Bike Sharing Demand Forecast

Tobias Hanl th2999 **Shuyu He** sh4330 **Jingyi Feng** jf3495 **Mendel Branover** mb4869 **Zixiang Yin** zy2444

Abstract

Bike-sharing systems are gaining popularity in major cities. The process of renting a bike through an app on the users' smartphone yields lots of information for the provider. Particularly important is the number of bikes rented within a given time period, so that the provider can make proper adjustments to meet the demand. In this project, we analyzed hourly data from Capital Bikeshare in Washington, D.C. over a span of two years and took two distinct approaches to predict future demand. The best regression model HGBT achieved an R^2 of 0.8928 on the validation and an RMSLE of 0.4652 on the hidden testing set. The best time series (LSTM-based) model achieved an R^2 of 0.9505 and an RMSLE of 0.4540 on the validation and test respectively, which ranks top 27% among all submissions on Kaggle.

1 Introduction

Nowadays, we see more and more bike-sharing systems available to people in big cities. They make bike renting easier and more accessible through apps on our smartphones. People are able to rent a bike from one location and return it to a different location without much effort. In the project, we used historical bike-sharing usage data along with weather and temperature data to forecast the bike rental demand of the Capital Bikeshare program in Washington, D.C.

The dataset provided by Fanaee-T and Gama [2013] includes Capital Bikeshare's hourly bike rental data spanning two years with the corresponding weather and seasonal information. The dataset contains two files, train.csv, and test.csv, with 10,887 and 6,493 rows respectively. Both files comprise four categorical variables and eight quantitative variables.

2 Exploratory Data Analysis

After parsing *datetime*, we drew a heatmap to drop highly correlated features. Then, we plotted the distribution of our target *total* and decided to treat log(*total*) as our new target as it's skewed.

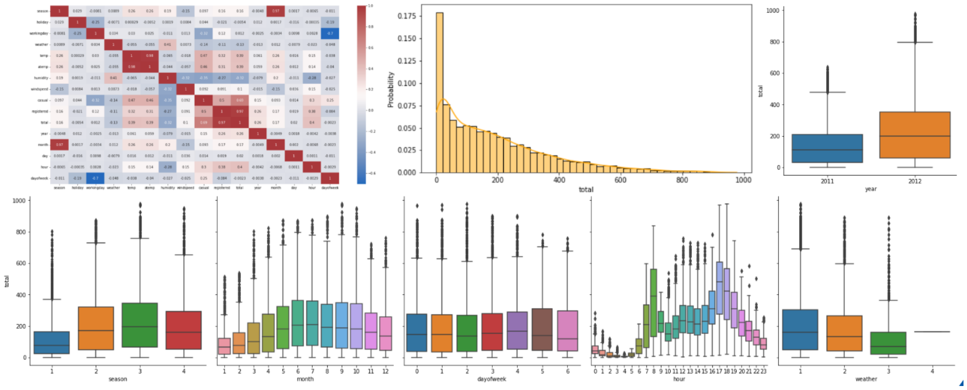

Figure 1: Exploratory Data Analysis

We also plotted several boxplots as shown above against the target by different features and found that the key observation are:

- The peak of the rental time is 8 am and 5 pm to 6 pm during weekdays.
- More rentals are in fall but fewer in spring.
- Rentals generally decrease as wind speed and humidity increase.

3 Methods

In general, we approached this project from two perspectives. One is **Regression** and the other is **Time Series**. For regression, we ignored the dependencies among data entries and assumed that all entries were drawn i.i.d. from an unknown distribution F_θ. For time series, we respected the data dependency in continuous time steps by converting sequential data into samples, each with a size of time steps $\times$ # of features.

3.1 Feature Engineering

According to the exploratory analysis in section 2, we dropped features *datetime*, *day*, *temp*, *season*, and *workingday* to avoid multilinearity. Regarding categorical features in the dataset, we applied different encoding techniques to do feature engineering as well as kept the feature space reasonable:

- One-hot encoding - *weather*
- Ordinal encoding - *holiday*, *year*
- Target encoding - *dayofweek*, *month*, *hour*
- Standard scaling - *atemp*, *humidity*, *windspeed*

3.2 Regression

After feature engineering, we first approached the project from the perspective of regression and focused on six types of models, i.e. Linear Elastic Net, K-Nearest Neighbour (KNN), Linear Support Vector Machine (SVM), Decision Tree, Random Forest, and Histogram Gradient Boosting Tree (HGBT).

3.3 Time Series

From the perspective of time series, we only built a Long Short-Term Memory (LSTM) model which is a Recurrent Neural Network. We trained the plain LSTM model with a single dense output layer, but the best R^2 we got was less than 0.85. For this reason, we further processed the features by doing One-hot encoding on features *dayofweek* and *month* and built the following architecture. In fact, we also re-trained all regressors with a One-hot encoding on features *dayofweek* and *month*, but no improvement was made. Therefore, we decided to keep the original regressors for model evaluation in the next section.

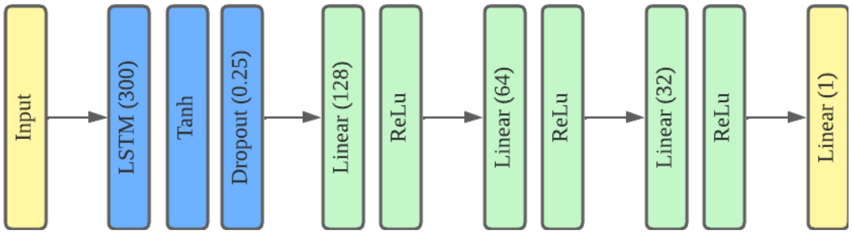

Figure 2: Time series model architecture

4 Results

For regression models, we tuned two hyperparameters for each type of model during cross-validation and used mean R^2 as the metric for model selection. We compared the performance of regressors with and without feature engineering by plotting the heatmaps below. Based on our result, feature engineering greatly improved the model performance. Among all regressors, the highest R^2 of 0.8929 was achieved by the HGBT model with a *max_bin* of 117 and a *learning_rate* of $10^{-0.86}$.

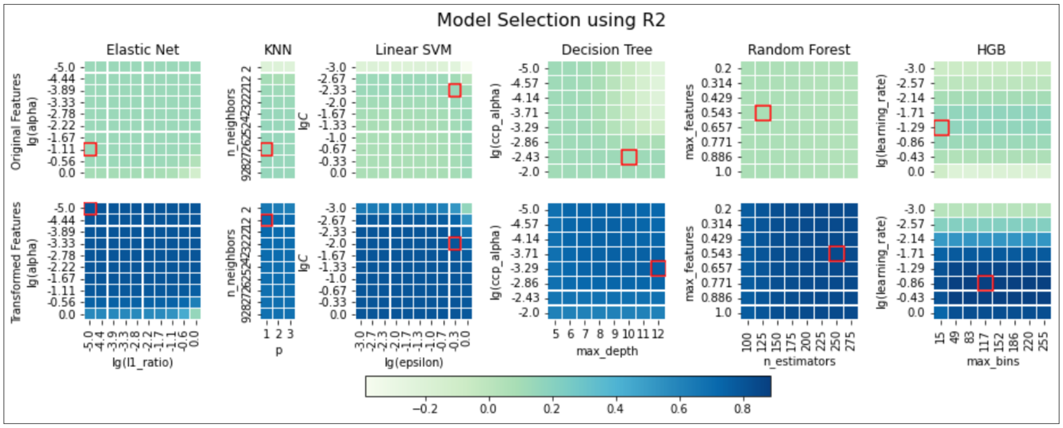

Figure 3: Heatmaps for model selection of regressors

For the LSTM-based model, we only tuned the number of steps looked back and did the model selection by MSE loss on the hold-out validation set due to limited computation resources. The optimal model achieved a R^2 of 0.9505 with a *look_back* of 12.

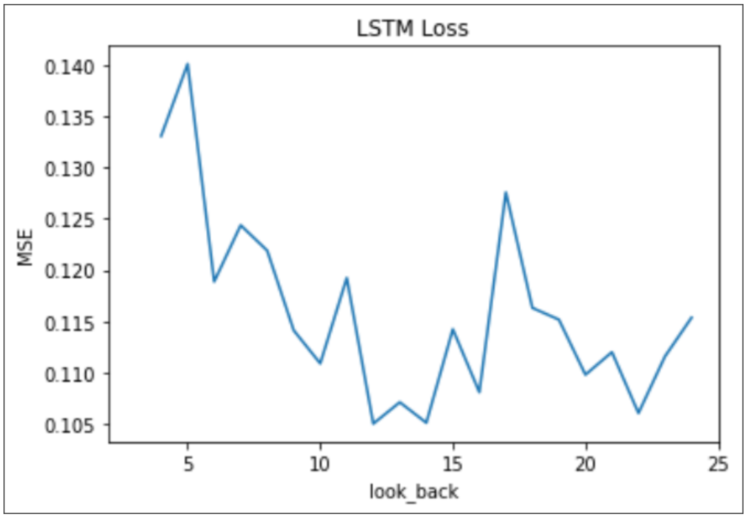

Figure 4: LSTM loss versus *look_back* period

Comparing the best-performing models on the hidden Kaggle test dataset, the LSTM-based model performed best with an RMSLE of 0.4540 (top 27%) compared to 0.4652 (top 31%) for Regression.

5 Conclusion

In terms of regression, feature engineering on *datetime* helped us greatly improve the model performance. For the LSTM-based model, the performance was further improved by One-hot encoding *dayofweek* and *month*. However, Regressors' performance declined when One-hot encoding was used. In conclusion, our best model gets an RMSLE of 0.4540 on the hidden testing set which ranks top 27% among all submissions on Kaggle.

References

Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, pages 1–15, 2013. ISSN 2192-6352. doi: 10.1007/s13748-013-0040-3. URL [WebLink].