Santander Customer Transaction Prediction using Azure ML

Data Mining: ISM6136.001S19

By

Divya Macha

Kirti Tiwari

Srikrishna Srinivasan

Ritik Gupta

Shreyas Hastantram

May 1, 2019

Table of Contents

Introduction

Santander has been serving customers in the Northeast since 2013. Their mission is to help customers prosper. They do it with simple ways to spend, save, and manage customers money better.

Project Goal: The goal of this analysis is to use Microsoft Azure Machine Learning Studio to explore, modify, model and assess this data and need to predict whether the customers will perform transaction for the next time in future or not.

Questions that frame the exercise?

Santander uses a lot of binary classification to find out interesting insights like will the customer be able to pay the loan, is customer satisfied, etc.The goal here is to harness technology to predict whether a customer will make a specific transaction in the future or not for the leader or

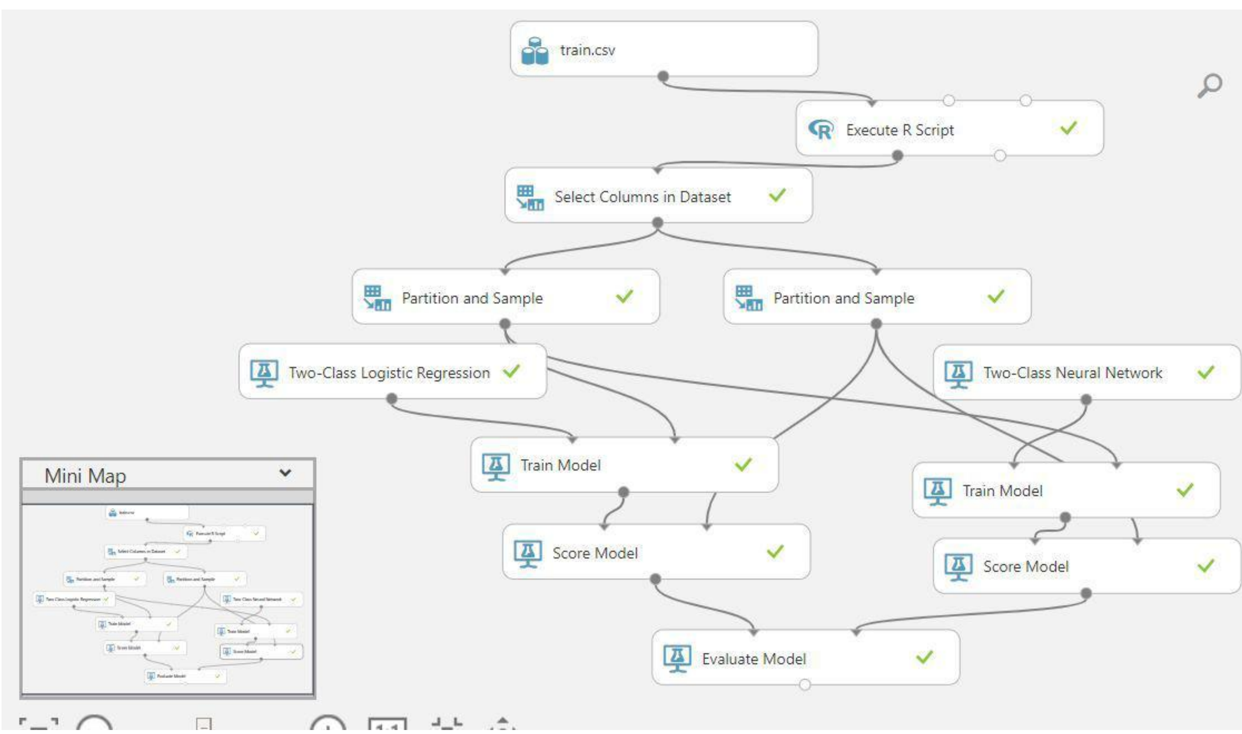
CEO to devise plan for strategic marketing. In the data which we have, we can observe that most of the customers won't make a purchase i.e target is 0 all the time. So, Important concerns are why is most of the customers are not making purchase, Or on what conditions are they making a purchase. How can the company assess which customers will make a purchase. Furthermore, Is there some way to find a solution to make the customer, make transaction.

Recommendations for the problem?

If we look the problem as the data science problem, we need to predict target variable i.e will the customer make transaction or not the next time. And for that we need to fit the model to give us correct accuracy. Once we get the accurate target variables, CEO can look into factors affecting that and help guide marketing team.

Models which we used to solve this real world problem is Logistic Regression and Neural Networks

Here is our model,



We have used Azure ML to build this model, train.csv is the dataset which is downloaded from the kaggle website.

Dataset Description

Experiment created on 24/03/2019 › train.csv › dataset

rows	columns
200000	202

view as

ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7
train_0	0	8.9255	-6.7863	11.9081	5.093	11.4607	-9.2834	5.1187	18.6266
train_1	0	11.5006	-4.1473	13.8588	5.389	12.3622	7.0433	5.6208	16.5338
train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	14.6155
train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	14.925
train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	19.2514
train_5	0	11.4763	-2.3182	12.608	8.6264	10.9621	3.5609	4.5322	15.2255
train_6	0	11.8091	-0.0832	9.3494	4.2916	11.1355	-8.0198	6.1961	12.0771
train_7	0	13.558	-7.9881	13.8776	7.5985	8.6543	0.831	5.689	22.3262
train_8	0	16.1071	2.4426	13.9307	5.6327	8.8014	6.163	4.4514	10.1854
train_9	0	12.5088	1.9743	8.896	5.4508	13.6043	-16.2859	6.0637	16.841
train_10	0	5.0702	-0.5447	9.59	4.2987	12.391	-18.8687	6.0382	14.3797

It is evident from the screenshot that there is 202 columns and 200000 rows,the variables names has been hidden from the dataset, they have generic names.

The company is not letting there trade secrets out, variables might also been scaled so that the person working on the data won't be able to predict what this variable corresponds to in real world.

All the variables are distributed normally which indicates the fact that this is a real data and reliable data.

There are lot of zeros in the dataset when compared to 1 , this indicates there might be lot of failed transactions than the successful one's.

R Script

```
# Map 1-based optional input ports to variables
dataset <- maml.mapInputPort(1) # class: data.frame

# Contents of optional Zip port are in ./src/
# source("src/yourfile.R");
# load("src/yourData.rdata");

dataset1=dataset[1:10000,]


# Select data.frame to be sent to the output Dataset port
maml.mapOutputPort("dataset1");
```

We wanted our model to perform quickly hence we wrote a simple r script to select 10000 rows from the dataset.

We can from the command dataset[1:10000,], it is saying select 10000 rows and select all the columns.

The same dataset will be sent as the output from the module.

We are using a rscript module in azure ml to perform this function.

Since there is 202 variables and we wanted few among them, we ran a feature selection model on the data and we found 49 columns are more correlated with the target variable.

Experiment created on 24/03/2019 › Select Columns in Dataset › Results dataset

rows	columns
10000	49

target	var_0	var_1	var_2	var_6	var_12	var_13	var_18	var_21	var_22
0	8.9255	-6.7863	11.9081	5.1187	14.0137	0.5745	4.284	16.2191	2.5791
0	11.5006	-4.1473	13.8588	5.6208	14.0239	8.4135	7.8	2.7407	8.5524
0	8.6093	-2.7457	12.0805	6.9427	14.1929	7.3124	4.7011	18.1377	1.2145
0	11.0604	-2.1518	8.9522	5.8428	13.8463	11.9704	15.9426	12.5579	6.8202
0	9.8369	-1.4834	12.8746	5.9405	13.8481	7.8895	6.5263	18.9608	10.1102
0	11.4763	-2.3182	12.608	4.5322	13.638	1.2589	6.7341	11.9882	1.0468
0	11.8091	-0.0832	9.3494	6.1961	14.1629	13.3058	21.1976	24.2595	8.1159
0	13.558	-7.9881	13.8776	5.689	14.2919	10.9699	5.1548	14.4195	1.2375
0	16.1071	2.4426	13.9307	4.4514	14.0654	-3.0572	3.8349	6.3738	6.558
0	12.5088	1.9743	8.896	6.0637	13.9639	0.8071	5.4649	4.4221	6.1695
0	5.0702	-0.5447	9.59	6.0382	13.9059	9.0796	17.4442	8.2466	-0.5277
0	12.7188	-7.975	10.3757	5.6464	13.6713	9.5331	17.5407	2.3546	7.6435

Logistic Regression

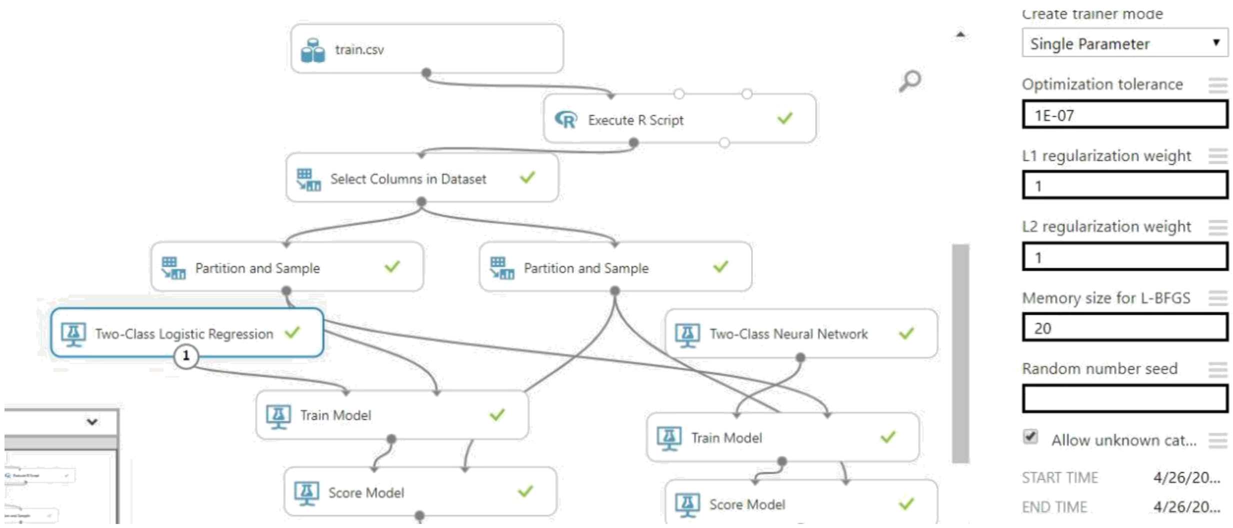
For this problem we used our first model as logistic regression. We chose logistic regression because we want to know the strength and statistical significance of each variable contributing to binary state of the target variable and in turn predicts its probability. We used L1 and L2 weights as 1 to prevent overfitting.

Using the two class logistic regression tab in the Azure ML we passed our features and sampled data. Based on this training data, logistic regression model will be calculated with each variable with some beta coefficient. This all will flush into sigmoid function to create a probabilistic value to determine the target. The sigmoid function is specified as

$$P(target=1)=1/1+e^{-z}$$

$$z= B0+xB1+xB2+xB3\ldots\ldots\ldots..$$

In our model all the x are the variables with their B terms indicating bent to probability



After, Applying logistic regression we train our model and after training we score our model to observe the table with probabilistic values to determine target.

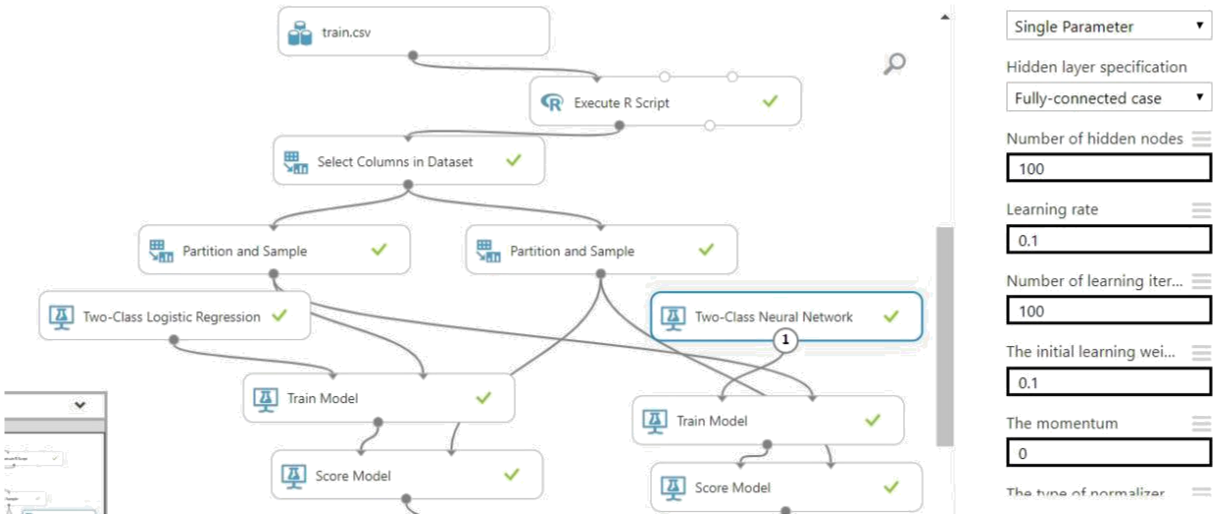
Neural Networks

We have also applied Neural Network model so that we can compare the results with Logistic Regression and see which model has performed better .

We used Two-Class Neural Network here. It is one of the supervised learning methods for classification. It is a set of interconnected layers starting with Input Layer which has number of nodes equal to the number of features of the training data and based on that we will predict the outcome.The input layer is connected to the nodes in the hidden layer which is comprised of weighted edges and nodes and finally connected to the output layer. We can also customize some features for the hidden layer so that it can give better performance like number of hidden layers (default is 1),number of hidden nodes(default 100), learning rate,number of learning iterations, momentum etc.

As we used two-class neural network, all inputs must map to one of two nodes in the output layer.

Below is the screenshot of Neural Network model from Azure ML displaying its feature selection as well.

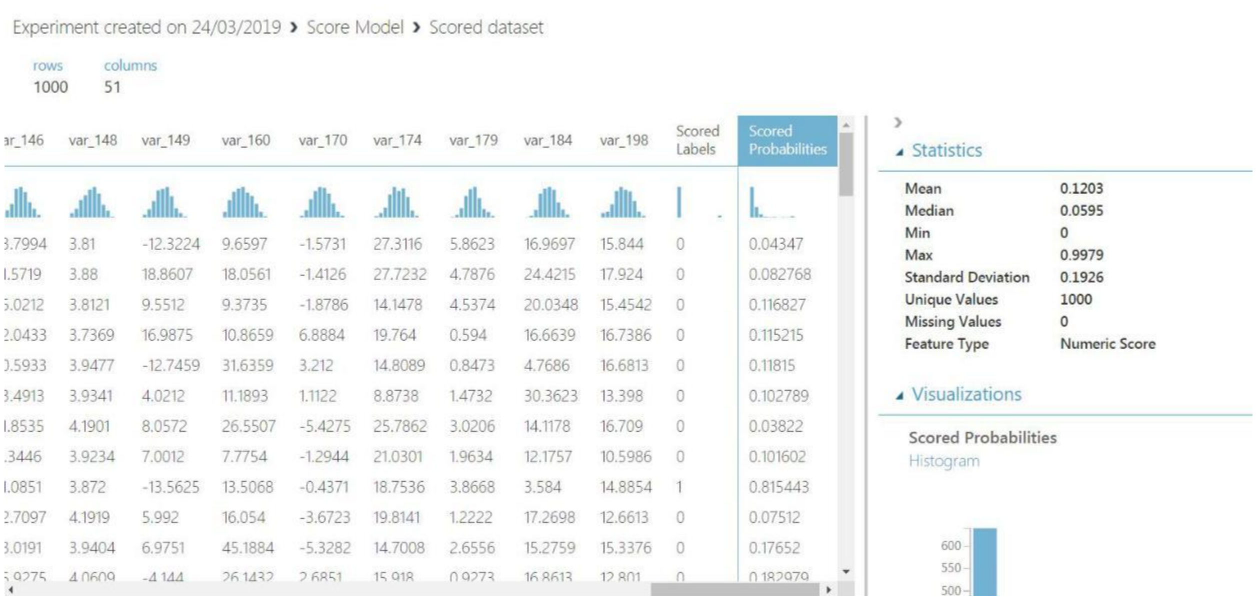


Next step is to train the model and validate its results and we can see the scored labels and scored probabilities which the model has predicted when we visualize the scored model results. We can check the Statistics like mean, median, standard deviation etc and based on that can determine if the model is good or not.

Scored Model Result Evaluation

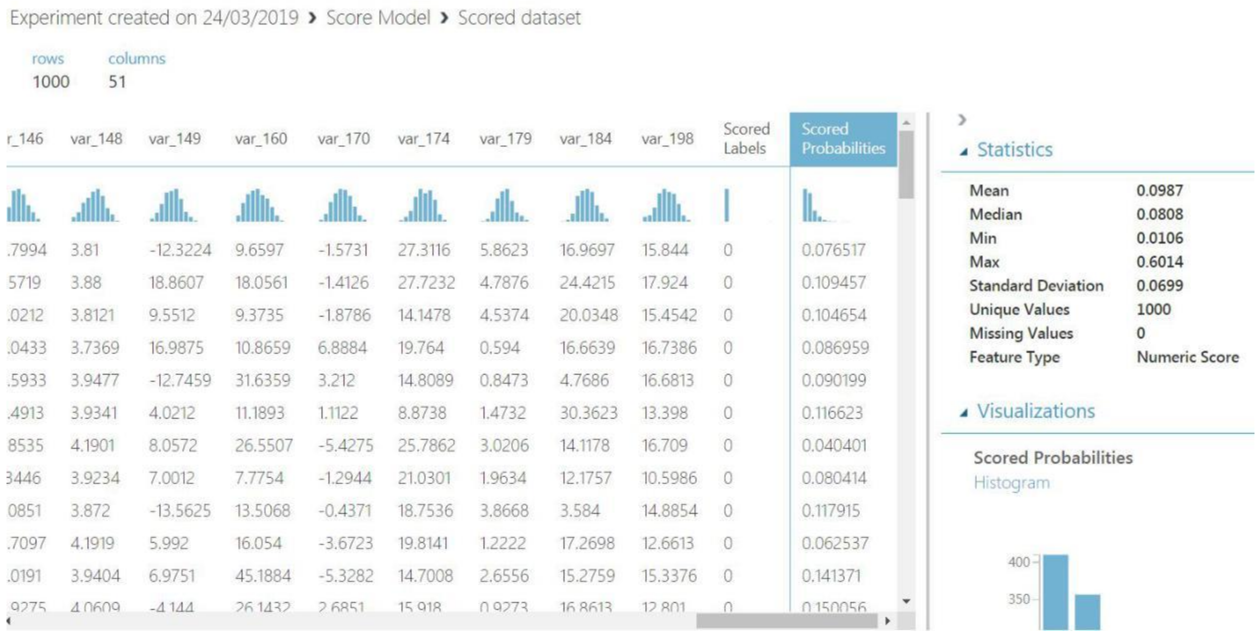
Below are the evaluation Results for the two model which we have used.

Score Model Results for Two-class Neural Network



The mean and standard deviation are not very close, therefore this is not as good compared to logistic regression (shown below)

Score Model Results for Two-class Logistic Regression :



Experiment created on 24/03/2019 › Score Model › Scored dataset

rows	columns
1000	51

r_146	var_148	var_149	var_160	var_170	var_174	var_179	var_184	var_198	Scored Labels	Scored Probabilities
.7994	3.81	-12.3224	9.6597	-1.5731	27.3116	5.8623	16.9697	15.844	0	0.076517
5719	3.88	18.8607	18.0561	-1.4126	27.7232	4.7876	24.4215	17.924	0	0.109457
.0212	3.8121	9.5512	9.3735	-1.8786	14.1478	4.5374	20.0348	15.4542	0	0.104654
.0433	3.7369	16.9875	10.8659	6.8884	19.764	0.594	16.6639	16.7386	0	0.086959
.5933	3.9477	-12.7459	31.6359	3.212	14.8089	0.8473	4.7686	16.6813	0	0.090199
.4913	3.9341	4.0212	11.1893	1.1122	8.8738	1.4732	30.3623	13.398	0	0.116623
8535	4.1901	8.0572	26.5507	-5.4275	25.7862	3.0206	14.1178	16.709	0	0.040401
3446	3.9234	7.0012	7.7754	-1.2944	21.0301	1.9634	12.1757	10.5986	0	0.080414
0851	3.872	-13.5625	13.5068	-0.4371	18.7536	3.8668	3.584	14.8854	0	0.117915
.7097	4.1919	5.992	16.054	-3.6723	19.8141	1.2222	17.2698	12.6613	0	0.062537
.0191	3.9404	6.9751	45.1884	-5.3282	14.7008	2.6556	15.2759	15.3376	0	0.141371
.9275	4.0609	-4.144	26.1432	2.6851	15.918	0.9273	16.8613	12.801	0	0.150056

Statistics

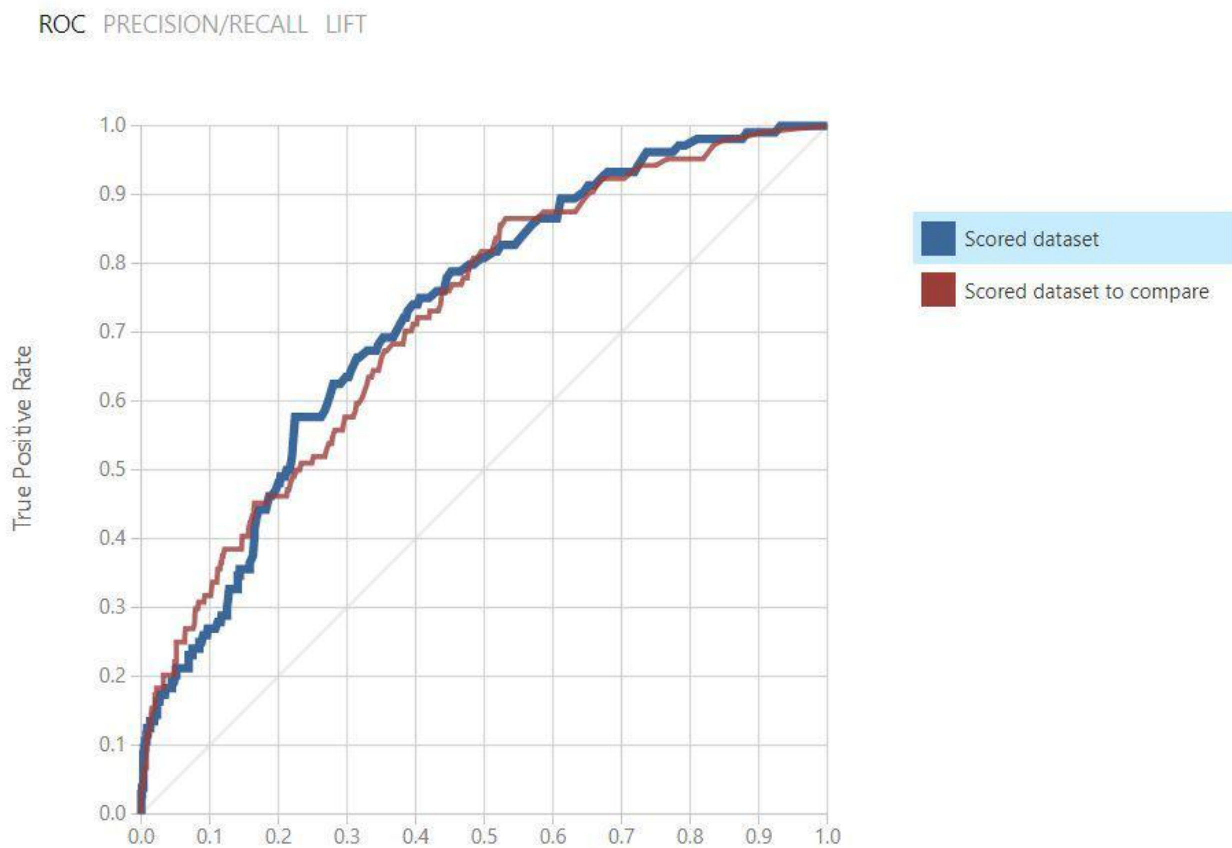
Mean	0.0987
Median	0.0808
Min	0.0106
Max	0.6014
Standard Deviation	0.0699
Unique Values	1000
Missing Values	0
Feature Type	Numeric Score

Visualizations

Scored Probabilities
Histogram

400
350

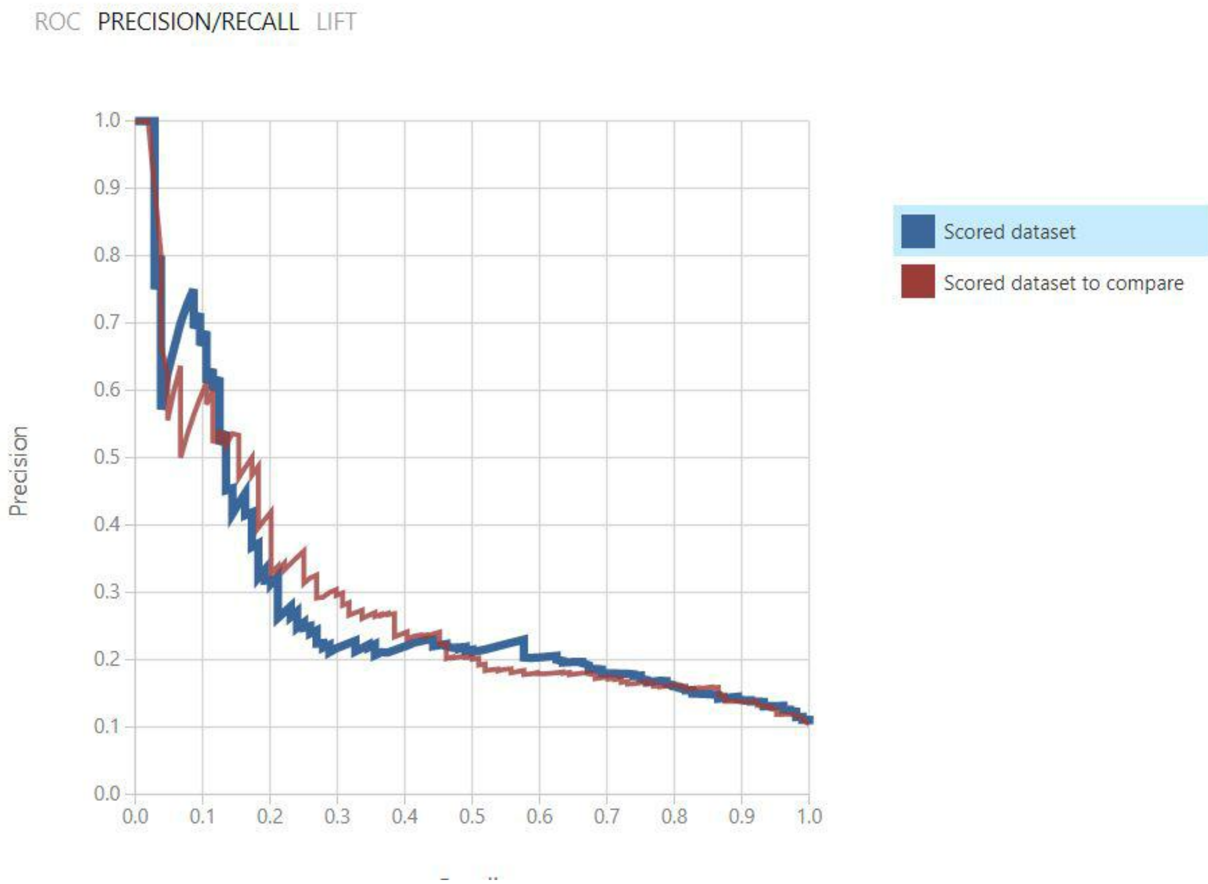
Interpretation and Comparison of Evaluation Model Results



ROC Plot and AUC measure:

1. Blue curve is for the model on the left and it is the Two class Logistic regression
2. Red curve is for the model on the right and it is the Two class Neural network
3. Lines connect lower left corner to upper right corner in the graph
4. Each point represents how model performs along two dimensions – false positive and true positive
5. There is a threshold. By changing it we decrease the frequency of one type of error at the expense of increasing other type of error
6. The line which comes closest to upper-left corner provides best predictions
7. If the curve is towards the middle, it is not good in performance
8. The worst possible model is a diagonal line, it produces random predictions with the same distribution as class distribution
9. It strikes right balance between the two types of errors

Logistic regression is a better solution as seen above.



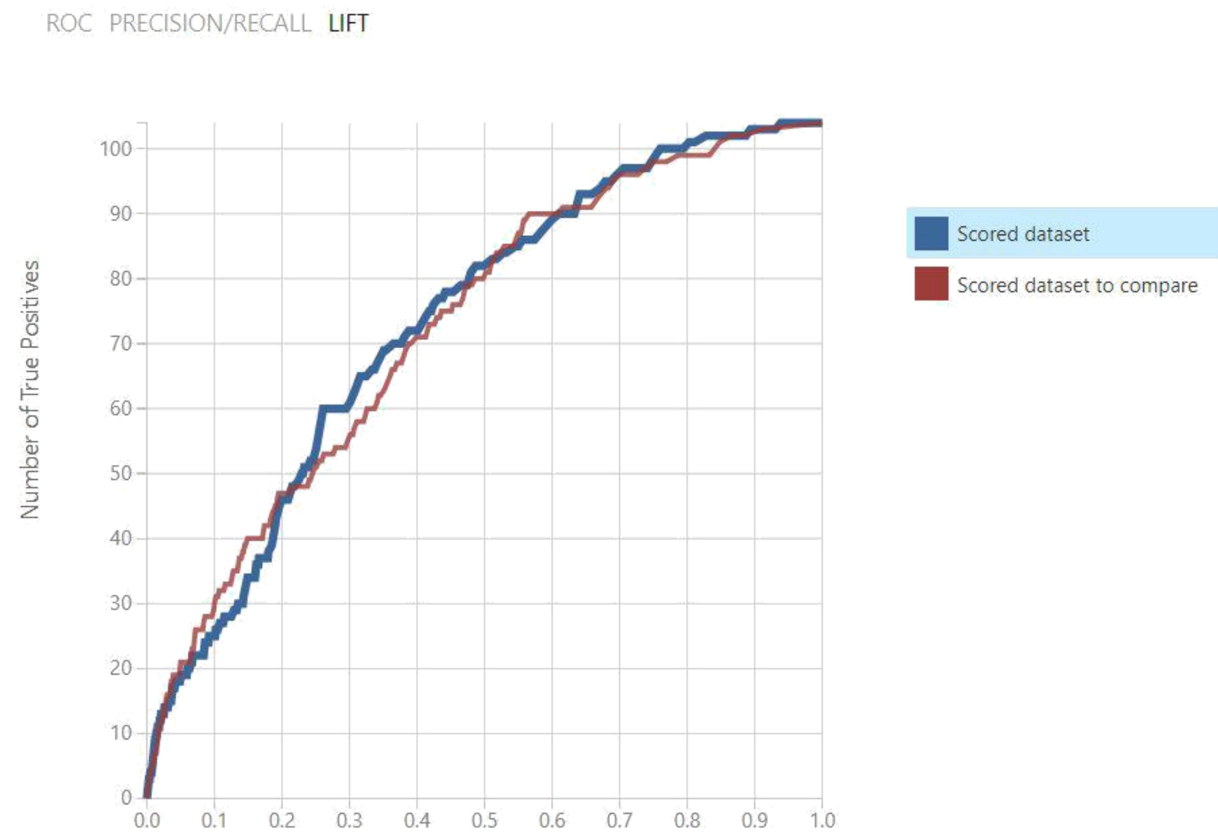
Precision / Recall plot

The curves are pulling to the left (usually it is expected to pull to right, but left is considered acceptable). Blue curve pulls better, hence it is a better model.

1. Recall = TP / (TP + FN)

2. Precision = TP / (TP + FP)

3. The curve to the upper right corner is the best model



Area Under the Curve / AUC:

1. Shows the amount of the area under the ROC curve, value between 0 and 1
2. Value should never be less than 0.5 (diagonal)

The models are good, as they are pulling to the right, asway from diagonal.

False Positives: Actual value is NO and predicted value is YES

True Positives: Actual value is YES and predicted value is YES

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
3	101	0.899	1.000	0.5	0.728
False Positive	**True Negative**	**Recall**	**F1 Score**		
0	896	0.029	0.056		
Positive Label	**Negative Label**				
1	0				

Accuracy = Correctly predicted to Total = TP+TN / (TP + TN + FP + FN)
Precision = Correctly predicted positive to total predicted positive = TP / (TP + FP)
Recall (sensitivity) = Correctly predicted positive to all observations = TP / (TP+FN)
F1 score= weighted average of precision and recall = 2*(Recall*Precision)/(R+P)
The model has a very good Accuracy, as True Positives and True Negatives are much higher compared to False Positive and False Negative.

Precision is 1 because False Positives are 0

Recall is very low, because, True Positive is a small fraction compared to False Negatives.