Customer Transaction Prediction

Contents

Introduction

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.

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?

According to the past data and from the given problem, we find out it is a Classification problem under Supervised Machine Learning. We train the model with past data and use it to predict results for the new data or test data.

Data

Given data contains numeric feature variables, the target column which holds binary values, and a ID_code column which has string values. The task is to predict the value of target column in the test set by analyzing the training data.

- ID_code (string)
- Target
- Data has 200 numerical variables, named from var_0 to var_199;
- It has 201 predictors or independent variables and 1 target variable 'target'

```
train.head()
```

	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_193	var_194	var_195	var_196	v
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	1.6910	18.5227	-2.3978	7.8784	
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	10.9516	15.4305	2.0339	8.1267	
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	1.6858	21.6042	3.1417	-6.5213	
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	1.4214	23.0347	-1.2706	-2.9275	
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	9.1942	13.2876	-1.5121	3.9267	

5 rows × 202 columns

Methodology

Pre-Processing Techniques

Data pre-processing is a data mining technique that involves transforming raw data into understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. So, data pre-processing is a proven method of resolving such issues.

Missing Value Analysis:

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

Observation:

In both our given data sets train and test, we do not have any missing data values. So, we do not have to proceed with any further techniques to impute or replace the data.

Feature Selection:

It is the method of extracting relevant and meaningful features out of the data and to identify and remove the irrelevant attributes that do not contribute much.

- ***CORRELATION:*** 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 and all variables are important.

Modelling

- Splitting the Data: Splitting of data is done to make it easier for the algorithm model to make predictions. Generally, we split the dataset into 70:30 ratio or 80:20 ratio i.e, 70 percent data is taken into the training set and 30 percent in the test set. However, this splitting varies according to the dataset shape and size. Our data has following parts:
    - X_train – is the training part of the matrix of features
    - X_test – is the test part of the matrix of features
    - Y_train – is the training part of the dependent variable(target) associated to X_train here.
    - Y_test – is the test part of the dependent variable associated to X_train here.

  O/P: `((160000, 200), (40000, 200), (160000,), (40000,))`

- Feature Scaling: It is the method to limit the range of variables so that they can be compared on common grounds.
- Handling Imbalanced Data: Imbalanced data means when the observations in the classes are not in proportionate ratio. Class imbalance can be found in many different

areas including medical diagnosis, spam filtering, and fraud detection. From our plot, we find out that the given data is imbalanced and so different error metrics are used instead of accuracy.
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 that is often used to describe the performance of a classification model. It is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. Terms used are:
    - True Positive (TP): Observation is positive and is predicted to be positive.
    - False Negative (FN): Observation is positive and is predicted to be negative.
    - True Negative (TN): Observation is negative and is predicted to be negative.
    - False Positive (FP): Observation is negative and is predicted to be positive.

- **Precision:** Precision is calculated as total number of correctly classified positives divided by total number of positive predictions. Precision is also called Positive Predictive Value. It is a measure of a classifier's exactness. High precision indicates that an example labeled as positive is indeed positive.

- **Recall:** It is the ratio of the total number of correctly classified positive examples to the total number of positive values in. Recall is also called Sensitivity or the True Positive Rate. It is a measure of a classifier's completeness. High recall indicates the class is correctly recognized (small FN value).

- **F1 Score:** It is the weighted average of precision and recall. It uses Harmonic mean in place of Arithmetic mean. The F-measure will always be nearer to the smaller value of Precision or Recall.

- ***Logistic Regression:***
  Logistic regression is the appropriate regression analysis to conduct when the dependent variable is binary. It is also a predictive analysis. It is used to describe data and to explain the relationship between one dependent binary variable and one or more independent variables.

  O/P:-

```
Confusion Matrix:
[[35498   500]
 [ 2954  1048]]
Accuracy: 0.913650
precision: [0.92317695 0.67700258]
recall: [0.98611034 0.26186907]
fscore: [0.95360645 0.37765766]
```

  This predicts that the model is 91% accurate. Precision for '0-No' is 92% and for '1-Yes' is 67%.

- ***Decision Tree***

  Decision tree builds classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.

  O/P:-

```
Accuracy score 0.899800
Confusion Matrix:
[[35969    29]
 [ 3979    23]]
precision: [0.90039551 0.44230769]
recall: [0.9991944  0.00574713]
fscore: [0.94722566 0.01134682]
```

  This predicts that the values have been classified 89% accurately with a precision value of 90% for ‘0’ and 44% for ‘1’.

- ***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 individual trees, then predict the class that gets the most votes. This technique is called Random Forest. The model averages out all the predictions of the Decisions trees.

```
Accuracy score 0.900475
[[35992     6]
 [ 3975    27]]
precision: [0.90054295 0.81818182]
recall: [0.99983332 0.00674663]
fscore: [0.94759429 0.0133829 ]
```

  This predicts that the values have been classified with an accuracy of 90%.

Conclusion

Model Evaluation

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=(TP+TN)/Total no of prediction

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.

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). F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall and there is an uneven class distribution (large number of Actual Negatives).The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :F1 Score tries to find the balance between precision and recall.

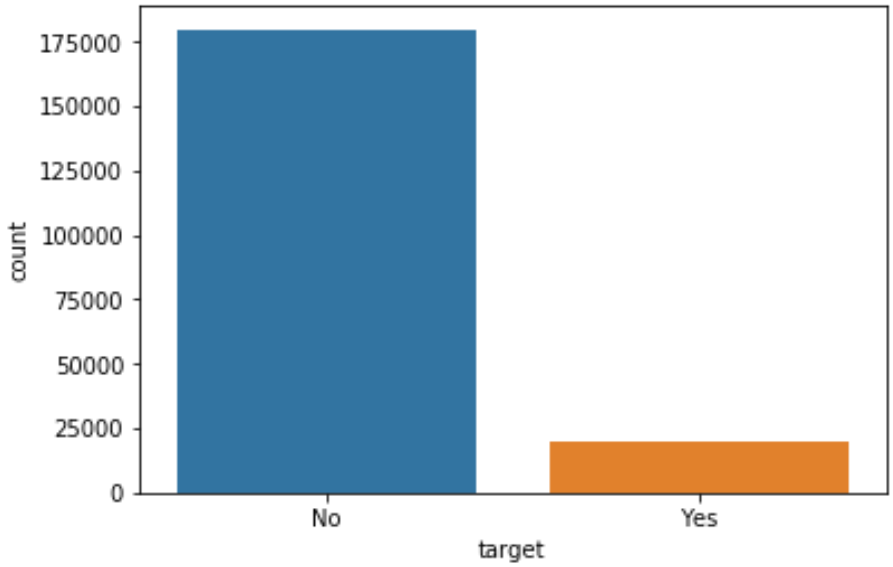
- Precision : It is the number of correct positive results divided by the number of positive results predicted by the classifier.Immediately, you can see that Precision talks about how precise/accurate your model is out of those predicted positive, how many of them are actual positive**.**

- 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).

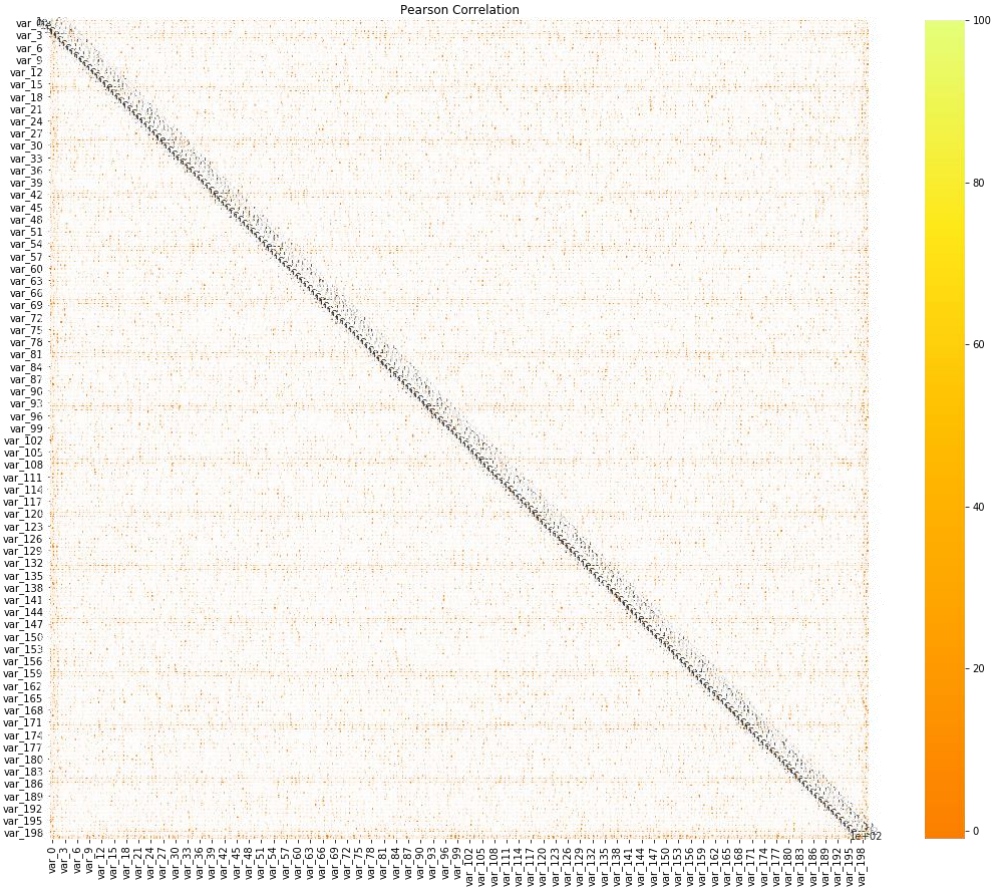
Recall= (true positives/true positives)+false negatives

For instance, in fraud detection or sick patient detection. If a fraudulent transaction (Actual Positive) is predicted as non-fraudulent (Predicted Negative), the consequence can be very bad for the bank.

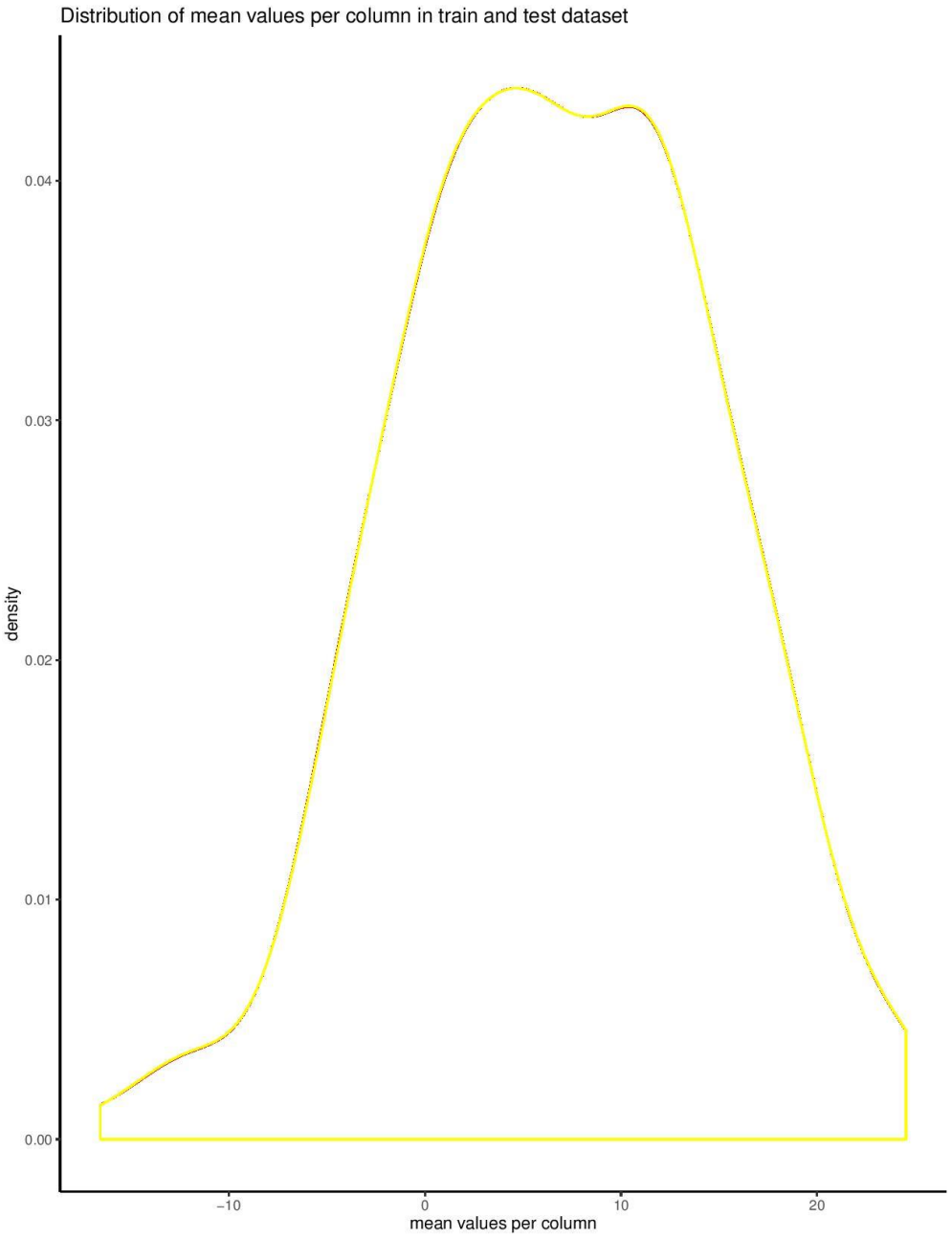
Visualization

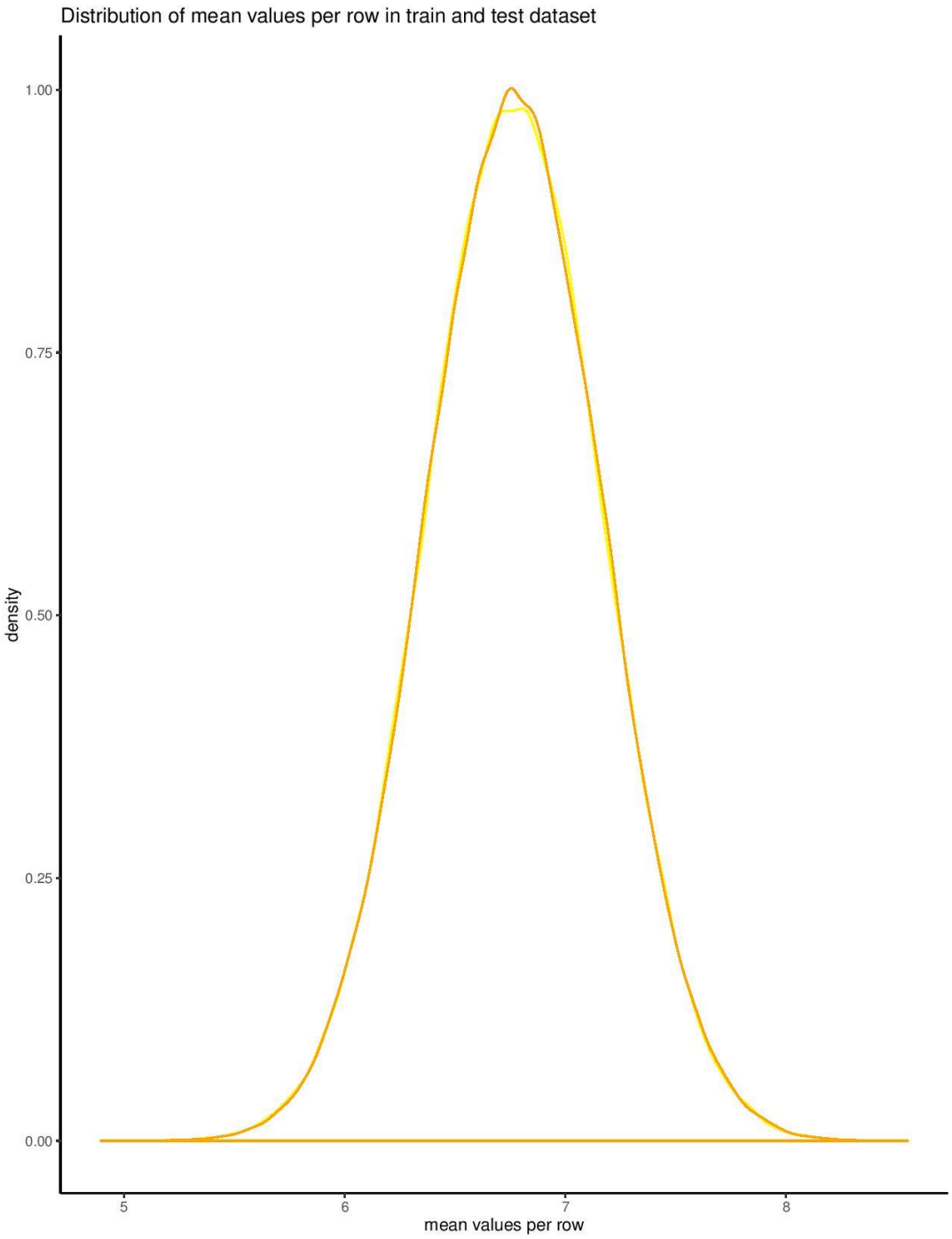


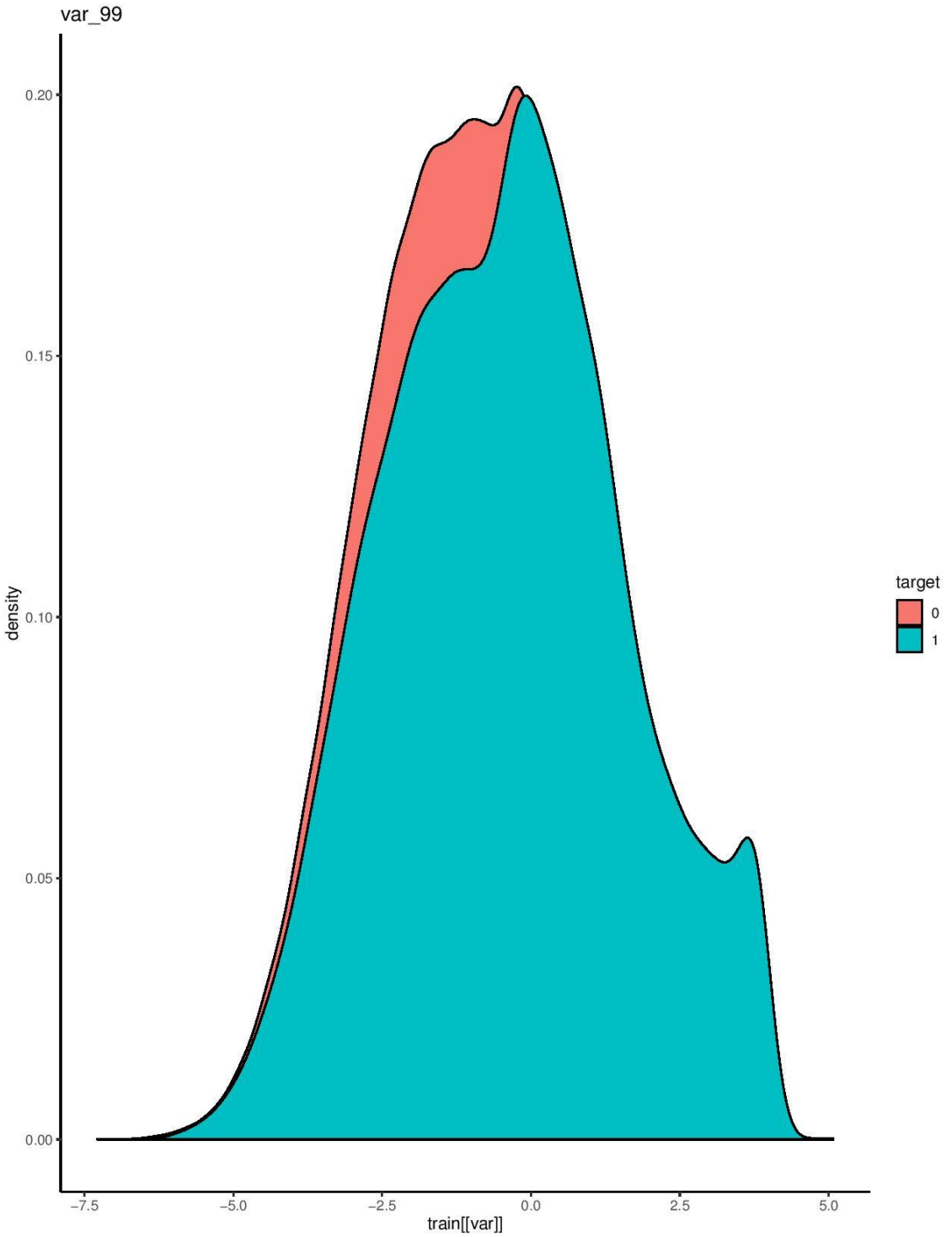
Predicts that our dataset is imbalanced. 90% of the customers will not make a transaction and 10% will make a transaction.

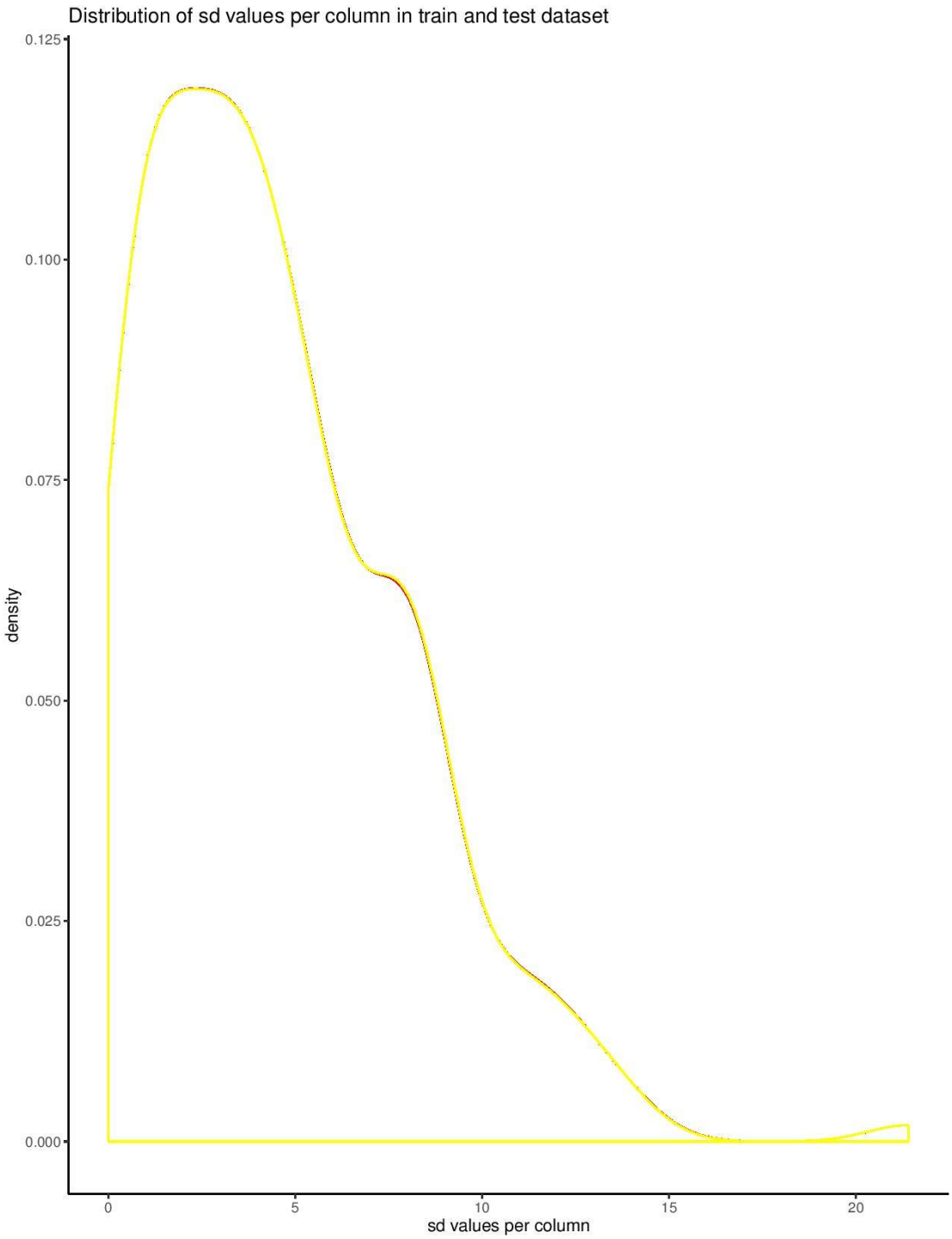


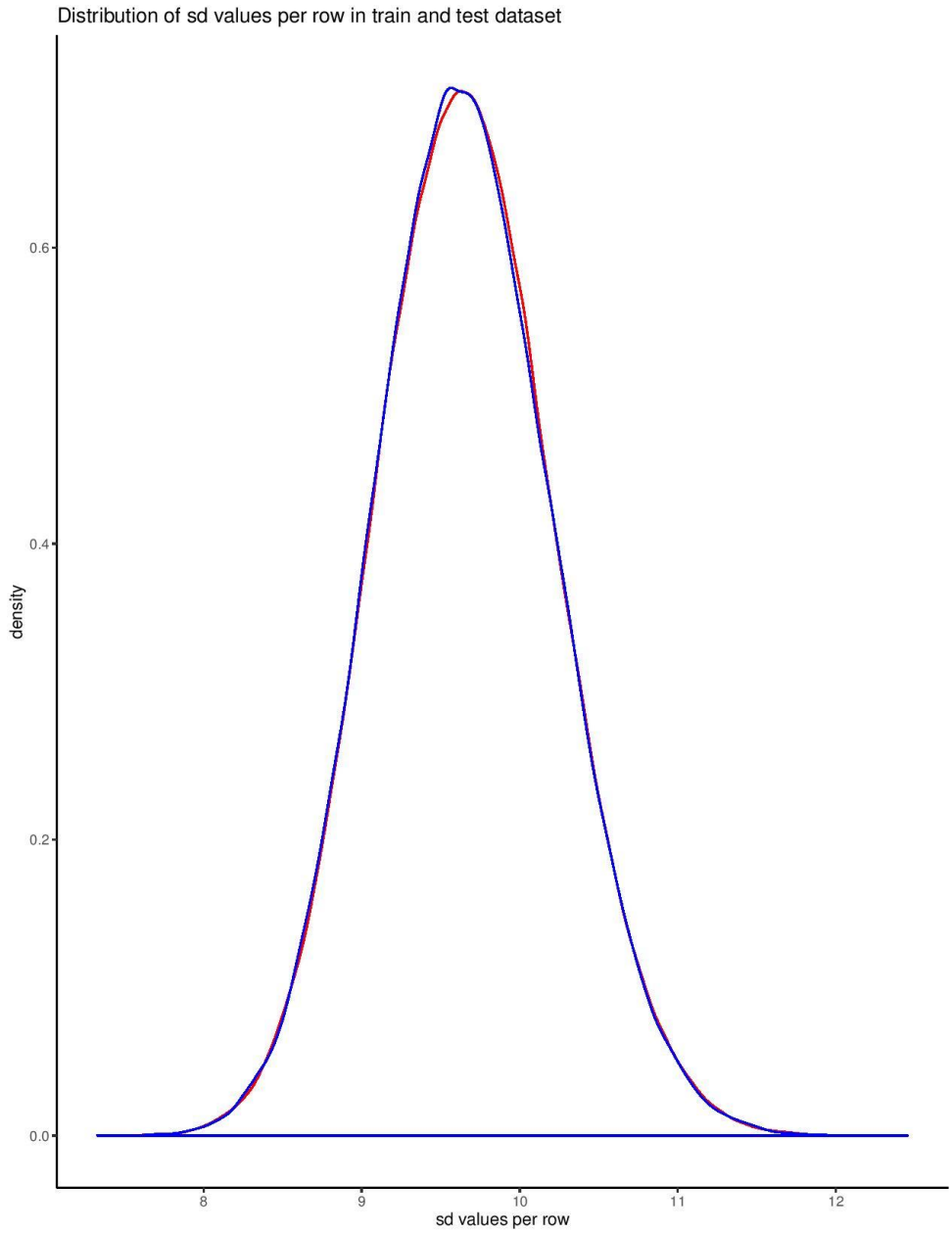
Predicts that the correlation is low and so the columns don't need to be dropped.

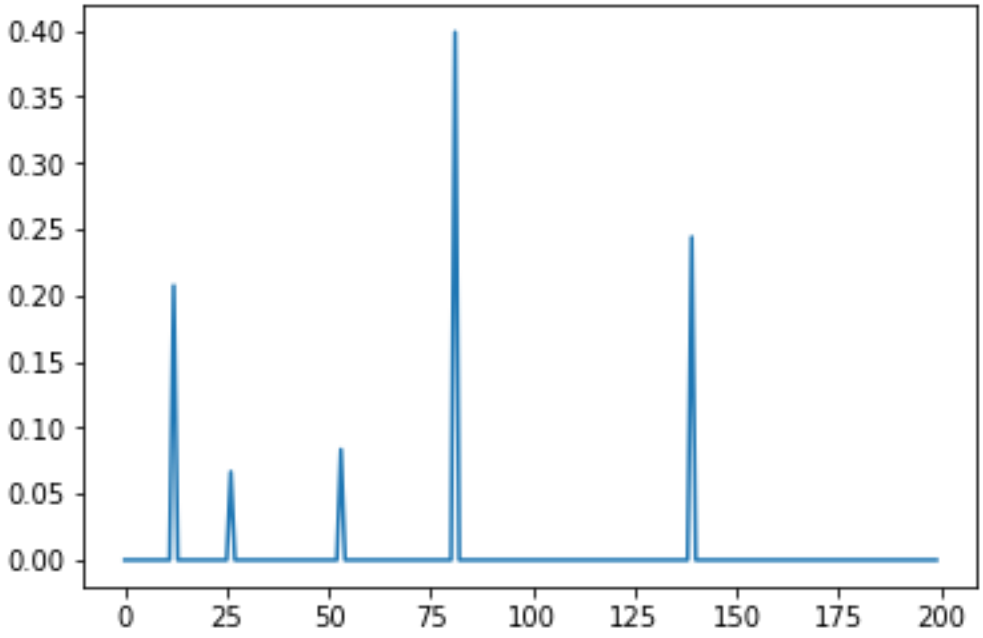

Distribution of mean values per column in train and test dataset
density
0.04
0.03
0.02
0.01
0.00
−10
0
10
20
mean values per column


Distribution of mean values per row in train and test dataset
1.00
0.75
0.50
0.25
0.00
density
5
6
7
8
mean values per row

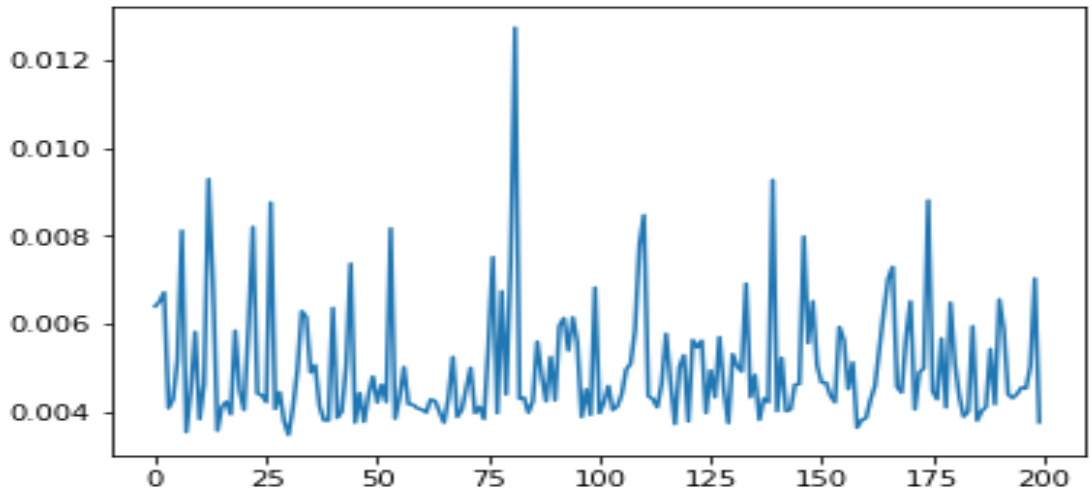

var_99
0.20
0.15
0.10
0.05
0.00
density
−7.5
−5.0
−2.5
0.0
2.5
5.0
train[[var]]
target
0
1


Distribution of sd values per column in train and test dataset
0.125
0.100
0.075
0.050
0.025
0.000
density
0
5
10
15
20
sd values per column


Distribution of sd values per row in train and test dataset
density
0.0
0.2
0.4
0.6
8
9
10
11
12
sd values per row



Decision tree feature importance



Random Forest feature importance