Machine Learning Engineer Nanodegree

Capstone Project: Home Credit Default Analysis using Machine Learning and Deep Learning

Monimoy Deb Purkayastha
September 9th 2018

I. Definition

Project Overview

For financial institutes like banks giving loan to customers is a complicated process. Banks want to ensure that it gives loans to those customers who have low risk. If the customer defaults in repaying loans it will be a loss to Bank. That is why Banks perform extensive credit risk analysis before approving the loan to customer.

In this capstone project I have chosen Kaggle competition challenge "Home Credit Default Analysis" where I also participated in the competition. The URL for the competition is: https://www.kaggle.com/c/home-credit-default-risk

Home Credit is a global financial institute which provides loans to lender. Home Credit operates in 10 countries globally

Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Home Credit makes use of a variety of alternative data--including telco and transactional information--to predict their clients' repayment abilities.

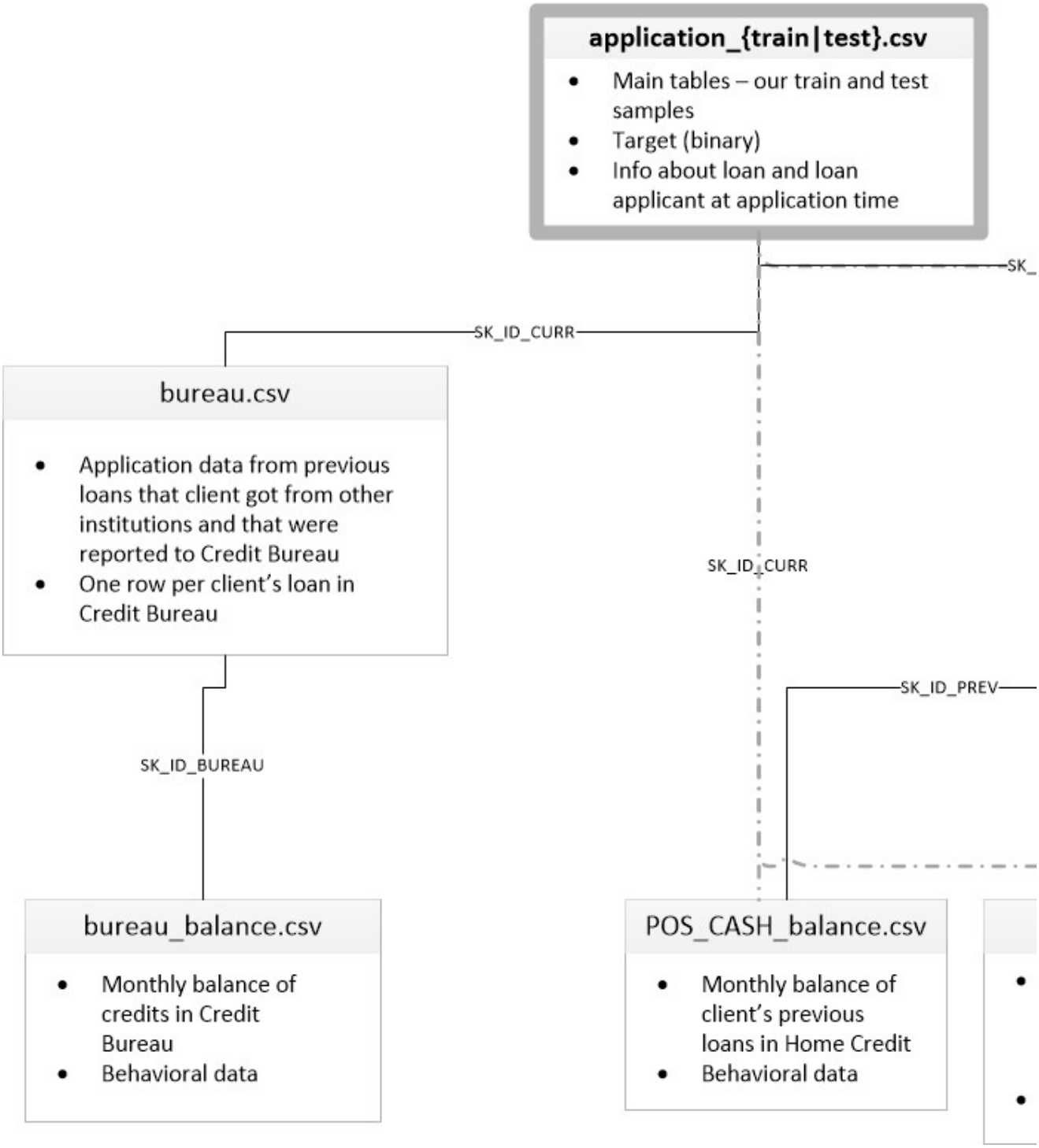
While Home Credit is currently using various statistical and machine learning methods to make these predictions, they're challenging Kagglers to help them unlock the full potential of their data. Doing so will ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful.

I have analysed the data provided by Home Credit data using statistical techniques. Next I have applied Machine Learning and Deep Learning techniques to predict risk of each customer

Datasets:

Dataset is provided in https://www.kaggle.com/c/home-credit-default-risk/data

- **application_{train|test}.csv**
  - This is the main table, broken into two files for Train (with TARGET) and Test (without TARGET).
  - Static data for all applications. One row represents one loan in our data sample.
- **bureau.csv**
  - All client's previous credits provided by other financial institutions that were reported to Credit Bureau (for clients who have a loan in our sample).
  - For every loan in our sample, there are as many rows as number of credits the client had in Credit Bureau before the application date.
- **bureau_balance.csv**
  - Monthly balances of previous credits in Credit Bureau.
  - This table has one row for each month of history of every previous credit reported to Credit Bureau – i.e the table has (#loans in sample * # of relative previous credits * # of months where we have some history observable for the previous credits) rows.
- **POS_CASH_balance.csv**
  - Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.
  - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (#loans in sample * # of relative previous credits * # of months in which we have some history observable for the previous credits) rows.
- **credit_card_balance.csv**
  - Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.
  - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (#loans in sample * # of relative previous credit cards * # of months where we have some history observable for the previous credit card) rows.
- **previous_application.csv**
  - All previous applications for Home Credit loans of clients who have loans in our sample.
  - There is one row for each previous application related to loans in our data sample.
- **installments_payments.csv**
  - Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.
  - There is a) one row for every payment that was made plus b) one row each for missed payment.
  - One row is equivalent to one payment of one installment OR one installment corresponding to one payment of one previous Home Credit credit related to loans in our sample.
- **HomeCredit_columns_description.csv**
  - This file contains descriptions for the columns in the various data files.


application_{train|test}.csv
Main tables – our train and test samples
Target (binary)
Info about loan and loan applicant at application time
SK_
SK_ID_CURR
bureau.csv
Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
One row per client's loan in Credit Bureau
SK_ID_CURR
SK_ID_PREV
SK_ID_BUREAU
bureau_balance.csv
Monthly balance of credits in Credit Bureau
Behavioral data
POS_CASH_balance.csv
Monthly balance of client's previous loans in Home Credit
Behavioral data

I have chosen the project for my capstone for following reasons:

i. Benefit the deserving customer to fulfil their dream of getting dream Home
ii. Home Credit to ensure that they are giving the loans to those customers who will repay the loan in time
iii. Myself as a participant to get the benefit of exposure to real life problem by applying knowledge gathered during Udacity Nanodegree

Problem Statement

The goal is to analyze the data provided by Home Credit and and process the data and train suing various machine learning/deep learning models and finally choose the model that gives the best performance which can be used for calculating the risk associated with the customer applying for loan.

The solution should maximize the ROC-AUC score for the test data given.

For submission to Kaggle, For each `SK_ID_CURR` in the test set, we have to predict a probability for the `TARGET` variable. The file should contain a header and have the following format:

```
SK_ID_CURR,TARGET
100001,0.1
100005,0.9
100013,0.2
```

The main steps involved are

i. Process and merge datasets and create a new dataset which can be processed
ii. Fill up missing values in data
iii. Cleanup the data which is not needed
iv. Analysis of data by trying different visualization techniques (cross tab bar graphs, violin graph, heatmap of linear correlation coefficients)
v. Perform transformation of data using log transformation, normalization, one-hot encoding
vi. Try different machine learning and deep learning techniques to train the data
vii. Choose the best model which has best ROC-AUC score but which also satisfies all the requirements (like it should be capable of generating actual probability)
viii. Get the prediction probability on the model using test data and transform to the format expected by Kaggle
ix. Submit to Kaggle and get the submission score

Metrics

This Kaggle competition is judged based on ROC-AUC score, so I have considered ROC-AUC score as the main metric for evaluating various models.
However, I have calculated Accuracy and F-score for completeness and for

comparing models.

The definitions of these metrics are given below:

ROC-AUC score: An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate. **True Positive Rate** (**TPR**) is a synonym for recall and is therefore defined as follows:

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

An ROC curve plots TPR vs. FPR at different classification thresholds.

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two dimensional area underneath the entire ROC curve. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0.

Accuracy: Accuracy is a common metric for binary classifiers. It takes into account both true positives and true negatives with equal weight.

accuracy = (true positives + true negatives)/dataset size

Precision:

precision = (true positives)/(true positives + false positive)

Recall:

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

F1-score:

F1-score = 2*(precision * recall)/(precision + recall)

Reference:
https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

II. Analysis

Data Exploration

i. Get statistics related to fields in master data (e.g CNT_CHIDREN, AMT_INCOME_TOAL,AMT_CREDIT, AMT_ANNUITY, AMT_GOODS_PRICE)

Find the statistics (mean, median, standard deviation) about these fields below:

SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	
count	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000
mean	278180.518577	0.080729	0.417052	11.909245	13.070108	10.067282	12.948771
std	102790.175348	0.272419	0.722121	0.488906	0.715193	0.549482	0.814384
min	100002.000000	0.000000	0.000000	10.152338	10.714440	0.000000	0.000000
25%	189145.500000	0.000000	0.000000	11.630717	12.506181	9.712630	12.382129
50%	278202.000000	0.000000	0.000000	11.899215	13.149068	10.122784	13.017005
75%	367142.500000	0.000000	1.000000	12.218500	13.603123	10.451522	13.429114
max	456255.000000	1.000000	19.000000	18.577685	15.214228	12.460818	15.214228

Interpretation: From this I get mean count of children is 0.08, median is 0 standard deviation is 0.2724, it means samples on a average less number of children and standard deviation is also low. It could be possible that most of the customers are either do not have family or could be single

For field AMT_INCOME_TOTAL mean: 0.417 median: 0 , std deviation: 0.72
But for AMT_CREDIT field is mean is: 11.909, median: 11.899, std deviation: 0.488

Ideally credit of a person should be less than 10 times to income but here there is a imbalance As ratio of mean of these AMT_CREDIT to 28.55, so need to be careful while approving loans

ii. Get statistics related to various loan information)

	CNT_CREDIT_PROLONG_ACTIVE	CNT_CREDIT_PROLONG_CLOSED	CNT_CREDIT_PROLONG_BAD_DEBT	CNT_CREDIT_PROLONG_SOLD	Active	Closed	Bad_debt	Sold	CASH_LOANS	CONSUMER_LOANS	REVOLVING_LOANS	XNA
count	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000

	CNT_CREDIT_PROLONG_ACTIVE	CNT_CREDIT_PROLONG_CLOSED	CNT_CREDIT_PROLONG_BAD_DEBT	CNT_CREDIT_PROLONG_SOLD	Active	Closed	Bad_debt	Sold	CASH_LOANS	CONSUMER_LOANS	REVOLVING_LOANS	XNA
mean	0.021820	0.012744	0.000003	0.000114	1.762275	2.984391	0.000065	0.018383	2.038184	2.033280	0.524755	0.001018
std	0.183987	0.133433	0.001803	0.010668	1.804891	3.359529	0.008064	0.146403	3.180878	1.828083	0.989541	0.034999
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000	3.000000	4.000000	0.000000	0.000000	3.000000	3.000000	1.000000	0.000000
max	9.000000	6.000000	1.000000	1.000000	32.000000	108.000000	1.000000	9.000000	60.000000	45.000000	31.000000	3.000000

Interpretation:
Event Active credit mean: 1.76, median: 1, std deviation: 1.80. Which means significant number of customers have a average of 1 or more existing loans

For Closed credit mean: 2.98 std deviation: median: 2 Std Deviation: 3.35
Which means lot of customers in training dataset have closed 2 or more loans, which is Good while approving loans

For Bad debt credit mean:0.000065, median: 0 std deviation: 0.00806, it means not many Customers have bad debt which is a good sign while approving loans

For cash loan mean: 2.08, median: 1.0 std deviation: 3.18
For consumer loan mean: 2.03, median: 2.0 std deviation: 1.8

I could not conclude much from this data

Event Rate:

Event rate percentage is calculated by dividing number of 1 in TARGET field by total number of records multiplied by 100. I found event rate have value 8.0728%

Interpretation :
I concluded that class is highly **imbalanced** as there are around 92% of values which are 0

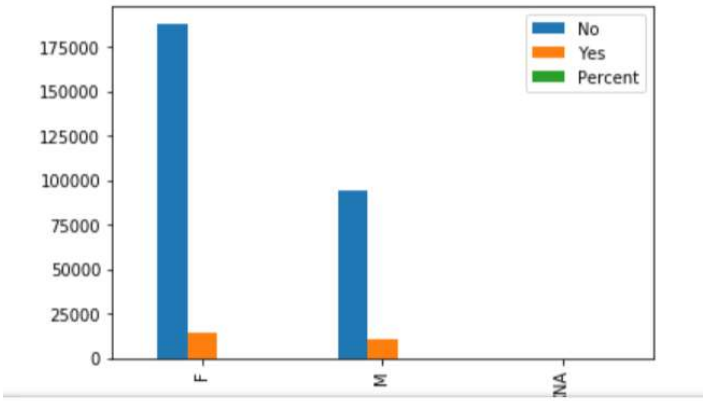
- *If a dataset is present for this problem, have you thoroughly discussed certain features about the dataset? Has a data sample been provided to the reader?*

Exploratory Visualization

(Note: I have included only a few visualizations for keeping the project report within 30 pages, rest of visualizations can be found with Jupyter Notebook for the project)

i. Crosstab values and bar plot between CODE_GENDER vs TARGET

CODE_GENDER	No	Yes	Percentage Yes
F	188278	14170	6.999328
M	94404	10655	10.141920
XNA	4	0	0.00000



Interpretation:

From the plot loan allocation for Male vs Female looks to be equally distributed

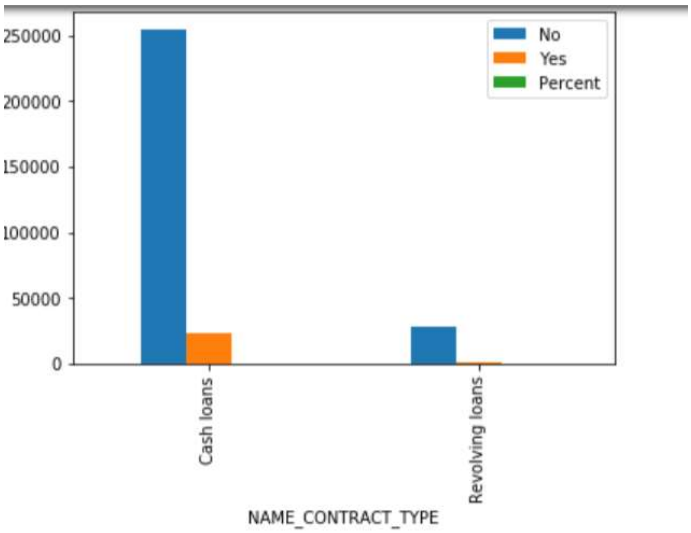
As Yes value % for Female is 6.99, dividing by event rate, lift value = 6.99/8.0728= 0.865

For Yes value % for Male10.14, dividing by event rate, lift value = 10.14/8.0728 = 1.25

As we do not see a significant difference in lift values, the CODE_GENDER is not much relevant for this project. Also, we do not want the algorithm to have any gender bias Because of legal issues in many countries, so we will drop the field CODE_GENDER

ii. Crosstab values and bar plot between NAME_CONTRACT_TYPE vs TARGET

NAME_CONTRACT_TYPE	No	Yes	Percentage Yes
Cash Loans	255011	23221	8.345913
Revolving Loans	27675	1604	5.478329



Interpretation:

From the plot loan allocation for Cash loans vs Revolving loans does not looks to be equally distributed

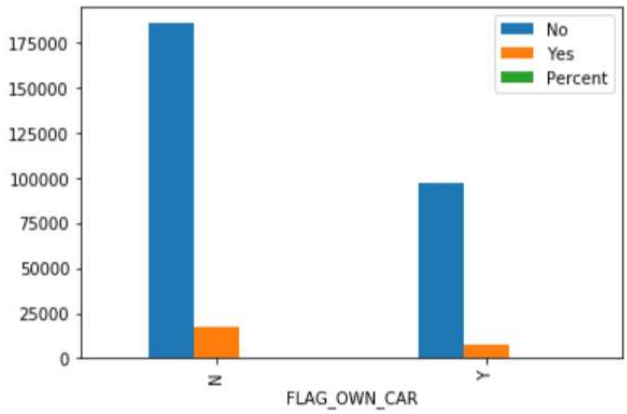
As Yes value % for Cash loans is 8.345913, dividing by event rate, lift value = 8.345913/8.0728= 1.033

For Yes value % for Revolving loans is 5.478329, dividing by event rate, lift value = 5.478329/8.0728 = 0.678

The NAME_CONTRACT_TYPE could be important feature to consider

iii. Crosstab values and bar plot between FLAG_OWN_CAR vs TARGET

NAME_CONTRACT_TYPE	No	Yes	Percentage Yes
Cash Loans	185675	17249	8.500227
Revolving Loans	97011	7576	7.243730



Interpretation:

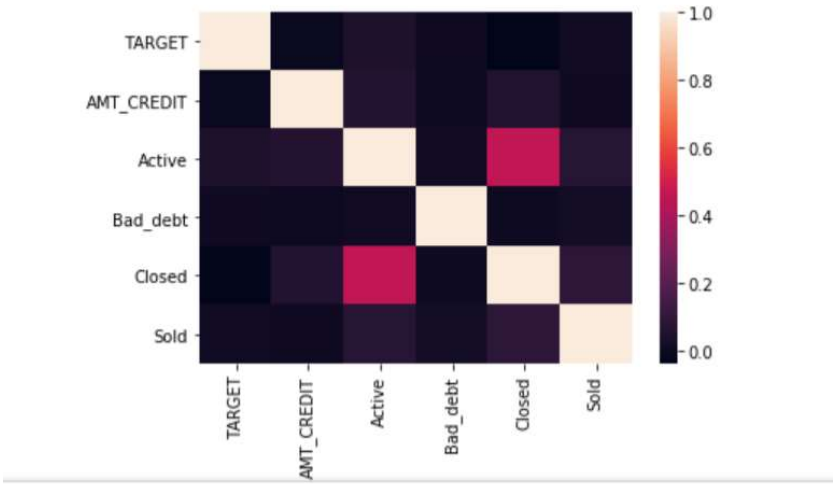
As Yes value % for FLAG_OWN_CAR N is 8.345913, dividing by event rate, lift value = 8.500227/8.0728= 1.0529

For Yes value % for FLAG_OWN_CAR N is 7.243730, dividing by event rate, lift value = 5.478329/8.0728 = 0.897

As lift values are not significantly different between FLAG_OWN_CAR N vs Y, this feature may not be very important

iv. Linear correlation coefficients and heat maps between features TARGET, AMT_CREDIT, Active, Bad_debt , Closed, Sold

	TARGET	AMT_CREDIT	Active	Bad_debt	Closed	Sold
TARGET	1.000000	-0.012181	0.043569	0.003531	-0.037233	0.009347
AMT_CREDIT	-0.012181	1.000000	0.061461	-0.003743	0.056674	0.007242
Active	0.043569	0.061461	1.000000	0.008212	0.455955	0.070171
Bad_debt	0.003531	-0.003743	0.008212	1.000000	0.002678	0.012759
Closed	-0.037233	0.056674	0.455955	0.002678	1.000000	0.084678
Sold	0.009347	0.007242	0.070171	0.012759	0.084678	1.000000

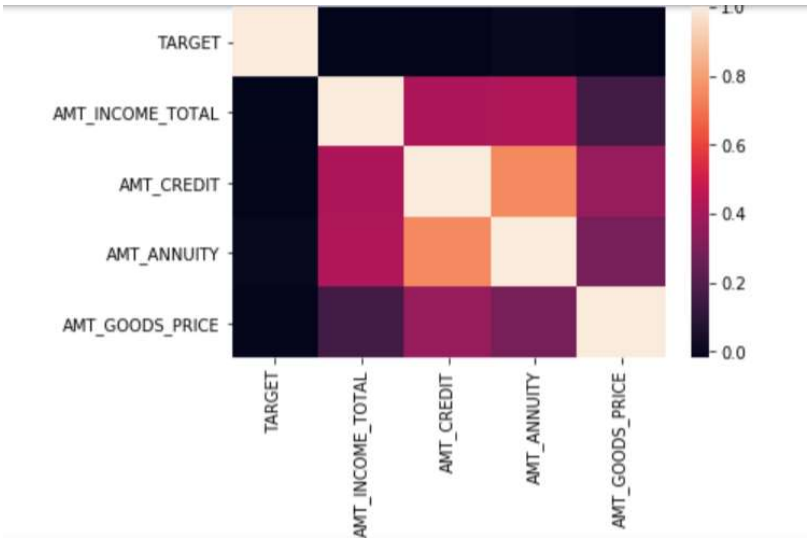


Interpretation:

From the heatmap only Active and Closed have linear relationship

v. Linear correlation coefficients and heat maps between features TARGET,AMT_INCOME_TOTAL,AMT_CREDIT,AMT_ANNUITY,AMT_GOODS_PRICE

	TARGET	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE
TARGET	1.000000	-0.017830	-0.010122	0.002893	-0.006468
AMT_INCOME_TOTAL	-0.017830	1.000000	0.419369	0.422166	0.161980
AMT_CREDIT	-0.010122	0.419369	1.000000	0.752855	0.367808
AMT_ANNUITY	0.002893	0.422166	0.752855	1.000000	0.290103
AMT_GOODS_PRICE	-0.006468	0.161980	0.367808	0.290103	1.000000



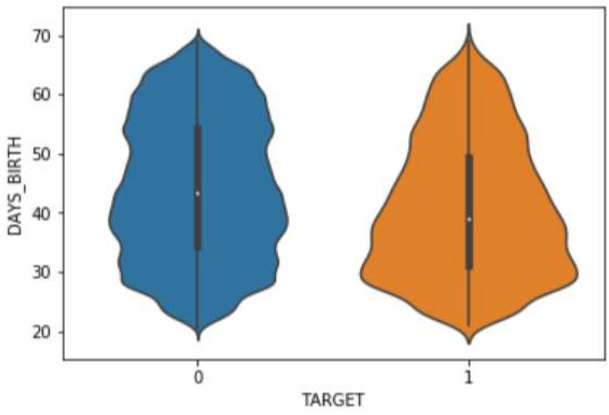
Interpretation:

From the heamap and correlation coefficients, found that AMT_ANNUITY is strongly dependent on AMT_CREDIT

Also, AMT_CREDIT, AMT_ANNUITY is dependent on AMT_INCOME_TOTAL Also, AMT_CREDIT is to some extent dependent on AMT_GOODS_PRICE.

I will drop the column AMT_ANNUITY

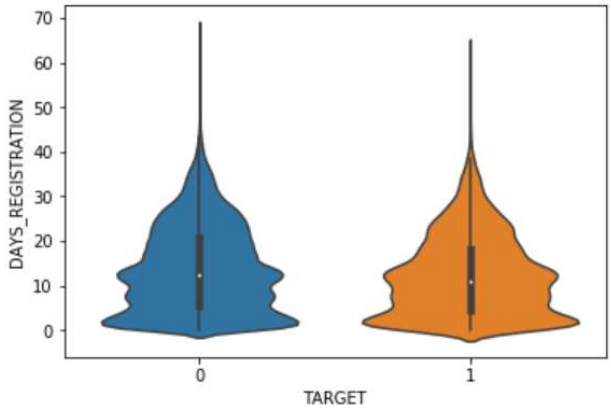
vi. Violin plot between DAYS_BIRTH and TARGET



Interpretation:

Most loans have been given around age 30 after that loan approval rate sequentially decreases so this is a important field to consider

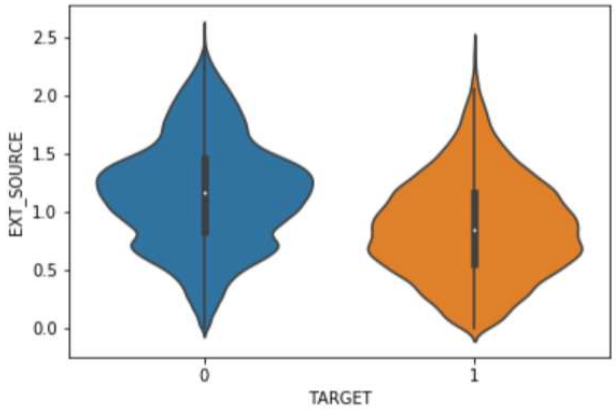
i. Violin plot between DAYS_REGISTRATION and TARGET



Interpretation:

TARGET i.e. Loan approval and not approval against DAYS_REGISTRATION almost same distribution, so it may not be a important feature to consider

ii. Violin plot between EXT_SOURCE and TARGET



Interpretation

Most loans have been when EXT_SOURCE is around 1.0 after that it decreases. But the distribution of loan approved and not approved vs EXT_SOUCE is almost similar. So could not conclude if the feature is important

Algorithms and Techniques

The following six traditional supervised learning models that are currently available in scikit-learn are used to train the data:

Decision Trees: Decision tree is used for prediction and assessing the relative importance of variables. for the current problem we will need to do prediction for home loan with many features, decision tree can be used.

Advantage of Decision Trees are:
i. Decision tree is simple to understand and interpret. People are able to understand decision tree models after a brief explanation.
ii. Decision tree have value even with little hard data. Important insights can be generated based on experts describing a situation (its alternatives, probabilities, and costs) and their preferences for outcomes.
iii. Allow the addition of new possible scenarios.

Logistic Regression: logistic regression is a simple model moves with non-linear function hence can work with linearly and non-linearly separable problems.

Advantages of Logistic Regression are:

- It does not assume a linear relationship between the Independent Variable and Dependent Variable
- It may handle nonlinear effects
- The Dependent variable need not be normally distributed.
- There is no homogeneity of variance assumption.

iii. **Gaussian Naive Bayes** (GaussianNB): Gaussian Naive Bayes is a simple but powerful algorithm for predictive modelling suitable for current problems.

iv. **Gradient Boosting:** Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

Most Kaggle competition winners use stack/ensemble of various models as it gives good performance, this is because it reduces both bias and variance. As currently problem is also from Kaggle so Gradient boosting is suitable for this.

The ensemble algorithms are robust to outliers, scalable, and able to naturally model non-linear decision boundaries thanks to their hierarchical structure.

v. **XGB Boosting**: XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. As this is also an ensemble boosting method, it gives good performance because it reduces both bias and variance. This is suitable for current problem

vi. **Random Forest**: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

As this is also uses ensemble method, gives good performance, suitable for current problem

For the initial training with all the models I have used default parameters. The idea is that once best model is found, it can be further refine using GridSearchCV technique

As a further define I have used Deep Neural Network as it gives better performance than any of traditional techniques

Benchmark

This problem is evaluated with ROC-AUC score. As this is a Kaggle problem, there is clear benchmark of Private Leadership score of 0.80570 obtained by Home Aloan team.

My model metrics can be compared against this score of 0.80570

III. Methodology

Data Pre-processing

In this step I have first all the loaded datasets as a pandas data frame. Using head function, check 5 records of each of dataframes to understand fields in each dataframe

- application_train.csv into application_train_data dataframe
- application_test.csv into application_test_data dataframe
- bureau.csv into bureau dataframe
- bureau_balance.csv into bureau_balance dataframe

- POS_CASH_balance.csv into POS_CASH_balance dataframe
- credit_card_balance.csv into credit_card_balance dataframe
- previous_application.csv into previous_application dataframe
- installments_payments.csv into installments_payments dataframe

By looking at data it appears that application_train.csv is the master dataset to be used for training, while application_test.csv which does not have field TARGET is used for evaluation by Kaggle. The data transformation and/or any other operation which is applied on that application_train.csv, the same transformation and/or operations are applied on application_tools.csv

SK_ID_CUR is the primary key that is used for identifying customers

I found that bureau.csv is an important dataset as it has the credit history information for the customers.

There are four types credit based on CREDIT_ACTIVE in bureau.csv (Active, Closed, Sold, Bad Debt). As each customers can more than one type of credit. So for each customer and each type CREDIT_ACTIVE I summarized into separate dataframes. Also, I created another dataframe which contain count of each type of Credit per customer. Next I merge all these dataframes with application_train, application_test by performing left join on field SK_ID_CUR.

I found that previous_application.csv is another important dataset. There are four important types of previous_apllication based on NAME_CONTRACT_TYPE and these are (i.e. Cash Loans, Consumer Loans, Revolving Loans, XNA). Also, each customer can have more than one type of loan. So for each customer and each type NAME_CONTRACT_TYPE I summarized into separate dataframes. Also, I created another dataframe which contain count of each type of NAME_CONTRACT_TYPE per customer. Next, I merge all these dataframes with application_train, application_test by performing left join on field SK_ID_CUR.

I have noticed that installments_payments, POS_CASH_balance, credit_card_balance are mainly transactional purpose, may not be very important. So, I am not considering these for further analysis

After all these operations I get the following:

- dataframe application_bureau_loan_train_data for training
- dataframe application_bureau_loan_test_data for testing

Missing Data Analysis

Missing data will create difficulty in computations of various statistical and machine learning operations.

In this step, we first get which all columns have missing values and then calculate percentage of records which have missing values in each column.

Next find out all the columns whose type is string and fill value 'NA' for all the missing values For remaining missing values which are numerical fill value 0

Fields Transformations

i. **Logarithmic Transformation**: For highly-skewed feature distributions such as AMT_INCOME_TOTAL', 'AMT_CREDIT', logarithmic transformation is done on the data so that the very large and very small values do not negatively affect the performance of a

learning algorithm. Using a logarithmic transformation significantly reduces the range of values caused by outliers..

ii. **Normalizing Numerical Features**

In addition to performing transformations on features that are highly skewed, it is often good practice to perform some type of scaling on numerical features. Applying a scaling to the data does not change the shape of each feature's distribution (such as AMT_INCOME_TOTAL, AMT_CREDIT above); however, normalization ensures that each feature is treated equally when applying supervised learners.

iii.**One hot encoding categorical features**

One hot encoding for categorical features Categorical variables having more than two possible vlaues are encoded using the one-hot encoding scheme. One-hot encoding creates a "dummy" variable for each possible category of each non-numeric feature. For example, assume someFeature has three possible entries: A, B, or C. We then encode this feature into someFeature_A, someFeature_B and someFeature_C.

iv. **Label Encoding**: Categorical variables having more than two possible are encoded using Label Encode to have values 0 and 1

v. **Other transformation**: values Fields EXT_SOURCE_1, EXT_SOURCE_2, EXT_SOURCE_3 are summed to create a new field EXT_SOURCE and original fields are dropped

v. **Drop fields** which are not needed for training: Some of the fields that are removed are

CODE_GENDER,
WEEKDAY_APPR_PROCESS_START,
HOUR_APPR_PROCESS_START,
REGION_POPULATION_RELATIVE,
FLAG_MOBIL,
FLAG_EMP_PHONE,
FLAG_WORK_PHONE,
FLAG_CONT_MOBILE,
FLAG_PHONE,
FLAG_EMAIL

In addition SK_ID_CUR field is removed before training data because it is unique for each customer

Implementation

Applying Machine Learning on the processed data

i. Cross Validation¶

In this step, data is split between training and test with 70% data is used for training and 30% is used for Test. For doing this train_test_split method of sklrean.cross_validation module is used

ii. Naive Predictor Performance¶

The purpose of generating a naive predictor is simply to show what a base model without any intelligence would look like. In the real world, ideally your base model would be either the results of a previous model or could be based on a research paper upon which you are looking to improve. When there is no benchmark model set, getting a result better than random choice is a place we could start from.

iii. Train using Machine Learning

In this step, all the machine learning models are applied to fit the training data. Next prediction is done using the test data and evaluation metrics of ROC_AUC score, Accuracy Score and F1-score are evaluated, also training time and prediction time are noted

The six machine learning algorithm are tried in two phases. In the first phase Decision Tree, Gaussian Naïve Bayes and Logistic Regression is used.

In the second phase Gradient Boost, XGBoost and RandomForest are used.

Once evaluations in each phase is over, metrics are shown graphically

Next, comparison is done on primarily ROC-AUC score for each of models. Other metrics (accuracy score, F1-score) are also considered as a secondary evaluation metrics. One important criteria to consider is the model should generate actual probability using predict_proba function on the model as this is needed by Kaggle.

iv. Preparing to submit to Kaggle¶

Following steps are for preparing data for submission to Kaggle

- Use the predict_proba to the the GradientBoost model to get the probabily
- Get SK_ID_CURR field of test_data and store as test_data_id
- Merge first column of probability and merge with test_data_id
- Save the dataframe to the CSV file
- Submit to Kaggle and get the score

Refinement

I initially tried to tune GradientBoost using GridSearchCV using different parameters 'n_estimators': [100, 150, 200], 'learning_rate':[0.5, 0.2, 0.1], 'max_depth': [3,6]

But for some reason, Jupyter notebook was hanging. After multiple attempts, started evaluating Deep Learning Network.

As deep neural network gives good performance, for refining the model, I created a 3 -layer Deep Neural Network with 256 x 256 as 2 Hidden layer with Mini-batch Gradient Descend the following parameters:

- Batch size = 256
- AdamOptimizer as optimizer
- Learning rate = 0.001
- Epochs = 25
- Keep probability = 0.8

With these I got the following results:

ROC-AUC score : 0.70
Testing Accuracy: 0.92038286
Training Accuracy: 0.9187715

IV. Results

Model Evaluation and Validation

Below are the metrics results that I got when I applied traditional machine learning models on the test data

Metric	Decision Tree	Logistics	Gaussian NB	XGBBoost	GradientBoost	RandomForest
ROC-AUC (test)	0.537319	0.502448	0.557352	0.501365	0.503695	0.500826
ROC-AUC (300 training)	1.0	0.521739	0.564746	0.5	0.498194	0.956521
Accuracy (test)	0.85338	0.920361	0.325134	0.920545	0.920480	0.920470
Accuracy (300	0.85338	0.926666	0.306667	0.923333	0.92	0.993333

Metric	Decision Tree	Logistics	GaussianNB	XGBBoost	GradientBoost	RandomForest
training)						
F1 score (test data)	0.149094	0.010771	0.164253	0.005697	0.015830	0.003531
F1 score(300 training)	1.0	0.083333	0.161290	0.0	0.0	0.954545
Training Time	24.800899	23.82941	1.342295	117.141126	166.722213	28.703508
Prediction Time	0.171837	0.109350	0.623674	0.922175	0.567169	1.050263

From the above result we can see the ROC-AUC score best model is GaussianNB having score 0.557352. However, we can not take this model as Accuracy is very low (only 0.32) also in sklearn this model does not provide the actual probability.

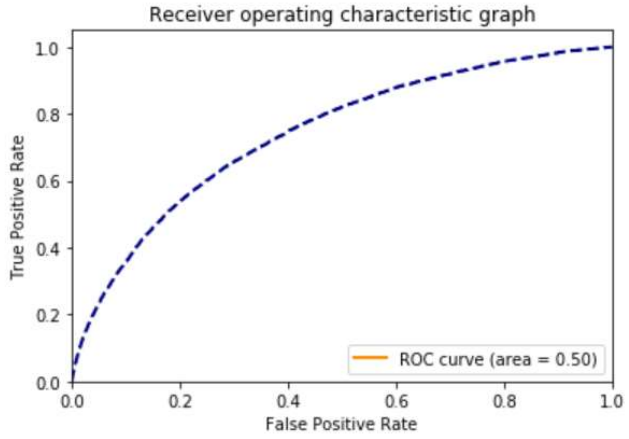
The next best model is: Decision tree having ROC-AUC score of 0.537319 and accuracy score 0.85338. But I could not consider this model, as Decision Tree model has overfitting issues. Another main reason is that it does not provide the actual probability.

Next best model is: GradientBoost which has good ROC-AUC score of 0.503695 and accuracy score of 0.920480. Although, scores of XGBBoost is comparable with GradientBoost. But I would still go with GradientBoost, becuase it has better F1 score (0.015830) , than XGBBoost (0.005697)

GradientBoost has following advantages:

- Gradient Boost is a ensemble method which performs very well. This is used for winners of Kaggle competitions.
- Gradient Boost build trees one at a time, where each new tree helps to correct errors made by previously trained tree. With each tree added, the model becomes even more expressive. There are typically three parameters - number of trees, depth of trees and learning rate, and the each tree built is generally shallow.
- The only problem with Gradient Boost is that it can overfit but it can be overcome by tuning leaning rate.

ROC Curve with the below is as below:

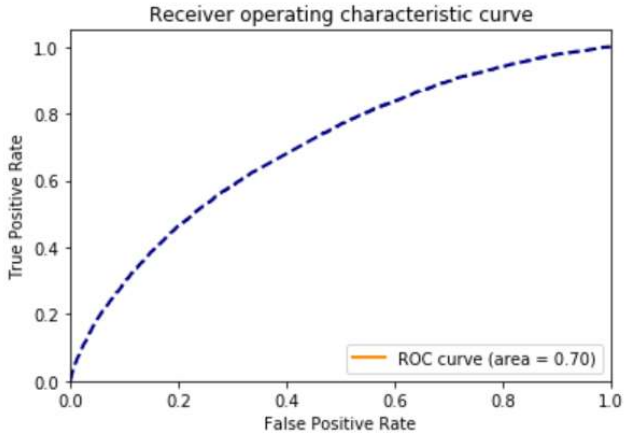


Kaggle Score

When I submitted to Kaggle My score was (0.55958) is as shown below:¶

If image below is not visible, please use the URL:

https://drive.google.com/open?id=10CulFKD6OA21edTiAHSzeI1gbfxClhb9



But Neural Network provides better performance and has the following advantages:

- Deep Neural is responsible for the rapid development that's going on in the Machine Learning industry right now. Before Deep learning, we were not nearly as good at stuff like image classification and speech recognition as we are today. Just look at all the things around you that are powered by deep learning - your fitbit, Siri, Google Home, Amazon Alexa, and so many more.

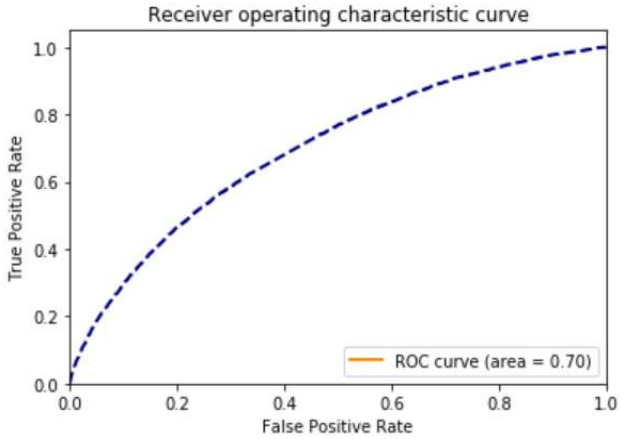
- Secondly, ANNs provided us the first step towards AI by generating a model based on how our own human body learns. Through mimicking neuron interaction within the body, researchers about 20 years ago were actually able to conquer something that had never been done before. Before neural nets, there were very few, if at all, models that were actually trained on how our body

(Reference: https://www.quora.com/What-are-the-advantages-disadvantages-of-Artificial-Neural-networks)

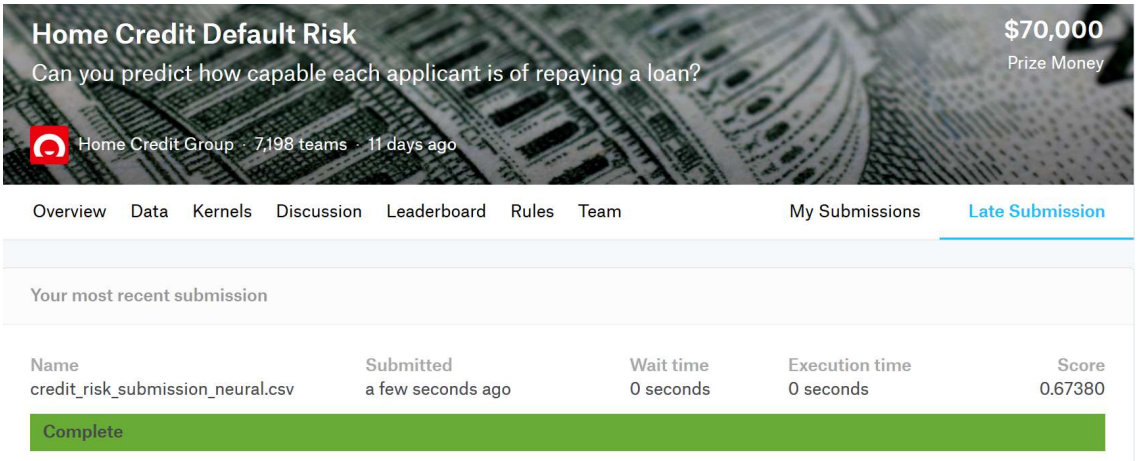
I further refined and used 3 -layer Deep Neural Network with 256 x 256 hidden layers, I got the following results:

ROC-AUC score : 0.70
Testing Accuracy: 0.92038286
Training Accuracy: 0.9187715

ROC Curve is as below:



When I did Kaggle Submission, I got score of 0.67380
If image is not visible, please use link is below:
https://drive.google.com/file/d/1JoPdA8VYwjJXkFNJu5_WzANr-BbmZzot/view



As this was late submission, I did not get the leadership ranking. But if I had submitted the same before deadline, I would have got ranking of 6443 (for both public and private leadership)

Note: The Highest score in Private Leadership is 0.80570 and total number of teams: 7198

Justification

So my final Model is Deep Neural Nework with 256 x 256 hidden layers with Mini-batch Gradient Descend the following parameters:

- Batch size = 256
- AdamOptimizer as optimizer
- Learning rate = 0.001
- Epochs = 25
- Keep probability = 0.8

As observed, this has provided the best results for me:

ROC-AUC score : 0.70
Testing Accuracy: 0.92038286
Training Accuracy: 0.9187715
Kaggle Score: 0.67380

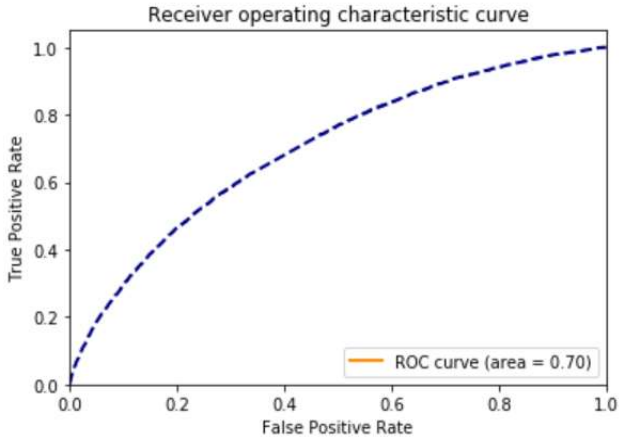
This is clearly far better results than when I used GradientBoost

Benchmark here is Kaggle score of 0.80570

V. Conclusion

Free-Form Visualization

I have included the ROC curve for final Deep Neural Network model with ROC-AUC score of 0.70 as below:



Final Kaggle score is 0.67380 which is a decent score given this is my first competition in Kaggle. Thanks a lot to Udacity for providing platform compete in Kaggle

Reflection

The project is about credit analysis of customers of Home Credit. I have taken the data provided by Kaggle, analysed, refined and then applied 6 traditional machine learning algorithm and then finally Deep Neural Network which gave the better performance than traditional models and I have chosen this as the final model.

In this project, I have gone through the entire development cycle machine learning including cleaning, processing, analysing, training different models and refine to choose the best model.

The initial difficulty that I faced is some reason I was not able to use GridSeachCV for optimizing GradientBoost as whenever I tried my Jupyter Notebook got hung.

But finally, I had choosen, Deep Neutral Network which gave far better performance than any of traditional techniques. For training using Deep Neutral Network I had initial difficulty while using GPU in my laptop but I fixed those issues

Improvement

There is always scope for improvement as Aleady some Kaggles obtained score of 0.80. I can further deep neural network model by tuning the following parameters/hyper parameters:

- Learning rate
- Keep probability
- Adding more layers
- Early stop
- Adjusting epochs