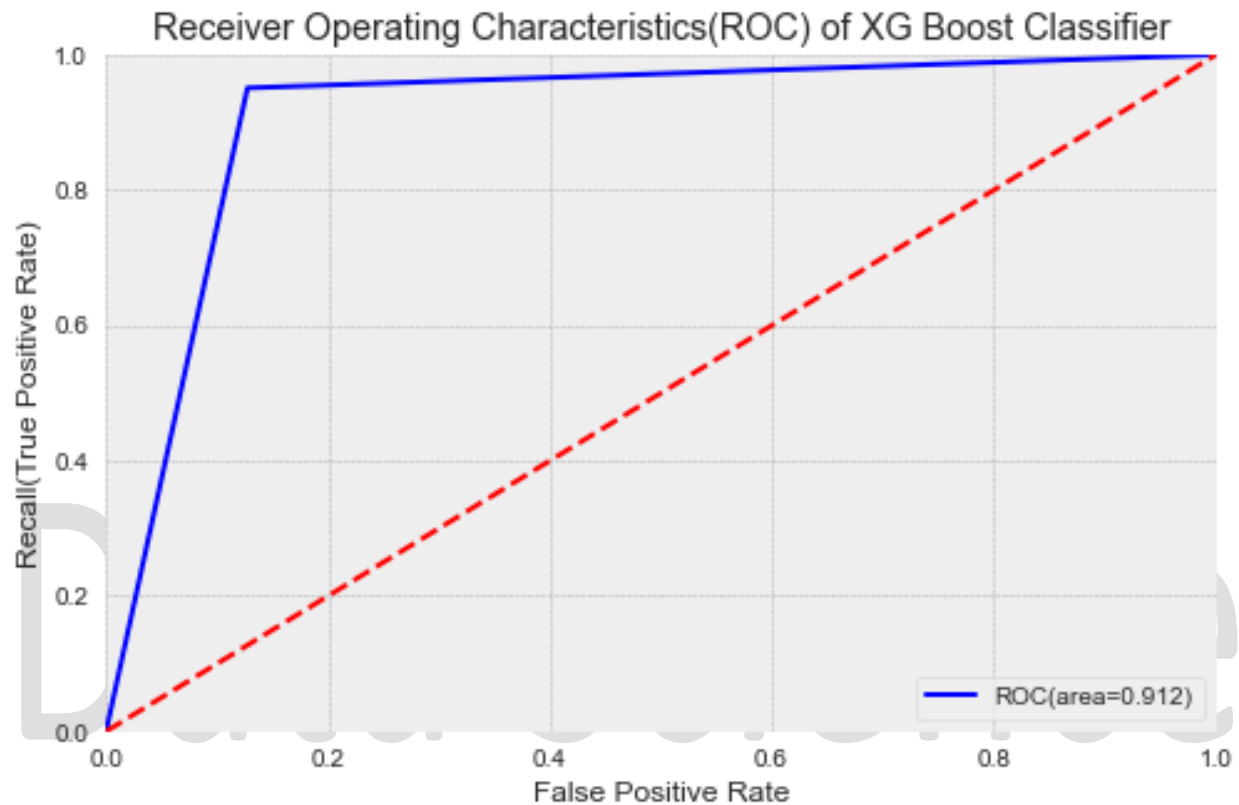
7.3. Rebuilding the model

- ✚ The models used for reconstruction with best parameters obtained after hyperparameter tuning are Logistic Regression, Gaussian Naïve Bayes, Random Forest, XG Boost and Stacking Classifiers

7.4. Model selection

- ✚ I used all the models used in grid search construction and after rebuilding with all those models with best parameters. I again calculated the evaluation metrics for all these models and found that XG Boost outperforms all the models after hyper parameter tuning and even grid score is found to be

98.74% and cross validation score is 87.51%(for 130 imp features) and 89% for complete dataset(with all the 200 features).



	precision	recall	f1-score	support
0	0.95	0.87	0.91	35980
1	0.88	0.95	0.92	35980
accuracy			0.91	71960
macro avg	0.91	0.91	0.91	71960
weighted avg	0.91	0.91	0.91	71960

<u>Models</u>	<u>All features</u>	<u>Imp features</u>	<u>After tuning</u>
Logistic Regression	79.87%		75.12%
Decision Tree	60.57%		
Gaussian NB	86.35%	84.32%	84.31%
Random Forest	90.80%	89.90%	82.19%
XG Boost	86.09%	83.94%	89.05%
Stacking		75.76%	82.19%

- ✚ All the accuracy score or the score by cross validation are listed in the above table and many other predictions are taken into were made while model building and the top 3 models are highlighted from which XG Boost Classifier outperforms all other models from all the constructed algorithms.
- ✚ As XG Boost is best model for this dataset according to the above scores then searching for some other models which can beat this score will be Light gbm. If applied there is a chance of slight improvement from the studies found Light gbm gives more accurate predictions than XG Boost. So, I have taken that model also for further improvements on these scores. Also, Light gbm takes less computation time and accuracy improves.
- ✚ As XG Boost is giving almost 90% of accuracy so this model is better than any other algorithms and from the study I can say Light gbm also does the equivalent task as XG Boost.
- ✚ Now I am taking XG Boost as the best model for this project on Santander customer predictions and improving their business and helping them in achieving their mission of helping people and business prosper in the future.

Chapter 8: Conclusion

8.1. Predictions on test data

```
In [204]: Sa_test_pred['target']=test_pred
```

```
In [206]: Sa_test_pred.shape
```

```
Out[206]: (200000, 202)
```

```
In [207]: Sa_test_pred.head()
```

```
Out[207]:
```

var_2	var_3	var_4	var_5	var_6	var_7	var_8	...	var_191	var_192	var_193	var_194	var_195	var_196	var_197	var_198	var_199	target
12.9536	9.4292	11.4327	-2.3805	5.8493	18.2675	2.1337	...	11.8495	-1.4300	2.4508	13.7112	2.4669	4.3654	10.7200	15.4722	-8.7197	0
11.3047	5.1858	9.1974	-4.0117	6.0196	18.6316	-4.4131	...	8.8349	0.9403	10.1282	15.5765	0.4773	-1.4852	9.8714	19.1293	-20.9760	0
10.1407	7.0479	0.2628	9.8052	4.8950	20.2537	1.5233	...	10.9935	1.9803	2.1800	12.9813	2.1281	-7.1086	7.0618	19.8956	-23.1794	0
12.0220	6.5749	8.8458	3.1744	4.9397	20.5660	3.3755	...	9.0766	1.6580	3.5813	15.1874	3.1656	3.9567	9.2295	13.0168	-4.2108	1
14.1295	7.7506	9.1035	-8.5848	6.8595	10.6048	2.9890	...	9.1723	1.2835	3.3778	19.5542	-0.2860	-5.1612	7.2882	13.9260	-9.1846	1

8.2. Deployment and feedback

- After successful predictions it is required to send it to the required place for deployment where the model gets used.
- For that it is not possible or recommended to send just the python or R file it is required in some form of webpage or API through which clients or users can use the prepared model for their company or individual purpose.
- The deployment can be done through many ways like from AWS, Google, Heroku, Flask and many more cloud environments.
- From this cloud sources we create a project in that environments for making webpage with our model and other required resources.
- Some of the files made through this are app.yaml, main.py(model file), requirements.txt file, python.py file , train.py file etc.

- ✚ Before preparing these files in python we save our model through joblib package and save the model in pickle format and load this model in another python file and make necessary use of it.
- ✚ After making all the files we open the prompt or through Flask(Pycharm env) we give run the model and
- ✚ write the command 'pip requirements.txt'
- ✚ Now you will see the https://localhostxxxx or some IP address through which it successfully creates the webpage in which the model is implemented which can be used by client or users whoever requires this information. In this case Cab owners or Transportation providers use this for their Cab Fare prediction.
- ✚ I have added some links for this deployment or example for deployment from some sources in references section.
- ✚ This the end to end project made through Machine Learning algorithms for solving the real time problems and this project is particularly made for the purpose of Santander mission's goal and for their customer prediction for increasing the satisfaction and maintaining the loyalty of the customers.

Data Science

References:

https://en.wikipedia.org/wiki/Data_science

https://medium.com/@jonathan_hui/machine-learning-singular-value-decomposition-svd-principal-component-analysis-pca-1d45e885e491#:~:text=What%20is%20the%20difference%20between,PCA%20skips%20less%20significant%20components.

<https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/>

https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_logistic_regression.htm#:~:text=Logistic%20regression%20is%20a%20supervised,be%20only%20two%20possible%20classes.&text=Mathematically%2C%20a%20logistic%20regression%20model,as%20a%20function%20of%20X.

<https://www.geeksforgeeks.org/naive-bayes-classifiers/>

<https://iq.opengenus.org/gaussian-naive-bayes/>

<https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac>

<https://towardsdatascience.com/random-forest-classification-and-its-implementation-d5d840dbead0>

<https://builtin.com/data-science/random-forest-algorithm>

<https://medium.com/machine-learning-101/chapter-5-random-forest-classifier-56dc7425c3e1>

<https://blog.paperspace.com/random-forests/>

<https://www.dataversity.net/machine-learning-algorithms-introduction-random-forests/>

<https://www.datacamp.com/community/tutorials/xgboost-in-python>

<https://blog.quantinsti.com/xgboost-python/>

<http://theprofessionalspoint.blogspot.com/2019/03/advantages-of-xgboost-algorithm-in.html>

<https://medium.com/@rrfd/boosting-bagging-and-stacking-ensemble-methods-with-sklearn-and-mlens-a455c0c982de>

http://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/

<https://www.kdnuggets.com/2020/05/model-evaluation-metrics-machine-learning.html>

<https://www.jeremyjordan.me/hyperparameter-tuning/>

<https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/>

<https://aws.amazon.com/getting-started/hands-on/build-train-deploy-machine-learning-model-sagemaker/>

<https://www.deploymachinelearning.com/>

<https://www.analyticsvidhya.com/blog/2020/04/how-to-deploy-machine-learning-model-flask/>

<https://medium.com/datadriveninvestor/deploy-your-machine-learning-model-using-flask-made-easy-now-635d2f12c50c>

<https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5>

Data Science