Analyzing Bank Marketing Dataset using Different Optimization Algorithms in Neural Net Environment

Dipesh Chand
Faculty of Computer Science
Østfold University College
1783 Halden,Norway
dipesh.chand@hiof.no

Bibek Acharya
Faculty of Computer Science
Østfold University College
1783 Halden,Norway
bibek.acharya@hiof.no

***Abstract*—This project applied different optimization algorithms techniques in Neural Net environment to build the model to predict whether the customer will subscribe bank long-term deposit or not. A Portuguese retail bank collected data from 2008 through 2013. We will analyze the small set of data related to the bank client based on telephone communication. The Portuguese Bank had an issue of revenue declined, so they conducted a survey and campaign to identify existing clients that have higher chance to subscribe for term deposit and focus marketing effort of such customers. A customer-based analysis of banking services allows for understanding of the possible effects of the concentration on a wide variety of banking resources into a small group of national enterprises. This kind of study projects could be helpful to determine the likelihood of procurement of financial services.**

Keywords—Bank Marketing, Neural Network, ANN

I. INTRODUCTION

Businesses today often launch marketing campaigns to boost the sale of their products and services. While digital marketing becomes popular and have many advantages over traditional marketing, traditional marketing methods are still providing physical customer experience difficult to be offered by digital marketing and could not be completely replaced by digital marketing 1 . One of the drawbacks of traditional marketing is that it is typically more expensive than digital marketing since it involves one or multiple types of activities such as phone calls, customer visits, or physical prints etc. These activities often require significant efforts and investment from businesses. Therefore, for traditional marketing, it is important to target marketing activities towards desirable customers who are more likely to buy products and services than others. It will not be cost-effective if marketing campaign targets are simply randomly chosen without going through thorough review and selection.

Machine learning can provide a data-driven approach to help marketing campaign more targeted to desirable customers. In this project, using bank telemarketing as an example of traditional marketing, a machine learning model was developed and demonstrated its effectiveness in maximizing business return while minimizing marketing effort.

This paper is structured as follows:

Chapter 2 presents the problem statement and dataset description. Chapter 3 provides description about different related work. Chapter 4 explains about the different optimizing algorithms used to perform the classification on this project. Chapter 5 explores about the result of the classification model and compares them with the results of our project on classification model. Chapter 6 pointing forward to conclusion and several possibilities in this project aim which can be performed in near future.

II. PROBLEM STATEMENT AND DATASET DESCRIPTION

The most effective strategy to progress a business marketing at a least conceivable overhead is constantly seen as the fundamental issue by the supervisor. The foremost tremendous part of the challenge is to recognize the promising and potential clients of the displaying thing with limited data. Realizing that specific information which chooses the promising and potential client would empower executive to put more assets on positive portion towards the items and cut down the budget spent on non-promising client, so that to dispose of bottlenecks and make a progressively productive advancing way.

The dataset is related to direct marketing campaigns run by the Portuguese bank and

contains information on various features of interest for approximately 41,188 customers. The dataset has been taken from the UCI machine learning repository [1].

The features of interest can be broken down as follows:

age - Age of the client- (numeric)

job - Client's occupation - (categorical) (admin, bluecollar, entrepreneur,housemaid, management, retired, selfemployed, services, student, technician, unemployed, unknown)

marital - Client's marital status - (categorical) (divorced, married, single, unknown, note: divorced means divorced or widowed)

education - Client's education level - (categorical) (basic.4y, basic.6y, basic.9y, high.school, illiterate, professional.course, university.degree, unknown)

default - Indicates if the client has credit in default – (categorical) (no, yes, unknown)

housing - Does the client as a housing loan? - (categorical) (no, yes, unknown)

loan - Does the client as a personal loan? - (categorical) (no, yes, unknown')

contact - Type of communication contact - (categorical) (cellular, telephone)

month - Month of last contact with client - (categorical) (January- December)

day of week - Day of last contact with client - (categorical)
(Monday - Friday)

duration - Duration of last contact with client, in seconds - (numeric) For benchmark purposes only, and not reliable for predictive
modeling

campaign - Number of client contacts during this campaign -
(numeric) (includes last contact)

pdays - Number of days from last contacted from a previous campaign - (numeric) (999 means client was not previously contacted)

previous - Number of client contacts performed before this campaign - (numeric)

poutcome - Previous marketing campaign outcome - (categorical) (failure, nonexistent , success)

emp.var.rate - Quarterly employment variation rate - (numeric)

cons.price.idx - Monthly consumer price index - (numeric)

cons.conf.idx - Monthly consumer confidence index - (numeric)

euribor3m - Daily euribor 3 month rate - (numeric)

nr.employed - Quarterly number of employees - (numeric)

Output variable (desired target) - Term Deposit – subscription verified (binary: 'yes'or 'no')

The initial impression that can be created using the dataset are as:

- Total 41188 records
- 10 numeric attributes : age, duration, campaign, pdays, previous, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed
- 10 Factors:
  - 10 multi-valued categorical attributes : job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome
  - 1 categorical target attribute y
- No missing values: Preprocessing should be easier.

III. RELATED WORK

Moro, Cortez, and Rita [2] has utilized semi-automatic modeling procedure. In their study they have picked information from July 2012 and utilized 22 highlights of data. They have compared about 4 distinctive sort of data mining model. They are "logistic regression, decision trees, neural network (NN) and support vector machine". There are two measurements utilized in the investigation known as AUC (area of the receiver operating characteristic curve) and ALIFT (area of LIFT cumulative curve). The testing model was utilized in advancement stage utilizing the most recent information of July 2012 and another model called a rolling window scheme. The research has shows the result of AUC as 0.8 and ALIFT as 0.7. This has permitted 79% of subscribers as half possible customers. There two extraction strategies are utilized which are sensitive analysis and DT. They were applied to NN. This uncovered several key traits which was tenable and important to telemarketing managers for the campaign.

Vajiramedhin and Suebsing [3] suggested that the performance of the predictive model with the number of smaller features can be improved. Their experiment on Direct Bank Marketing dataset can enhance the predictive model performance both of the TP rate and the ROC rate while it employs the smaller storage space, reduces the computation time and gains the higher predictive performance.

Another study Elsalamony and Elsayad [4], evaluate and compare the classification

performance of the two different techniques models Multilayer perceptron neural network (MLPNN) and C5.0 on the bank direct marketing dataset to classify for bank deposit subscription. In their study they used statistical measures; Classification accuracy, sensitivity and specificity and found that C5.0 has slightly better performance than MLPNN. And also Importance analysis has shown that attribute "Duration" in both models has achieved the most important attribute.

IV. METHODOLOGY

This project is follow up project of our last project "Analyzing Bank Marketing Dataset using Different Classification Algorithms". In this project we are using the power of Neural Network with the combination of different optimization algorithms. The objective of this project is to predict using Neural Net technique if the client will subscribe to a Term Deposit and to improve the performance of model and increase the classification accuracy using various approaches. In order to obtain more accurate and precise model to predict desired output, we will use several optimization algorithms such as AdaDelta, AdaGrad, Adam, Nadam, RMSprop and SGD. After we perform all of the above techniques, we would be able to understand the data and suggest the best fit model for prediction of "customer term deposit" more accurately and precisely.

Since we are working on the same dataset in our last project and this project and this project is follow up project of our first project, we have decided to utilize some of the work done in our previous project. We have done lots of data understanding and exploring and one of our objective is to make comparison between outcome s of this project to outcomes of our last project so we considered to more focus upon data modeling part rather than data understanding and exploring. And also we decided to utilize the outcomes of data pre-processing. ie. removing the 'euribor3m' variable before performing any further activities based on analysis Correlation factor and VIF.

In this project we decided to use python as a programming language and implemented keras which is a high-level neural networks API, capable of running on top of Tensorflow or Theanos [5] . We used Keras running on top of Tensorflow in this project because it is user friendly, modular, easy to extend, and to work with Python. Neural layers, cost functions, optimizers, initialization schemes, activation functions, and regularization schemes are all standalone modules that we can combine to create new models. New modules are simple to add, as new classes and functions. Models are defined in Python code, not separate model configuration files [6].

Also we used scikit-learn which is library in Python that provides many unsupervised and supervised learning algorithms. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy [7].

We perform the following steps in our study:

1. Data Pre-processing
2. Data Modeling

A. DATA PRE-PROCESSING

As we already mentioned we follow some of the outcome from our previous project so we are not including 'euribor3m' variable in this project also. Further more the duration variable is also not included in the modeling just like in our previous project based on the following dataset author's notes.

Important note: *This attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model [4].*

1) Data Preparation

In our dataset we have 10 multi-valued categorical attributes : job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome and 1 categorical target attribute y

Categorical data are usually more challenging to deal with than numerical data. In particular, many machine learning algorithms require that their input is numerical and therefore categorical features must be transformed into numerical features before we can use any of these algorithms [8]. So we use scikit-learn's LabelEncoder to encode a pandas DataFrame of string labels which converts each class under specified feature to a

numerical value [9]. Label encoding is simply converting each value in a column to a number. It is quite easy to roll your own label encoder that operates on multiple columns of your choosing, and returns a transformed dataframe. Creating a custom encoder involves simply creating a class that responds to the fit(), transform(), and fit_transform() methods. Using this technique we transform all 10 multi-valued categorical attributes and 1 binary target attributes. Figure 1 explains about the attributes before and after implementing encoding respectively.

```
In [21]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 19 columns):
age               41188 non-null int64
job               41188 non-null int64
marital           41188 non-null int64
education         41188 non-null int64
default           41188 non-null int64
housing           41188 non-null int64
loan              41188 non-null int64
contact           41188 non-null int64
month             41188 non-null int64
day_of_week       41188 non-null int64
campaign          41188 non-null int64
pdays             41188 non-null int64
previous          41188 non-null int64
poutcome          41188 non-null int64
emp.var.rate      41188 non-null float64
cons.price.idx    41188 non-null float64
cons.conf.idx     41188 non-null float64
nr.employed       41188 non-null float64
y                 41188 non-null int64
dtypes: float64(4), int64(15)
memory usage: 6.0 MB
```

Fig.1. *Attributes description after implementing Encoding*

We retrieve source data with the help of pandas and then we encoded the categorical attributes into numerical and data is still pandas dataframe. So we convert a Pandas data frame to the x,y inputs that TensorFlow needs.

2) Data Splitting

The dataset is divided into training data and test data with the intention of using the training data to find the parameters of the particular model being used (fitting the model on the training data) and then applying this to the test data to determine the model's performance and to draw conclusions about its predictive capability. This can be done with a train_test_split class of sklearn library by specifying test_size for test set and the remaining will be training set. In this project the dataset is split into training set and testing set with 80% for training and 20% for testing. There is third dataset which we utilized in this project, validation set which we created at the time of model creation. We divided the validation set to 20% from training set instead of overall dataset.

Training Dataset: The sample of data used to fit the model.
Validation Dataset: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.
Test Dataset: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset [10].

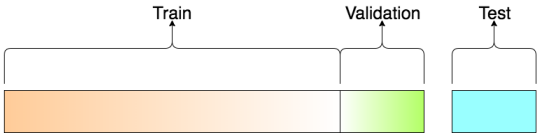

Fig.2. *Description showing the splitting of dataset*

3) Data Scaling

A generic dataset is made up of different values which can be drawn from different distributions, having different scales and, sometimes, there are also outliers. A machine learning algorithm isn't naturally able to distinguish among these various situations, and therefore, it's always preferable to standardize datasets before processing them.
To solve this we use scikit-learn's StandardScaler. The idea behind StandardScaler is that it will transform your data such that its distribution will have a mean value 0 and standard deviation of 1. Given the distribution of the data, each value in the dataset will have the sample mean value subtracted, and then divided by the standard deviation of the whole dataset.

B. DATA MODELING

In order to model the data, we used six optimization algorithms in this project to predict the term deposit subscription.

1. AdaDelta
2. AdaGrad
3. Adam
4. Nadam
5. RMSprop
6. SGD

1) AdaDelta

Adadelta is common optimization algorithm that helps improve the chances of finding useful solutions at later stages of iteration, which is difficult to do when using the Adagrad algorithm

for the same purpose[11]. The interesting thing is that there is no learning rate hyperparameter in the Adadelta algorithm. It uses an EWMA (Exponentially Weighted Moving Average) on the squares of elements in the variation of the independent variable to replace the learning rate. The Adadelta algorithm uses the variable s_t, which is an EWMA on the squares of elements in mini-batch stochastic gradient g_t. At time step 0, all the elements are initialized to 0. Given the hyperparameter $0 \leq \rho < 1$, at time step $t > 0$.

$$s_t \leftarrow \rho s_{t-1} + (1-\rho) g_t^2$$

The Accuracy and Loss of AdaDelta Algorithm Model in our modeling as compare to train and Validation data can be viewed in Figure 3.

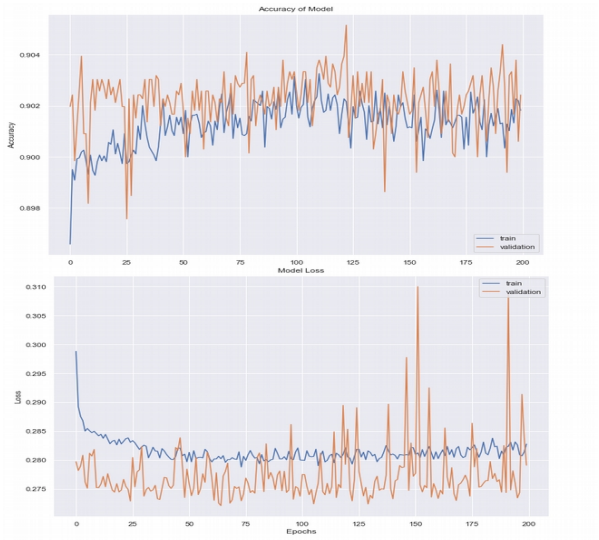

Fig.3. Curve showing Accuracy and Loss of AdaDelta Algorithm Model as compare to train and Validation data

2) AdaGrad

Adagrad, an algorithm that adjusts the learning rate according to the gradient value of the independent variable in each dimension to eliminate problems caused when a unified learning rate must adapt to all dimensions [12]. It constantly adjusts the learning rate during iteration to give each element in the independent variable of the objective function its own learning rate. When using Adagrad, the learning rate of each element in the independent variable decreases or remains unchanged during iteration. The Adadelta algorithm uses the cumulative variable s_t obtained from a square by element operation on the mini-batch stochastic gradient g_t. At time step 0, Adagrad initializes each element in s_0 to 0. At time step t, we first sum the results of the square by element operation for the mini-batch gradient g_t to get the variable s_t:

$$s_t \leftarrow s_{t-1} + g_t^2$$

The Accuracy and Loss of AdaGrad Algorithm Model in our modeling as compare to Training and Validation data can be viewed in Figure 4.

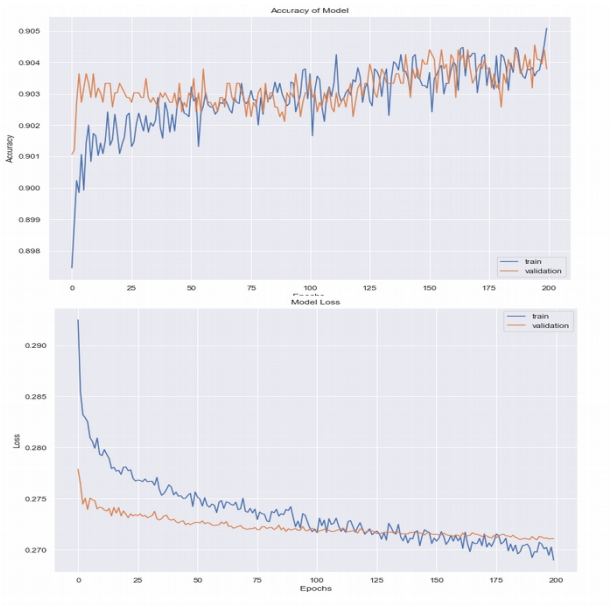

Fig.4. Curve showing Accuracy and Loss of AdaGrad Algorithm Model as compare to train and Validation data

3) Adam

Adam, an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments [13]. Adam uses the momentum variable v_t and variable s_t, [14] which is an EWMA on the squares of elements in the mini-batch stochastic gradient from RMSProp and initializes each element of the variables to 0 at time step 0. Given the hyperparameter $0 \leq \beta_1 < 1$, the momentum variable vt at time step t is the EWMA of the mini-batch stochastic gradient g_t.

$$v_t \leftarrow \beta_1 v_{t-1} + (1-\beta_1) g_t.$$

$$s_t \leftarrow \beta_2 s_{t-1} + (1-\beta_2) g_t^2.$$

The Accuracy and Loss of Adam Algorithm Model in our modeling as compare to Training and Validation data can be viewed in Figure 5.

Fig.5. Curve showing Accuracy and Loss of Adam Algorithm Model as compare to train and Validation data

4) Nadam

Nadam (Nesterov-accelerated Adaptive Moment Estimation) is combination of Adam and NAG. In order to incorporate NAG (Nesterov accelerated gradient) into Adam, the momentum term is modified [15].

The Accuracy and Loss of Nadam Algorithm Model in our modeling as compare to Training and Validation data can be viewed in Figure 6.

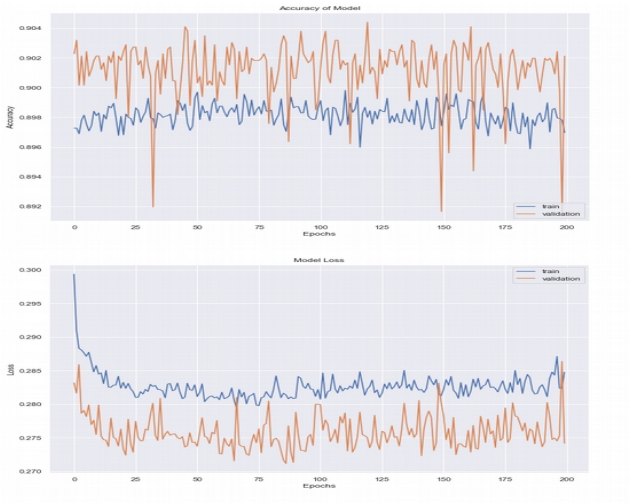

Fig.6. Curve showing Accuracy and Loss of Nadam Algorithm Model as compare to train and Validation data

5) RMSProp

The RMSProp is the commonly used optimizing algorithm in machine learning. It is the updated version of Adagrad which uses an EWMA on the squares of elements in the mini-batch stochastic gradient to adjust the learning rate [16]. The state variable s_t is the sum of the square by element all the mini-batch stochastic gradients g_t up to the time step t, RMSProp uses the exponentially weighted moving average on the square by element results of these gradients. Specifically, given the hyperparameter $0 \leq \gamma < 1$, RMSProp is computed at time step $t > 0$

$$s_t \leftarrow \gamma s_{t-1} + (1-\gamma) g_t^2$$

The Accuracy and Loss of Adam Algorithm Model in our modeling as compare to Training and Validation data can be viewed in Figure 7.

Fig.7. Curve showing Accuracy and Loss of RMSProp Algorithm Model as compare to train and Validation data

6) SGD

SGD (Stochastic Gradient Descent) is an is an iterative method for optimizing. Stochastic gradient descent reduces computational cost at each iteration. Stochastic gradient descent updates the weight parameters after evaluation of the cost function after each sample. Rather than summing up the cost function results for all the sample then taking the mean, it updates the weights after every training sample is analyzed [17].

The Accuracy and Loss of SGD Algorithm Model in our modeling as compare to Training and Validation data can be viewed in Figure 8.

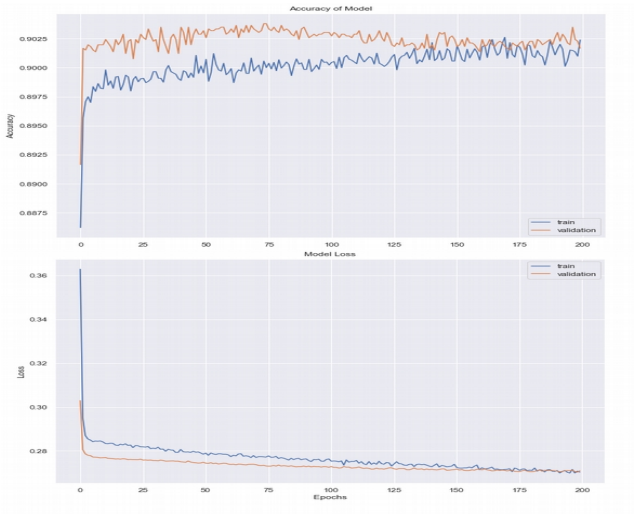

Fig.8. Curve showing Accuracy and Loss of SGD Algorithm Model as compare to train and Validation data

With these optimization algorithm we build our Neural Net. We construct Neural net of 3 hidden layer. Overall our Neural Network has 320 nodes with 22800 edges distributed as 18 Nodes for Input Layer, 150 nodes, 100 nodes and 50 nodes for First hidden layer, Second hidden layer and Third hidden layer respectively and 2 nodes for Output layer. The Neural Net model can be modeled as in Figure 7.We used dropout layer between each hidden layer with dropout rate of 20%. In total there are 3 dropout layer – First Dropout Layer between First and Second Hidden Layer, Second Dropout Layer between Second and Third Hidden Layer and Third Dropout Layer between Third Hidden Layer and Output Layer. Dropout is a technique where randomly selected neurons are ignored during training. They are "dropped-out" randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass [18]. Dropout is an approach to regularization in neural networks which helps reducing interdependent learning among the neurons. The effect of this is that neurons are prevented from co-adapting too much which makes over fitting less likely.

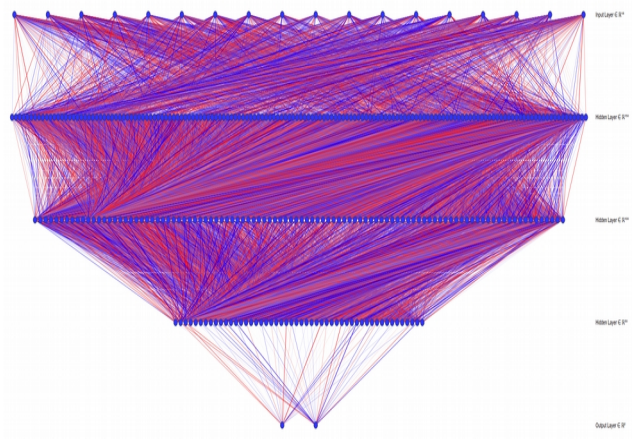

Fig.9. Neural Net model

V. Model Evaluation and Result

We performed six different optimizing algorithms in Neural Net models to classify whether a customer would open a bank account or not. We analyze the Loss and Accuracy of model. Plots of Loss and Accuracy of models with reference to Epochs can be viewed in Figure 3, Figure 4, Figure 5, Figure 6, Figure 7 and Figure 8. K-Folds cross validation using function of Sklearn library were also performed with the different value of K and found the best output with value as 5 so we perform all the cross validation with value as 5 and calculated their mean accuracy. We also consider Confusion Matrix metrics for evaluation which is a breakdown of predictions into a table showing correct predictions (the diagonal) and the types of incorrect predictions made (what classes incorrect predictions were assigned). Figure 8, shows confusion matrix of different classification models we used in this study.

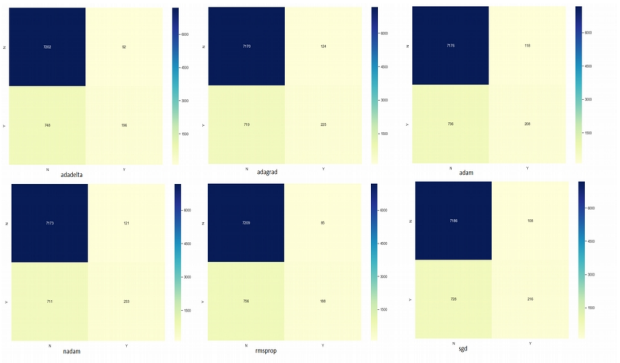

Fig.10. Confusion Matrix of all the Algorithm Model

The Figure 11, Figure 12, Figure 13, Figure 14, Figure 15, Figure 16 shows the classification report of the respective Modeling done in this study.

Classification report:

	precision	recall	f1-score	support
0	0.91	0.99	0.94	7294
1	0.68	0.21	0.32	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.79	0.60	0.63	8238
weighted avg	0.88	0.90	0.87	8238

Fig.11. Classification report of AdaDelta Algorithm Model

Classification report:

	precision	recall	f1-score	support
0	0.91	0.98	0.94	7294
1	0.64	0.24	0.35	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.78	0.61	0.65	8238
weighted avg	0.88	0.90	0.88	8238

Fig.12. Classification report of AdaGrad Algorithm Model

Classification report:

	precision	recall	f1-score	support
0	0.91	0.98	0.94	7294
1	0.64	0.22	0.33	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.77	0.60	0.64	8238
weighted avg	0.88	0.90	0.87	8238

Fig.13. Classification report of Adam Algorithm Model

Classification report:

	precision	recall	f1-score	support
0	0.91	0.98	0.95	7294
1	0.66	0.25	0.36	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.78	0.62	0.65	8238
weighted avg	0.88	0.90	0.88	8238

Fig.14. Classification report of Nadam Algorithm Model

Classification report:

	precision	recall	f1-score	support
0	0.91	0.99	0.94	7294
1	0.69	0.20	0.31	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.80	0.59	0.63	8238
weighted avg	0.88	0.90	0.87	8238

Fig.15. Classification report of RMSPro Algorithm Model

Classification report:

	precision	recall	f1-score	support
0	0.91	0.99	0.95	7294
1	0.67	0.23	0.34	944
micro avg	0.90	0.90	0.90	8238
macro avg	0.79	0.61	0.64	8238
weighted avg	0.88	0.90	0.88	8238

Fig.16. Classification report of SGD Algorithm Model

The undermentioned table demonstrates the prediction accuracy of the six optimizing algorithms we have used in this project.

Optimizing Algorithms	Prediction Accuracy (%)	Cross Validation Accuracy (%)
AdaDelta	89.8033	89.8725
AdaGrad	89.7669	89.8846
Adam	89.6334	89.6752
Nadam	89.9004	89.8725
RMSprop	89.7912	89.8907
SGD	89.8590	90.0060

Table.1. Prediction Accuracy and CV Accuracy of different classification models

From the Table 1, it can be seen that the Prediction accuracy and cross validation are almost same, the differences in all the algorithms are minute, but when we look at the both accuracies at the same time SGD seems to be best fit since its Prediction accuracy is second highest i.e 89.8590% which is just 0.0414% less than Nadam which seems to be highest 89.9004 but it has the highest Cross Validation Accuracy of 90.0060 % which is 0.1153 % higher than the second best ie RMSprop . Again when Analyzing the classification report of all the model same case occurs as, all the metrics have very negligible differences so its really hard to decide which one is best suitable for predicting the result based on these only.

And the Table 2 shows the prediction accuracy of the four different classification algorithms we have used in our previous project.

Classification Model	Prediction Accuracy (%)
Decision Tree Model with rpart	88.16
Decision Tree Model with C5.0	89.03
Random Forest Model	89.22
eXtreme Gradient	89.66

Boosting (XGBoost).	

Table.2. Prediction Accuracy of different classification models

From the Table1 and Table 2, it can be seen that almost all the algorithms using Neaural Net are the top performing algorithms than the top performing algorithms which we had implemented in our previous study. Even though the differences in all the metric values are not that much, we will choose any algorithms using Neural Net in terms of predicted accuracy as the best model in our prediction because they ranks almost above in all the considered metrics .

VI. CONCLUSION

The results explained in the previous chapter concludes the study. We managed to get results from the given data set and compare algorithms, which was our primary goal for this project. While programming, interpreting the results and studying the results, whole research group got a hands-on experience with modern tools for computational data analytic with a real business case.

The next step for this project would be the optimization of each algorithm and implementing better parameters in neural network. In this project, default settings of each algorithm was used. Adjusting settings to get better results would require a more in-depth study of each algorithm to know how the performance could be improved.

REFERENCES

[1] "UCI Machine Learning Repository: Bank Marketing Data Set." [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing. [Accessed: 15-April-2019].

[2] S. Moro, P. Cortez, and P. Rita, "A data-driven approach to predict the success of bank telemarketing," *Decision Support Systems*, vol. 62, pp. 22–31, 2014.

[3] C. Vajiramedhin and A. Suebsing, "Feature selection with data balancing for prediction of bank telemarketing," *Applied Mathematical Sciences*, vol. 8, no. 114, pp. 5667–5672, 2014.

[4] H. A. Elsalamony and A. M. Elsayad, "Bank direct marketing based on neural network and C5. 0 Models," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 2, no. 6, 2013.

[5] "Home - Keras 1.2.2 Documentation." [Online]. Available: http://faroit.com/keras-docs/1.2.2/. [Accessed: 15-April-2019].

[7] M. Mishra, "Hands-On Introduction To Scikit-learn (sklearn)," *Towards Data Science*, 29-Aug-2018. [Online]. Available: https://towardsdatascience.com/hands-on-introduction-to-scikit-learn-sklearn-f3df652ff8f2. [Accessed: 15-April-2019].

[8] H. Ferreira, "Dealing with categorical features in machine learning," *Medium*, 25-Jun-2018. .

[9] Y. Liu, "Encoding Categorical Features," *Towards Data Science*, 12-Sep-2018. [Online]. Available: https://towardsdatascience.com/encoding-categorical-features-21a2651a065c. [Accessed: 15-April-2019].

[10] T. Shah, "About Train, Validation and Test Sets in Machine Learning," *Towards Data Science*, 06-Dec-2017. [Online]. Available: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7. [Accessed: 15-April-2019].

[11] "10.9. Adadelta — Dive into Deep Learning 0.7 documentation." [Online]. Available: https://d2l.ai/chapter_optimization/adadelta.html. [Accessed: 15-April-2019].

[12] "10.7. Adagrad — Dive into Deep Learning 0.7documentation." [Online]. Available: https://d2l.ai/chapter_optimization/adagrad.html. [Accessed: 15-April-2019].

[13] "10.10. Adam — Dive into Deep Learning 0.7 documentation." [Online]. Available: https://d2l.ai/chapter_optimization/adam.html. [Accessed: 15-April-2019].

[14] "10.10. Adam — Dive into Deep Learning 0.7 documentation." [Online]. Available: https://d2l.ai/chapter_optimization/adam.html. [Accessed: 15-April-2019].

[15] "An overview of gradient descent optimization algorithms," *Sebastian Ruder*, 19-Jan-2016. [Online]. Available: http://ruder.io/optimizing-gradient-descent/. [Accessed: 05-May-2019].

[16] "10.8. RMSProp — Dive into Deep Learning 0.7 documentation." [Online]. Available: https://d2l.ai/chapter_optimization/rmsprop.html. [Accessed: 15-April-2019].

[17] "Stochastic Gradient Descent - Mini-batch and more," *Adventures in Machine Learning*, 30-Mar-2017. .

[18] J. Brownlee, "Dropout Regularization in Deep Learning Models With Keras," *Machine Learning Mastery*, 19-Jun-2016. .