CSCI-6409 PROCESS OF DATA SCIENCE

Final Project Report

GROUP - 4

GROUP MEMBERS:

Name	Banner ID
Jaimi Sheta	B00886563
Jay Bhagvanbhai Sonani	B00891984
Mitul Pravinbhai Malani	B00869519
Prit Ajaykumar Sorathiya	B00890175
Prit Thakkar	B00890731

Course Instructor: Dr. Martha Dais Ferreira

Gitlab repository link: https://git.cs.dal.ca/psorathiya/csci-6409-s22-g4

1. Introduction

In this technological age, we are moving toward online communication, which has increased people's screen time all over the world, and we want to study, analyze, and predict the amount spent by a specific user based on factors such as age, number of children, and smartphone device, to name a few.

Based on user activity data, the time spend online each day is growing since 2012 [1]. An estimation done by WHO suggests that on average, an individual will spend more than 3.4 million minutes over their span of life, which converts to 6 years and 8 months based on the projections [1]. For instance, Facebook tops the list with a daily time spent of 33 minutes [1].

As the time spent is going to increases, more and more businesses are trying to go online and are working on making interactive and intuitive experiences to attract users with the end goal of making users spend more time on their respective applications. We plan to analyze users' online activity by handling the time series data and build a modeling strategy to capture patterns to make a predictive model which can be used by any organization to predict the time a user will spend on their application based on the feature set.

2. Data understanding and preprocessing

Let us start with general outline information of the "Users Active Time Prediction" [2] dataset. The provided dataset consists of three files. First, a file named customer.csv contains features such as id (the unique identifier for a customer), gender, number_of_kids, smartphone_device, internet_provider, and application_name. All the features are self-explanatory, however, application_name indicates the name of the application a customer was using during the collection of data. Second, another file named pings.csv consists of features such as the id (customer's id) and timestamp (the UNIX timestamp when the event is logged). Apart from that, a test data file named test.csv includes id, date (the date for which we have to predict the number of hours), and onlinehours (the total number of hours spent on social media apps).

As we have to predict a user's online hours spend on social media apps, the problem is a supervised regression task. We have to calculate the online hours of users from the ping data to train the model so that onlinehours can be a target feature because we do not have users' online hours within our dataset explicitly. Also, the ping data contains the Unix timestamp (the number of seconds passed after 1970 to now), we have to convert it to the date type from the timestamp to train and test the model.

The dataset contains a duplicate index column called 'Unnamed: 0', we have to remove that before processing. We have customer id in the customers dataframe and in the pings dataframe, which is irrelevant but, we cannot remove it until we perform a groupby based on customer_id for model training. Gender is categorical ('MALE', 'FEMALE'), we need to do one hot encoding.

There are some missing values in the internet_provider field and we will replace such columns with the mode value of the feature. In the pings dataframe, we have a timestamp that cannot be used for model training as we need to convert it to more interpretable fields for model training.

The UNIX timestamp is sorted into ascending order, grouped by customer id and we have computed the difference between timestamps and later performed a sum on the same to get the number of hours as it is in the test data.

3. Literature review

As a part of solving our problem which looks like a time series at first, we incorporated many learnings from academia. Starting with choosing a baseline model as linear regression, when we use time series data, the representation of weights are features at that instant, multiplied by the regression coefficients and the standard error term. The Ordinary Least Squares (OLS) is an unbiased estimator, and we had to check if such an assumption holds in the time series data or not [3]. In time series, the autocorrelation of error terms is a problem, which can be detected by using the Durbin-Watson statistic [3]. Linear regression, unless combined with other models was not capable of capturing all the trends. We did convert it to a nontime series one, but it did not yield many results.

The ensemble is a machine learning technique that combines multiple base models to generate one optimal model. XGBoost algorithm focuses on penalizing the wrong outcomes with importance on determining the split on the dataset. To check what parameters to use, we referenced an academic paper where XGBoost was compared with Random Forest and Gradient Boosting with 10-fold cross-validation, and as a result, it was found that the default case of XGBoost performs worst as compared to a tune one [4]. However, later it was found that such is not the case every time, and in datasets where there is noisy data, default parameters have performed well. On the memory and time requirements, XGBoost depicts the fastest performance, about 3.5 times faster than Random Forest and 2.4-4.3 times gradient boosting [4]. When we used the learnings from this academia, we found out that our model works best on some sort of a hyper-parameter-tuned XGBoost model, which does not overfit and outperforms Random Forest and gradient boosting.

However, we wanted to do some more research on types of gradient boosting, and on some search, we came up with Light Gradient Boosting Machine (LGBM) [5] and as per the research article, LGBM outperforms XGBoost, let us know how. A different approach is used here, using scaling test data, random grid search, using a different configuration of hyperparameters, normalization, and standardization with LGBM set to a range of max depths to perform testing on how depth helps to better learn the data [5]. There are two techniques for LGBM, Gradient-Based One-Side Sampling and Exclusive Feature Bundling [6]. As a result, we found out that LGBM outperforms XGBoost in terms of accuracy as well as memory and storage [6]. We have incorporated all the findings in our approach, and we also found similar results by using LGBM as compared to XGBoost.

4. Description of the solution

In terms of a solution to the problem, we have used boosting method from the ensemble to fit the model with the test dataset. Specifically, we have used XGBoost (eXtreme Gradient Boosting) [7] and LGBM regression (Light Gradient Boosted Machine) [8] as a base model and final candidate model respectively.

First, let us talk more about an ensemble learning and boosting method of the same. The ensemble is a learning methodology that uses a collection (ensemble) of hypotheses instead of one (a typical machine learning model usually has a single hypothesis) or different training data within a single model and combines their predictions to provide an outcome of the model as a whole [9]. The main idea of using an ensemble is that it provides a better solution and reduces the error rate significantly compared to other single hypothesis models [9]. Also, we are using boosting algorithms i.e. XGBoost and LGBM regression as our models, so let us understand the boosting method as well. Boosting is a method in which models are trained sequentially and the successor model is trained by providing more focus on the instances that were incorrectly classified and less weightage on those with correct predictions [9]. At last, a combination of majority voting by all the models is done for the final outcome [9]. This is how boosting method works in ensemble learning.

Let us talk about the process we followed from the beginning to the end to train the model. Firstly, we have processed the data in such a way that it can be fitted with the model such as by applying one-hot encoding on categorical features. After that, we used scikit-learn's SelectKBest method as a feature selection and used f_regression as a score function. It is important that we choose the best features from the feature set that are more relevant and provides more correlation to the target feature so that we can have a better model in terms of predictions. We have selected 7 best features out of 12 features. We have done pre-processing on the data, removed outliers, and made sure that enough training data is provided to the model so that it cannot overfit the training dataset. Then, we split the dataset into training (40%), test (30%), and validation (30%) splits.

Then, we trained the XGBoost as a baseline model and LGBM Regression as a final candidate model. As we have discussed in the literature review, linear regression was not performing well on this kind of problem, which is why we have decided to go with the ensemble boosting algorithms. Finally, we fit the model with the training data. The candidate model was doing well compared to the base model, but the candidate model was not doing well enough on the validation and test dataset. Therefore, we also tried using L1 and L2 regularization one by one to stop the overfitting problem of the model, but we found almost no improvement even after using regularization. Hence, we removed the regularization term from the model.

5. Data analysis, Results, and Evaluation

Let us start with a discussion of analyses we have done on the dataset and patterns we have found within the dataset. Interestingly and as expected, the customers from the age group 20 to 30 spend more time on screen time compared to other age groups within the dataset. The male customers are threefold more than female customers. Customers having Apple and Samsung smartphones are at the top of the smartphone usage list and similarly Bell is the most used internet service provider within the dataset. When we plotted the bar plot of the used application names, Tiktok is the most used app, while other social media apps also have significant contributions to screen time usage. These were the most significant patterns that we found by analyzing the dataset. As we discussed earlier, we have opted for the best 7 features out of 12 features to fit the model. Also, we did manual hyperparameter tuning for the improvement of the model. We mainly used n_estimators, objective, verbosity, and random_state in the base model and n_estimators, boosting_type, and min_data in the candidate model for the hyperparameter tuning.

To evaluate the model, we have considered the R2-score as a main evaluation metric for the regression model. The reason behind choosing the R2-score as an evaluation metric is that R-squared is not a measure of how accurate the predictions are, but a measure of fit unlike MAE (Mean Absolute Error) and MSE (Mean Squared Error) [10]. Also, as referenced in an academic paper [11] that concludes that R2-score is more informative and does not have the interpretability limitations existing in other metrics. We have calculated the R2-score for both the base and candidate models for training, validation and test dataset. For the base model, the R2-score of the training dataset, validation dataset, and test dataset is 0.48, 0.40, and 0.41 respectively. These are slightly less than random guesses i.e., 0.5, which is not good enough for the model. Also, these results are for the base model. Now, in the case of the final candidate model, the R2-score of the training, validation, and test dataset is 0.72, 0.5, and 0.5 respectively. This means that the candidate model is doing well on the training dataset, but not well enough on the test and validation dataset. We did use regularization but did not improve the performance.

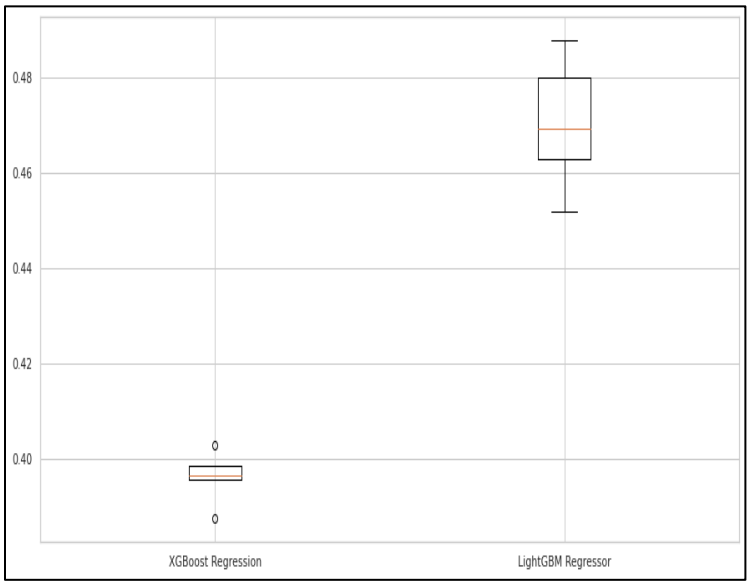

Figure 1: Evaluation comparison of the base and candidate model

Apart from that, we did data preprocessing and that is suitable for both dataset and model. We plotted the histogram of the target feature to check whether imbalanced or not, the target is not imbalanced. Also, we have plotted the box plot for the evaluation comparison of the base model XGBoost and candidate LGBM regression, which is shown in figure 1. It is seen that the LGBM regression model is more accurate and performant than XGBoost. Also, we have done a statistical significance test [12] on both models' prediction values. The p-value is 0.95 of the t-test results, which means we cannot reject the null hypothesis and there is no statistically significant relationship between the results of both models.

6. Conclusions

The introduction of new social media applications has a significant impact on the screen-time of smartphone customers all around the globe, which is a concern. We have reviewed, analyzed, and modeled the available customer and ping data to build a prediction system able to compute the amount of time a customer would spend on a given day by using XGBoost as a baseline model by performing hyperparameter tuning which yields an R2 score of 0.48 and Light Gradient Boosted Machine with hyperparameter tuning which leads to a final R2 Score of 0.72. In a nutshell, LGBM performs better after performing all the necessary modifications on the raw data.

References

[1] "Average time spent daily on social media (latest 2022 data)," BroadbandSearch.net. [Online]. Available: https://www.broadbandsearch.net/blog/average-daily-time-on-social-media. [Accessed: 5-Jun-2022].

[2] "Users Active Time Prediction", Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/datasets/bhuvanchennoju/mobile-usage-time-prediction. [Accessed: 5- Jun- 2022].

[3] W. W. S. Wei, "Time Series Regression," in International Encyclopedia of Statistical Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1607–1609.

[4] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," Artif. Intell. Rev., vol. 54, no. 3, pp. 1937–1967, 2021.

[5] H. Los et al., "Evaluation of xgboost and lgbm performance in tree species classification with sentinel-2 data," in 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, 2021, pp. 5803–5806.

[6] G. Ke et al., "LightGBM: a highly efficient gradient boosting decision tree," in Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 3149–3157.

[7] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[8] "Lightgbm.LGBMRegressor — LightGBM 3.3.2.99 documentation," Readthedocs.io. [Online]. Available: https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html. [Accessed: 02-Aug-2022].

[9] "Ensemble + Random Forests," Brightspace.com. [Online]. Available: https://dal.brightspace.com/d2l/le/content/224235/viewContent/3041820/View. [Accessed: 02-Aug-2022].

[10] B. O. Tayo, "What Really is R2-Score in Linear Regression?," Medium, 05-Nov-2021. [Online]. Available: https://benjaminobi.medium.com/what-really-is-r2-score-in-linear-regression-20cafdf5b87c. [Accessed: 02-Aug-2022].

[11] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation," PeerJ Comput. Sci., vol. 7, no. e623, p. e623, 2021.

[12] J. Dul, E. van der Laan, and R. Kuik, "A statistical significance test for necessary condition analysis," Organ. Res. Methods, vol. 23, no. 2, pp. 385–395, 2020.