Santander Customer Transaction Prediction

PROBLEM STATEMENT :

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

PROBLEM DESCRIPTION:

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?

We Already Have Past Data And From The Given Problem It Is Clear

That Our Output Is Classification….

So It Falls Under Supervised Machine Learning

We Train The Model With Past Data And When New Data Is Given We Predict The Outcome

DATA:

Given data contains numeric feature variables, the binary target column, and a string ID_code column. The task is to predict the value of target column in the test set..

ID_code (string);

Target;

200 numerical variables, named from var_0 to var_199;

It has 201 predictors or independent variables and 1 target variable ‘target’

METHODOLOGY

Missing Value Analysis:

Missing values are which, where the values are missing in an observation in the dataset. It can occur due to human errors, individuals refusing to answer while surveying, optional box in questionnaire.

Missing data mechanism is divided into 3 categories as below :

Missing Completely at Random **(MCAR),** means there is no relationship between the missingness of the data and any values, observed or missing. Those missing data points are a random subset of the data. There is nothing systematic going on that makes some data more likely to be missing than others.

Missing at Random **(MAR),** means there is a systematic relationship between the propensity of missing values and the observed data, but not the missing data. Whether an observation is missing has nothing to do with the missing values, but it does have to do with the values of an individual's observed variables. So, for example, if men are more likely to tell you their weight than women, weight is MAR.

Missing Not at Random **(MNAR),** means there is a relationship between the propensity of a value to be missing and its values. This is a case where the people with the lowest education are missing on education or the sickest people are most likely to drop out of the study. MNAR is called "non-ignorable" because the missing data mechanism itself must be modelled as we deal with the missing data.

Usually we only consider those variables for missing value imputation whose missing values is less than 30%, if it above this we will drop that variable in our analysis as imputing missing values which are more than 30% doesn't make any sense and the information would also be insensible to consider.

IN OUR GIVEN DATASET WE ARE NOT HAVING ANY MISSING VALUES….SO WE ARE NOT PROCEEDING WITH THIS STEP TO IMPUTE ANY MISSING VALUES.

DATA VISUALIZATION:

Fig: Density plots of features

We can observe that there is a considerable number of features with significant different distribution for the two target values.

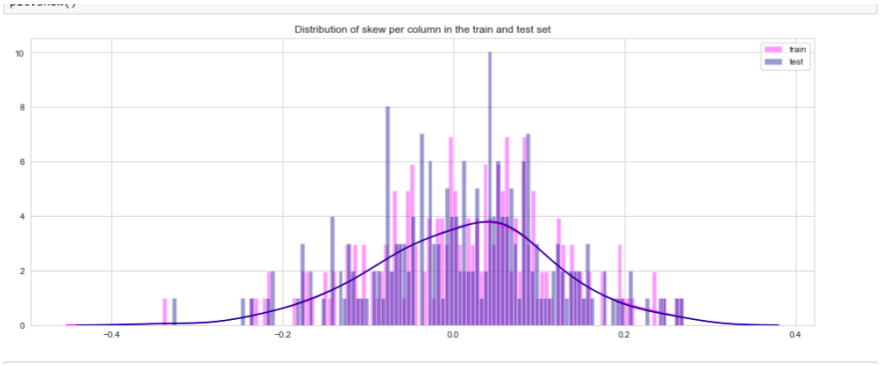


Fig: Distribution of skew per column in the train and test set

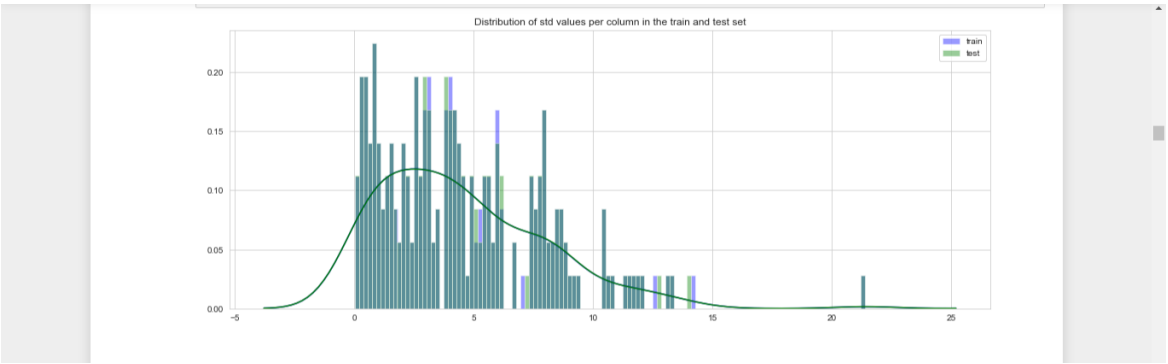


Fig: Distribution of std values per column in the train and test set

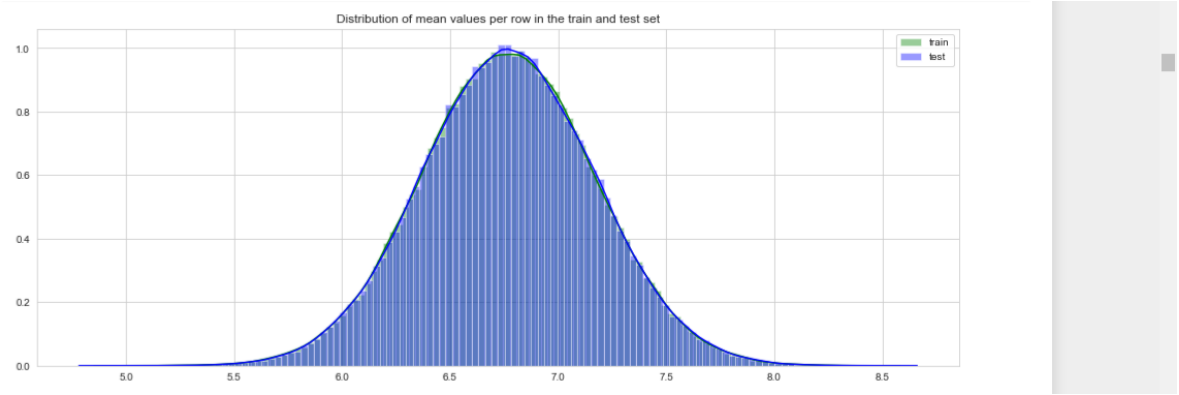


Fig:Distribution of mean values per row in the train and test set

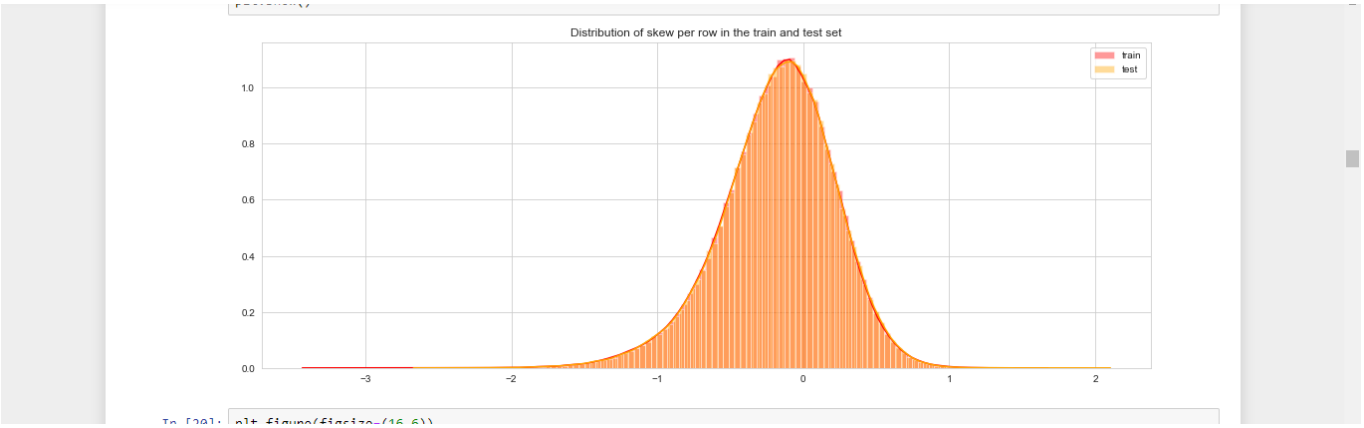


Fig: Distribution of skew per row in the train and test set

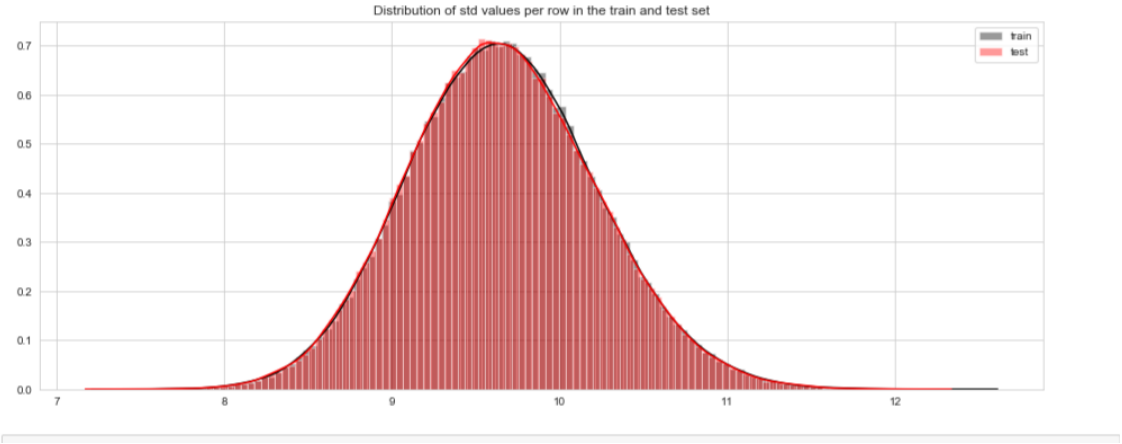


Fig : Distribution of std values per row in the train and test set

Fig: Distribution of mean values per column in the train and test set

CORRELATION:

Before performing any type of modelling we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. This process of selecting a subset of relevant features/variables is known as feature selection. There are several methods of doing feature selection.i have used correlation analysis

In our dataset, the correlation between the train attributes is very small.So, there is no need to remove variables

HANDLING IMBALANCED DATA:

Imbalanced classes are a common problem in machine learning classification where there are a disproportionate ratio of observations in each class. Class imbalance can be found in many different areas including medical diagnosis, spam filtering, and fraud detection.

Some popular methods for dealing with class imbalance.

1. Change the performance metric

Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be very misleading. Metrics that can provide better insight include:

- **Confusion Matrix:** a table showing correct predictions and types of incorrect predictions.
- **Precision:** the number of true positives divided by all positive predictions. Precision is also called Positive Predictive Value. It is a measure of a classifier's exactness. Low precision indicates a high number of false positives.
- **Recall:** the number of true positives divided by the number of positive values in the test data. Recall is also called Sensitivity or the True Positive Rate. It is a measure of a classifier's completeness. Low recall indicates a high number of false negatives.
- **F1: Score:** the weighted average of precision and recall.

2. Change the algorithm

While in every machine learning problem, it's a good rule of thumb to try a variety of algorithms, it can be especially beneficial with imbalanced datasets. Decision trees,Random Forests frequently perform well on imbalanced data. They work by learning a hierarchy of if/else questions and this can force both classes to be addressed.

3. Resampling Techniques—Oversample minority class

Our next method begins our resampling techniques.

Oversampling can be defined as adding more copies of the minority class. Oversampling can be a good choice when you don't have a ton of data to work with.

We will use the resampling module from Scikit-Learn to randomly replicate samples from the minority class.

Always split into test and train sets BEFORE trying oversampling techniques! Oversampling before splitting the data can allow the exact same observations to be present in both the test and train sets. This can allow our model to simply memorize specific data points and cause overfitting and poor generalization to the test data.

After resampling we have an equal ratio of data points for each class

4. Resampling techniques—Undersample majority class

Undersampling can be defined as removing some observations of the majority class. Undersampling can be a good choice when you have a ton of data -think millions of rows. But a drawback is that we are removing information that may be valuable. This could lead to underfitting and poor generalization to the test set.

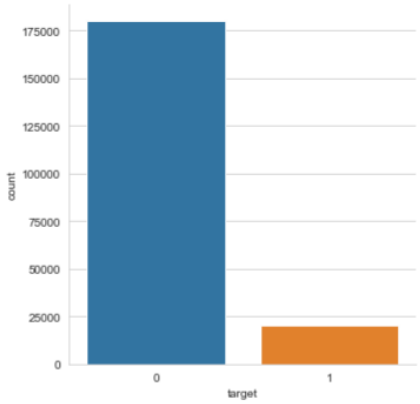
5. Generate synthetic samples

A technique similar to upsampling is to create synthetic samples.

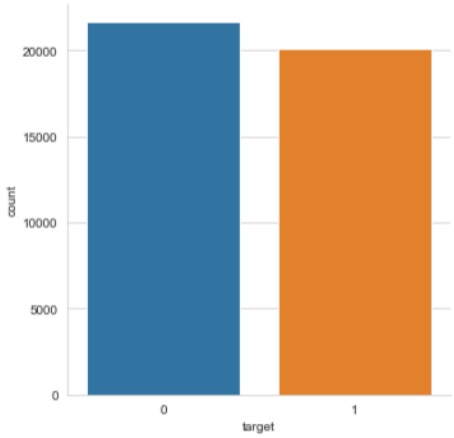
SMOTE (Synthetic Minority Oversampling Technique)uses a nearest neighbors algorithm to generate new and synthetic data we can use for training our model.

Again, it's important to generate the new samples only in the training set to ensure our model generalizes well to unseen data.

IN OUR CASE GIVEN DATA IS IMBALANCED……WHERE 90% OF SAMPLES BELONGS TO CLASS 0 AND ONLY 10% BELONGS TO CLASS 1



After applying undersampling technique,we get the balanced data as shown below



FEATURE SCALING:

Data scaling or feature scaling is a method used to standardize the range of variables or features present in the dataset so that they can be compared on a common ground.

Since the range of values for some variables in the raw data vary highly in magnitudes, units and range, we need feature scaling to bring all the features/variables to the same level of magnitudes, else the whole output of our analysis may get biased to one of the variables.

Most of the machine learning algorithms which use distance-based calculation might go wrong in their calculations if we do not scale our variables in the dataset before feeding into the model.

Another reason why feature scaling is applied is that gradient_descent converges much faster with feature scaling than without it.

Standardisation is a scaling method in which all the variables is brought into proportion with one another with values ranging from 0 to 1.

Xchanged=(X-Mean)/Standard Deviation

Standardisation rescales the data to have mean 0,and standard deviation of 1

PCA:

Principal Component Analysis, or PCA for short, is a method for reducing the dimensionality of data.

It can be thought of as a projection method where data with m-columns (features) is projected into a subspace with m or fewer columns, whilst retaining the essence of the original data.

The PCA method can be described and implemented using the tools of linear algebra.

PCA is an operation applied to a dataset, represented by an n x m matrix A that results in a projection of A which we will call B. Let's walk through the steps of this operation.

```
     a11, a12
A = (a21, a22)
     a31, a32

B = PCA(A)
```

The first step is to calculate the mean values of each column.

```
M = mean(A)
```

Next, we need to center the values in each column by subtracting the mean column value

```
C = A - M
```

The next step is to calculate the covariance matrix of the centered matrix C.

Correlation is a normalized measure of the amount and direction (positive or negative) that two columns change together. Covariance is a generalized and unnormalized version of correlation across multiple columns. A covariance matrix is a calculation of covariance of a given matrix with covariance scores for every column with every other column, including itself.

```
V = cov(C)
```

Finally, we calculate the eigendecomposition of the covariance matrix V. This results in a list of eigenvalues and a list of eigenvectors.

```
values, vectors = eig(V)
```

The eigenvectors represent the directions or components for the reduced subspace of B, whereas the eigenvalues represent the magnitudes for the directions

The eigenvectors can be sorted by the eigenvalues in descending order to provide a ranking of the components or axes of the new subspace for A.

If all eigenvalues have a similar value, then we know that the existing representation may already be reasonably compressed or dense and that the projection may offer little. If there are eigenvalues close to zero, they represent components or axes of B that may be discarded.

A total of m or less components must be selected to comprise the chosen subspace. Ideally, we would select k eigenvectors, called principal components, that have the k largest eigenvalues.

```
B = select(values, vectors)
```

Other matrix decomposition methods can be used such as Singular-Value Decomposition, or SVD. As such, generally the values are referred to as singular values and the vectors of the subspace are referred to as principal components.

Once chosen, data can be projected into the subspace via matrix multiplication.

```
P = B^T . A
```

Where A is the original data that we wish to project, B^T is the transpose of the chosen principal components and P is the projection of A.

This is called the covariance method for calculating the PCA, although there are alternative ways to to calculate it.

We can calculate a Principal Component Analysis on a dataset using the PCA() class in the scikit-learn library. The benefit of this approach is that once the projection is calculated, it can be applied to new data again and again quite easily.

When creating the class, the number of components can be specified as a parameter.

The class is first fit on a dataset by calling the fit() function, and then the original dataset or other data can be projected into a subspace with the chosen number of dimensions by calling the transform() function.

Once fit, the eigenvalues and principal components can be accessed on the PCA class via the *explained_variance_* and *components_* attributes.

OUR DATA IS HAVING 202 FEATURES,HAVING TOO MANY FEATURES LEAD TO OVERFITTING

SOO TO AVOID OVERFITTING AND FOR VISUALIZATION , I HAVE PERFORMED PCA FOR GIVEN DATASET

MODELLING:

This is the final phase of our project where we would build some machine learning models and will train our model on the data for future predictions. We would consider different machine learning algorithms to check which gives the best result.

Classification Accuracy

Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

It works well only if there are equal number of samples belonging to each class.

For example, consider that there are 98% samples of class A and 2% samples of class B in our training set. Then our model can easily get **98% training accuracy** by simply predicting every training sample belonging to class A.

When the same model is tested on a test set with 60% samples of class A and 40% samples of class B, then the **test accuracy would drop down to 60%.**Classification Accuracy is great, but gives us the false sense of achieving high accuracy.

The real problem arises, when the cost of misclassification of the minor class samples are very high. If we deal with a rare but fatal disease, the cost of failing to diagnose the disease of a sick person is much higher than the cost of sending a healthy person to more tests.

Confusion Matrix

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.

Lets assume we have a binary classification problem. We have some samples belonging to two classes : YES or NO. Also, we have our own classifier which predicts a class for a given input sample. On testing our model on 165 samples ,we get the following result.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Confusion Matrix

There are 4 important terms :

- **True Positives** : The cases in which we predicted YES and the actual output was also YES.
- **True Negatives** : The cases in which we predicted NO and the actual output was NO.
- **False Positives** : The cases in which we predicted YES and the actual output was NO.
- **False Negatives** : The cases in which we predicted NO and the actual output was YES.

Accuracy for the matrix can be calculated by taking average of the values lying across the **"main diagonal"** i.e

$$Accuracy = \frac{True Positives + False Negatives}{Total Number of Samples}$$

$$\therefore Accuracy = \frac{100 + 50}{165} = 0.91$$

Confusion Matrix forms the basis for the other types of metrics.

Area Under Curve

Area Under Curve(AUC) is one of the most widely used metrics for evaluation. It is used for binary classification problem. AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. Before defining AUC, let us understand two basic terms :

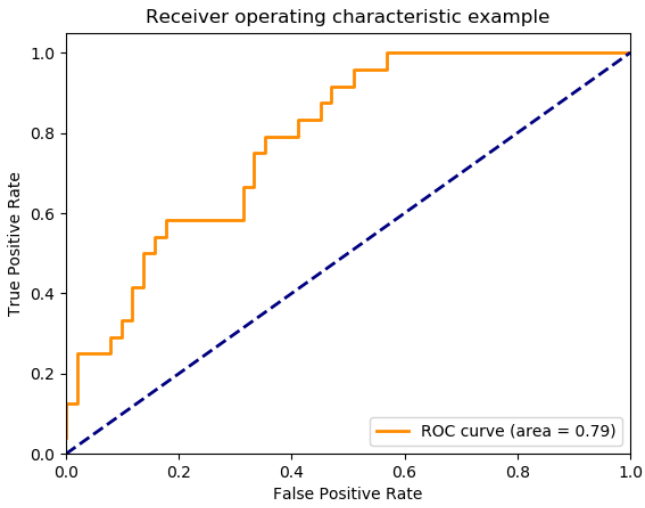
- **True Positive Rate (Sensitivity)** : True Positive Rate is defined as *TP/ (FN*+TP). True Positive Rate corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

$$TruePositiveRate = \frac{TruePositive}{FalseNegative + TruePositive}$$

- **False Positive Rate (Specificity)** : False Positive Rate is defined as *FP / (*FP+TN). False Positive Rate corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points.

$$FalsePositiveRate = \frac{FalsePositive}{FalsePositive + TrueNegative}$$

False Positive Rate and True Positive Rate both have values in the range [0, 1]. FPR and TPR both are computed at threshold values such as (0.00, 0.02, 0.04, …., 1.00) and a graph is drawn. AUC is the area under the curve of plot False Positive Rate vs True Positive Rate at different points in [0, 1].



As evident, AUC has a range of [0, 1]. The greater the value, the better is the performance of our model.

F1 Score

F1 Score is used to measure a test's accuracy

F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

F1 Score tries to find the balance between precision and recall.

- Precision : It is the number of correct positive results divided by the number of positive results predicted by the classifier.

  $$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

- Recall : It is the number of correct positive results divided by the number of *all* relevant samples (all samples that should have been identified as positive).

  Recall=true positives/true positives+false negatives

LOGISTIC REGRESSION:

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

Results for Logistic Regression:

```
precision: [0.76427924 0.75947114]
recall: [0.78091709 0.74174323]
fscore: [0.77250859 0.75050251]
```

RANDOM FOREST

Random forests are based on a simple idea: 'the wisdom of the crowd'. Aggregate of the results of multiple predictors gives a better prediction than the best individual predictor. A group of predictors is called an **ensemble**. Thus, this technique is called **Ensemble Learning**.

To improve our technique, we can train a group of **Decision Tree classifiers**, each on a different random subset of the train set. To make a prediction, we just obtain the predictions of all individuals trees, then predict the class that gets the most votes. This technique is called **Random Forest.**

Random forest chooses a random subset of features and builds many Decision Trees. The model averages out all the predictions of the Decisions trees.

Results for Random Forest:

```
precision: [0.75680421 0.77038797]
recall: [0.79851783 0.72485721]
fscore: [0.77710165 0.74692938]
```

There is not much difference between random forest and logistic regression only .1% is different

As logistic(75%)is more than random forest (74%),logistic regression is used for unseen data