Machine Learning - COSC 6342 Fall 2022

Final Project

Customer Churn Prediction

GitHub - https://github.com/sainarne15/CustomerChurnPrediction-Study

Team- 20:

LAKSHMI NARASIMHA SAI NARNE (2157472)

SAHITHI LAVU (2091440)

GREESHMA DAMERA (1951304)

Contents

1. INTRODUCTION

With increasing competition in all sectors, a company's primary aim is to retain its customers without losing them. Predicting customer churn, which involves customers that chose to leave a particular company or not renew their subscriptions with the company became very prominent to plan the future of the company. The key strategy involves retaining the churners who maximize the company's profit. This Customer Churn Prediction has applications in various domains including e-commerce websites, insurance companies, telecommunications, and the banking sector etc. In this project, we are going to focus on the telecommunications industry. Customer churn is quite common in telecom, and in this context, it means that a customer switches to another service provider. The result of the model is whether the customer is returning or renewing the subscription with the provider or not. Here, we tried to incorporate and compare various machine learning models like Logistic Regression, Support Vector Machines, Random Forest, Gradient Boosting, a simple neural network, and ELM (Extreme Learning Machines) to predict customer churn. The comparison is done using the accuracies obtained for each of the models on the dataset.

2. SURVEY

Various techniques have been used in estimating retention in telecommunication industries. In most of the studies, Machine Learning was implemented [1]. Different techniques have been implemented for predicting customer churn in telecommunication industries, some of them are mentioned below.

Gavril et al. [2] reported the latest data mining technique in predicting retention for prepaid clients. The data which is used in this research had more than 3000 client call information, and 21 features with a target variable having either Yes or No values. Some of the features consist of the quantity of receiving and sending messages, as well as the voicemail of every customer. To minimize dimensionality, PCA (Principal Component Analysis) is used. To forecast churn, models such as SVM, neural networks, and Bayes were used. AUC was used to determine the model's accuracy which was 99.55%, 99.10%, and 99.70% for neural networks, Bayes networks, and SVM [1].

He et al. [3] developed a model using Neural Network. In their study, they addressed the issue of customer attrition in a huge Chinese telecommunication firm with approximately 5.23 million consumers. The accuracy obtained from the model after evaluating the data is 91.1% [1].

Idris et al. [4] presented a programming-based technique using AdaBoost to address the retention issue in telecommunication industries. They used two datasets in their study to evaluate the performance of their model and obtained accuracies of 89% and 63% respectively. The datasets used are from two telecom services namely Orange Telecommunication and cell2cell [1].

Predicting customer churn is not only restricted to the telecommunications industry but has also been carried out in *e-Commerce websites, subscription-based businesses, insurance companies, banking, mortgage companies, search advertisements* and many more.

Alshamsi A. [5] generated a model for the e-Commerce sector which correlates the primary factors which lead to turnover or churn to address the problem of retaining customers in e-commerce considering rapid growth in e-commerce transaction volume and fierce market competition to meet customer demand. The study involved developing models using Logistic Regression, Decision tree, and Random Forest with accuracies of 80.5%, 84.9%, and 93.5% respectively [5].

Kamorudeen A. Amuda and Adesesan B. Adeyemo [6] generated a model for predicting customer retention in a banking industry utilizing neural networks multi-layered perceptron with 2 overfitting - Dropout, L2 regularization. This model which was created was compared to the model in Neuro solution infinity software. They used accuracy and ROC to compare both the models and found that both are comparable, and the accuracy obtained from them are 97.53% and 97.4%, respectively, with ROC curve graphs of 0.89 and 0.85 [6].

Adeyemo and Oyeniyi [7] Customer churn has emerged as a key issue in a client-focused banking business, and banks have attempted to measure customer interaction to discover early warning indications in customer behavior. They also focused on customer churn analysis in the banking system, developing a model that employed K-Means and Repeated Incremental Pruning to Produce Error Reduction to Generate Error Reduction (JRip algorithm). The dataset was collected from a

customer relationship management database and transaction warehouse of a large Nigerian bank. The results show patterns in consumer behavior and assist banks in identifying clients who are likely to churn [6].

XIA et al. [8] generated SVM on SRM (Structural Risk Minimization) to analyze customer churn for improving the ability of prediction in ML models. Dataset was obtained from the University of California's UCI database. In comparison to Decision Trees, Neural Networks, Logistic Regressions, and Naive Bayesian, findings demonstrate that SVM approach has best accuracy, covering, lift coefficient and hit and [6].

Wang et al. [9] proposed an ensemble model for predicting client attrition in search ads. The study's goal was to identify clients who were likely to abandon the advertising platform. Based on its actions in search ads, gradient boosting decision tree was utilized to predict customers who may leave in the near future. The GBDT (Gradient Boosting Decision Tree) has two types of features: dynamic features and static features. Dynamic features took into account a long series of client activities such as impressions and clicks. While static features took into account customer settings such as creation time and customer type. The dataset obtained was from the Bing Ads platform and the resulting AUC value was 0.8410 [6].
In this project, our main focus is to implement ELM and compare it's performance against some commonly used ML models for the customer churn prediction problem.

2.1 Extreme Learning Machines

Extreme learning machines is an algorithm that was proposed to train Single Hidden Layer Feedforward Neural Network. It randomly assigns values to weights in between input and output layers [10]. Additionally, it also randomly allocates bias values that are in hidden layer unlike conventional Gradient methods [10]. These parameters are left un-changed during training. Hidden layers' non-linear

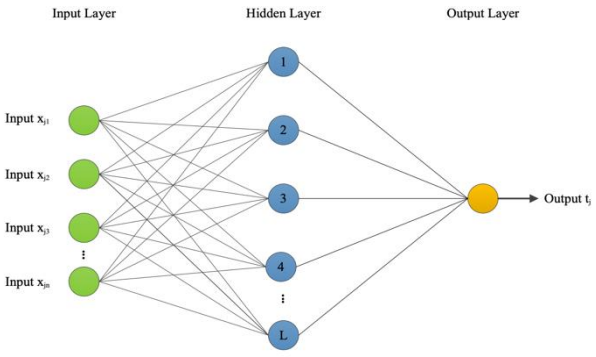

Image Source: [11]

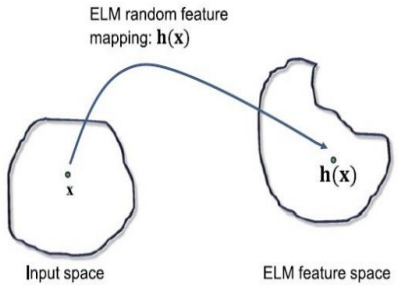

Image Source: [11], [12]

activation functions help in providing non-linearity to system [10]. Only Weights that are from hidden to output layers require learning [10]. Major dissimilarity between an ELM and a single layer neural network is that in ELM, weights from hidden to output layers are computed utilizing Moore-Penrose generalized inverse rather than back propagation. ELM is quicker over conventional models. ELM produces remarkable results as it does not require performing iterations for learning [10].

This image represents various factors for carrying out Extreme learning machines. Unlike conventional networks which require training each of its parameters, ELM has more ability to produce a best solution having random parameters [10].

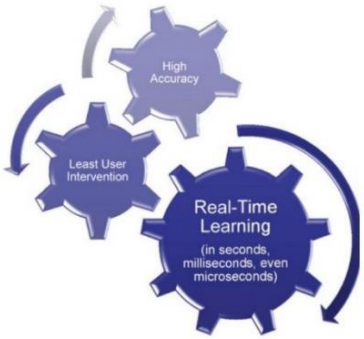

Image Source: [11], [12]

3. IMPLEMENTATION

For Customer Churn Prediction, we have built and tuned various models which include Logistic Regression, SVM, Random Forest, Gradient Boosting, and neural network. We also tried to build an Extreme Learning Machine to compare with the above models.

3.1 Dataset:

The data set for this project is taken from the Customer churn prediction 2020, Kaggle. This is split into two separate sets: Training set which contains 4250 samples and a test set with 750 samples. Each sample has 19 columns or features and 1 Boolean variable "churn" representing the sample's class. The churn which is the target variable has values "no" or "yes." Here "no" represents that the customer is not churned i.e., the customer remains with the provider, and "yes" represents that the customer is churned i.e., the customer will leave the respective provider.

3.2 Data Pre-processing:

1. Checked for NULL values and columns that have categorical variables.
2. Replaced the values of columns "international_plan," "voice_mail_plan," and "churn" from Boolean values ['no,' 'yes'] to [0, 1] respectively.
3. The column "area_code" has only three unique values. Performed one hot encoding and created a dummy variable for each unique value i.e., 3 dummy variables in total, and then assigned the values 1 or 0 for each row in the data frame to represent the presence or absence of value.
4. The column "state" has 51 unique values. Performing one hot encoding is not a clever idea as it creates 51 dummy variables. We applied hashing encoding and transformed the data to new dimensions.
5. The data set is split into two sets: the train set and the test set in the ratio 8:2, respectively.
6. Since numerical variables have different value ranges, performing normalization is important. A standard scalar function is applied only to the train data where it standardizes features by removing the mean and scaling the features to the unit variance.
7. We perform similar normalization on test data using train data mean and variance i.e., applying transformations by using the same parameters learned from the train data. The reason behind this is that the models are not supposed to learn anything about the test data.

3.3 Imbalance Classification Problem:

For this project, we also tried to explore the Imbalance classification problem that occurs due to the imbalance in the dataset, as there are more examples of the customers not churning in comparison with the other class. We have used random oversampling over the minority class to balance the data set. We implemented all the models with and without data balancing to see the changes in the performance of the various models used and noted them in tables 3.10 and 3.11.

3.4 Logistic Regression:

1. We imported Logistic Regression from the sklearn library and used 'liblinear' as our solver which supports binary linear classification. This solver only offers a dual formulation for the L2 penalty, but it supports both L1 and L2 regularization.

2. We then train the model using the train data and various metrics like confusion matrix, accuracy, and classification report are calculated based on the predictions made by the model on the test data.
3. The test accuracies obtained for the Logistic Regression model, before and after consideration of the classification imbalance problem are **85.76%** and **86.2%** respectively**.**

3.5 Support Vector Machines:

1. For support vector machines, we implemented it from the sklearn library's Support Vector Classifier from the **svm** module.
2. We then fitted the model on the train data and evaluated the performance of the model on the test data set using various metrics.
3. The test accuracies achieved for Support Vector Machine, before and after consideration of the classification imbalance problem are **91.05%** and **90.04%** respectively.

3.6 Random Forest:

1. For this model, we imported the random forest classifier from ensemble module that is present in the sklearn package.
2. We fine-tuned the parameters **"n_estimators," "max_depth,"** by performing **Grid Search** which is a process for selecting the values for model's parameters that maximizes the performance of the random forest.
3. **GridSearchCv()** evaluates all combinations of parameters we defined and cv = 5 is the number that determines the cross-validations we must try for each selected set of parameters. We then fit the model using train data and the best values we got from grid search for n_estimators and max_depth parameters are **500** and **15,** respectively.
4. Now, we fit the random forest classifier model built with the above calculated values for hyperparameters, on the training set.
5. The test accuracies obtained for Random Forest Classifier, before and after consideration of the classification imbalance problem are **95.17%.** and **91.5%,** respectively.

3.7 Gradient Boosting:

1. Gradient Boosting classifier from ensemble module that is present in sklearn package is used to build the model.
2. Like the Random Forest, we fine-tuned the parameters to achieve better accuracies.
3. The best values for the hyper-parameters **"n_estimators," "max_depth," "learning_rate"** obtained after performing grid search are **850, 5,** and **0.1,** respectively.
4. These values were used to implement the gradient boosting classifier and then training is performed on the training data set.
5. The test accuracies achieved for Gradient Boosting Classifier, before and after consideration of the classification imbalance problem are **95.52%** and **94.58%** respectively.

3.8 Neural Network:

1. A sequential model with three layers is built using keras library, an input layer with 16 nodes and ReLU (Rectified Linear Activation Unit) as an activation function, a hidden layer with 64 nodes and ReLU as an activation function, and finally an output layer with a single node that produces output using sigmoid as an activation function.
2. We then compiled the model using "Adam" as an optimization function and loss as "binary_crossentropy" as we are performing binary classification.
3. Regularization is important to prevent the overfitting of data. Here, we used the Early Stopping strategy which stops training when the model's performance is not improved. We used patience as '10' i.e., the model waits for 10 epochs to stop the training of the model if there is no improvement and we set restore_best_weights to "True" which indicates to use of the best model weight and not the last epoch weight.
4. We then train the model by specifying early stopping as our callbacks, epochs =80 and batch size =10. Here we split the part of the training set (20%) for validation.
5. The model stops training at the 26th epoch due to the early stopping and the test accuracies achieved for this model, before and after consideration of the classification imbalance problem are **90.7%** and **90.8%** respectively.

3.9 Extreme Learning Machines:

1. An ELM model is built for classification from the class developed by Li Xudong [13]. We changed and updated the class such that it can be used for this study, and we changed it into our own sklearn estimator. It is a single layer network with weights and bias are randomly initialized.
2. We built the network with 32 hidden units as default, and ReLU as the activation function as it is giving the best results compared to other functions such as tanh and leaky ReLU, etc.
3. There are three solutions used in this model in the place of backpropagation. We used no regularization and used the Moore-Penrose generalized inverse to calculate the beta matrix (The weights between the hidden and output layer) which later is used to compute the outputs.
4. We transformed the elm class to sklearn estimator for classification such that we can use the GridSearchCV to estimate the best number of hidden units for the model. We found that the model with 256 nodes performs better when we gave the grid with 2^2 to 2^{10} hidden units to search for the best number.
5. After training, the model gave an accuracy of **91%** on train data and a test accuracy obtained for ELM, before and after considering classification imbalance problem are **90%** and **87.88%** respectively.

3.10 Comparison Table for various models used in this Project without considering classification Imbalance Problem:

Model	Test Accuracy	Precision	Recall	F1- Score
Logistic Regression	85.7%	0 - 0.88 1 - 0.55	0 - 0.97 1 - 0.22	0 - 0.92 1 - 0.32
SVM	91.05%	0 - 0.91 1 - 0.89	0 - 0.99 1 - 0.45	0 - 0.95 1 - 0.60
Random Forest	95.17%	0 - 0.96 1 - 0.92	0 - 0.99 1 - 0.74	0 - 0.97 1 - 0.82
Gradient Boosting	95.52%	0 - 0.96 1 - 0.90	0 - 0.98 1 - 0.79	0 - 0.97 1 - 0.84
Neural Network	90.7%	0 - 0.93 1 - 0.72	0 - 0.96 1 - 0.60	0 - 0.95 1 - 0.66
ELM	89.5%	0 - 0.91 1 - 0.74	0 - 0.97 1 - 0.45	0 - 0.94 1 - 0.56

3.11 Comparison Table for various models used in this Project after considering classification Imbalance Problem:

Model	Test Accuracy	Precision	Recall	F1- Score
Logistic Regression	86.2%	0 - 0.89 1 - 0.57	0 - 0.96 1 - 0.29	0 - 0.92 1 - 0.39
SVM	90.04%	0 - 0.91 1 - 0.87	0 - 0.99 1 - 0.42	0 - 0.95 1 - 0.57
Random Forest	91.5%	0 - 0.93 1 - 0.79	0 - 0.97 1 - 0.58	0 - 0.95 1 - 0.67
Gradient Boosting	94.58%	0 - 0.96 1 - 0.83	0 - 0.97 1 - 0.79	0 - 0.97 1 - 0.81
Neural Network	90.8%	0 - 0.94 1 - 0.71	0 - 0.95 1 - 0.65	0 - 0.95 1 - 0.68
ELM	87.88%	0 - 0.90 1 - 0.64	0 - 0.96 1 - 0.41	0 - 0.93 1 - 0.50

4. CONCLUSION

Customer Churn Prediction is important for any company as the customers have diverse options to choose from and the competition is extremely high. In this work, we aimed to compare the performance of different models including Logistic Regression, SVM (Support Vector Machine), Random Forest, Gradient Boosting, Neural Networks, and another type of neural networks with a different approach namely Extreme Learning Machines after tuning them. In this study, we performed data cleaning, feature transformation, and normalization on the data set. We have also observed that data is imbalanced, we used random oversampling on the minority class, but interestingly, the performance did not improve much but instead it diminished on test data. We used GradientSearchCV to tune the models with 5-fold cross-validation. We also built the ELM model and tuned it using the same method after making the ELM class into sklearn estimator. We concluded our work with Gradient boosting performing better than all the models followed by random forest. Since the data is obtained from a past Kaggle Competition, we used the gradient boosting model which was trained without and with considering class imbalance on the hidden test data from the competition and found that the accuracy score achieved is 94.66% and 95.11% which might get the rank 21 in the leader board.

5. FUTURE WORK

In the future, we plan to use different models such as XGBoost which might perform better than the existing models for classification and Deep-ELM model for comparison. We also intend to tune more parameters for the models used in this project, as we have only used a limited number of hyperparameters here, and to add more GPU as the time needed for parameter tuning was excessive (more than 5 hours). We also plan to explore more into the data preprocessing techniques and other approaches to solve the classification imbalance problem to improve the performance.

6. REFERENCES

[1] A. K. Ahmad, A. Jafar, and K. Aljoumaa, “Customer churn prediction in telecom using machine learning in big data platform,” *J Big Data*, vol. 6, no. 1, p. 28, Dec. 2019, doi: 10.1186/s40537-019-0191-6.

[2] I. Brandusoiu, G. Toderean, and H. Beleiu, “Methods for churn prediction in the pre-paid mobile telecommunications industry,” in *2016 International Conference on Communications (COMM)*, Jun. 2016, pp. 97–100. doi: 10.1109/ICComm.2016.7528311.

[3] Y. He, Z. He, and D. Zhang, “A Study on Prediction of Customer Churn in Fixed Communication Network Based on Data Mining,” in *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, 2009, pp. 92–94. doi: 10.1109/FSKD.2009.767.

[4] A. Idris, A. Khan, and Y. S. Lee, “Genetic Programming and Adaboosting based churn prediction for Telecom.”

[5] A. Alshamsi, “Customer Churn prediction in ECommerce Sector.” [Online]. Available: https://scholarworks.rit.edu/theses

[6] K. A. Amuda and A. B. Adeyemo, “Customers Churn Prediction in Financial Institution Using Artificial Neural Network.”

[7] A. O. Oyeniyi and A. B. Adeyemo, “Customer Churn Analysis In Banking Sector Using Data Mining Techniques,” 2015. [Online]. Available: www.ajocict.net

[8] G. XIA and W. JIN, “Model of Customer Churn Prediction on Support Vector Machine,” *Systems Engineering - Theory & Practice*, vol. 28, no. 1, pp. 71–77, Jan. 2008, doi: 10.1016/S1874-8651(09)60003-X.

[9] Q.-F. Wang, M. Xu, and A. Hussain, “Large-scale Ensemble Model for Customer Churn Prediction in Search Ads,” *Cognit Comput*, vol. 11, no. 2, pp. 262–270, Apr. 2019, doi: 10.1007/s12559-018-9608-3.

[10] J. Wang, S. Lu, S. H. Wang, and Y. D. Zhang, “A review on extreme learning machine,” *Multimed Tools Appl*, Dec. 2021, doi: 10.1007/s11042-021-11007-7.

[11] M. Abbas, A. Albadr, and S. Tiun, “Extreme Learning Machine: A Review,” 2017. [Online]. Available: http://www.ripublication.com

[12] G. Huang, G.-B. Huang, S. Song, and K. You, “Trends in extreme learning machines: A review,” *Neural Networks*, vol. 61, pp. 32–48, Jan. 2015, doi: 10.1016/j.neunet.2014.10.001.

[13] Li Xudong, from NSSC.CAS Beijing, https://github.com/5663015/elm/