Universität St.Gallen

Universität St.Gallen – Hochschule für Wirtschafts-,
Rechts- und Sozialwissenschaften, Internationale Beziehungen und Informatik (HSG)

Predicting Wait Times at "Haunted Mansion" in Disney World's Magic Kingdom

A Regression and Time-Series Problem

Sara Knapp (21-621-420)
Lukas Gaiser (21-618-236)
Annina Mack (20-619-060)

Data Science Fundamentals Group Project
Submitted to Dr. Johannes Binswanger
05. December 2022

Figure 1: Haunted Mansion in Walt Disney World, Florida
Source: https://www.disneyworld.eu/attractions/magic-kingdom/haunted-mansion/

Table of Contents

Table of Figures

1 Introduction and Use Case

Americans, collectively, spend about 37 billion hours per year just waiting in line (Lam, 2015). While this in itself can cause anxiety and annoyance, our negative emotions are even more elevated when the wait times don't match our expectations. Businesses have long recognized the importance of queue management for the perceived quality of the service and purchasing experience. In that respect, estimating wait times accurately is especially important.

In this project, we will take up this exact issue using data from Disney World's amusement park in Florida. According to the source (Touringplans, 2022) the data was harvested from Disney's experience app, by staff actually tracking wait times in the parks, and by day-to-day visitors entering the information into the platform's own app. While there is data available for dozens of attractions, we are going to focus on the "Haunted Mansion", which is located in the Magic Kingdom and one of the most popular attractions of Disney World Florida. The goal is to predict waiting times using a set of features, including e.g. weather, and opening hours of other attractions, while also taking into account the time-series component. The result of the best predictive model could then be made available to customers in an app to help them plan their visit to the Haunted Mansion – not weeks in advance but rather dynamically in the park - or serve Disney as a dynamic crowd management tool.

2 Data Preprocessing

For our analysis, we used two datasets: (1) one containing timestamps and attraction-specific wait times ranging from 2015 to 2021 as well as a (2) second metadata set with daily data about weather, holidays, opening hours, and more. A first exploration of the data in Figure 2 gave us important insights about the seasonality of the wait times over the course of the day, the September low and the December peak. It also illustrates the effects of the pandemic. In order to eliminate any disadvantageous distortions related to this and maintain a continuous dataset, we decided to only keep data prior to March 2020.

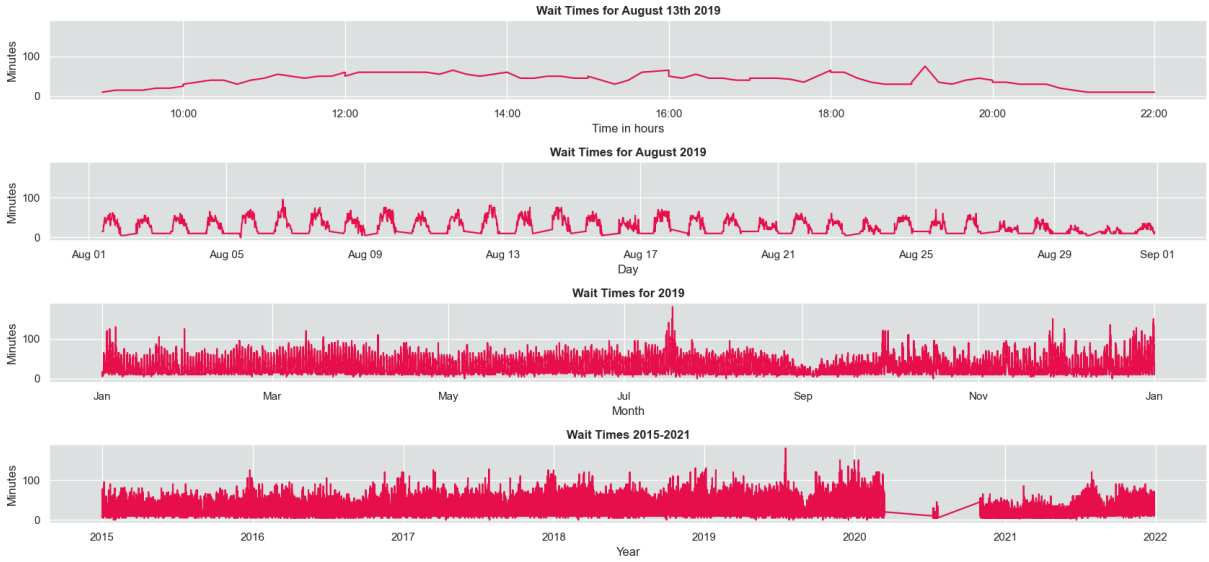

Figure 2: Line Plot of wait times for a day, a month, a year, and the whole period
Source: Own Figure

A problem we encountered when looking at dataset (1) is that for each time stamp either the “actual” wait time was reported or the “posted” one, which would be what you see displayed on a sign when first getting in line. Because only 3.41% of all observations were actual wait times, we decided to use the posted wait times as the target variable (y). This, however, needs to be kept in mind when evaluating the predictive power of our model.
After that, we started with the cleaning process of dataset (1). In order to create a more evenly spaced dataset, we decided to aggregate the data to the nearest 10 minutes of the hour and take the mean if there was more than one entry per 10 minutes.
Then we continued with dataset (2). We started by removing all columns that contained only NaN-values or were related to the other theme park in California. Categorical features, except for the time features, were converted to numerical data by one-hot-encoding and string percentage values were also dealt with accordingly. Lastly, we merged the two datasets on the ‘date’ column and imputed any remaining missing values with the value before or the median.

3 Machine Learning Models

First, simple models without any advanced feature engineering were applied, specifically a linear regression, Gradient Boosted Trees, and a Random Forest. Then, we modified the data to account for the time-series aspect and lastly added a lag feature. Due to the abundance of features available, we also tested some feature selection methods based on simple correlation as well as the Spearman’s rank. These methods, however, showed only very low absolute correlations between the variables and the target (below 0.3), and did not yield satisfactory results in combination with the models. With greater computing power, a Wrapper method might have improved the predictive power of the models even more.

3.1 Simple Models as a Baseline

Before applying any model, we used the command TimeSeriesSplit() to create training and test sets for an expanding window cross-validation. It was an important realization not to use the command train_test_split() instead, since we are working with time-sensitive data that cannot be randomly shuffled and past events that cannot be predicted by data from the future. Splitting the data three times with a constant testing set size resulted in the following graph (Figure 3).

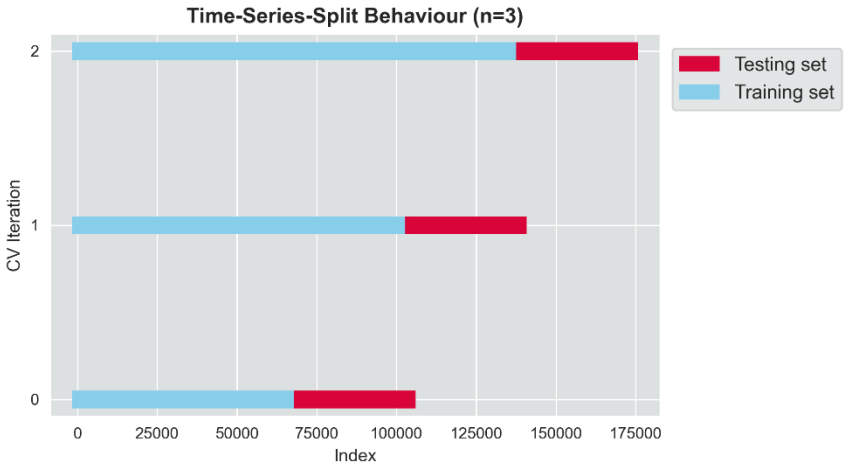

Figure 3: Time-Series-Split behavior (n=3)
Source: Own Figure

The Random Forest (n_estimators=15, max_depth=70) yielded a cross-validated, average RMSE of 14.1. Due to computing and time constraints, hyperparameter tuning was performed manually. We plotted the resulting predictions in comparison to the posted wait times for four random days in Figure 4 below. The trend over the course of the days is predicted reasonably well, however, the wait times are mostly underestimated.

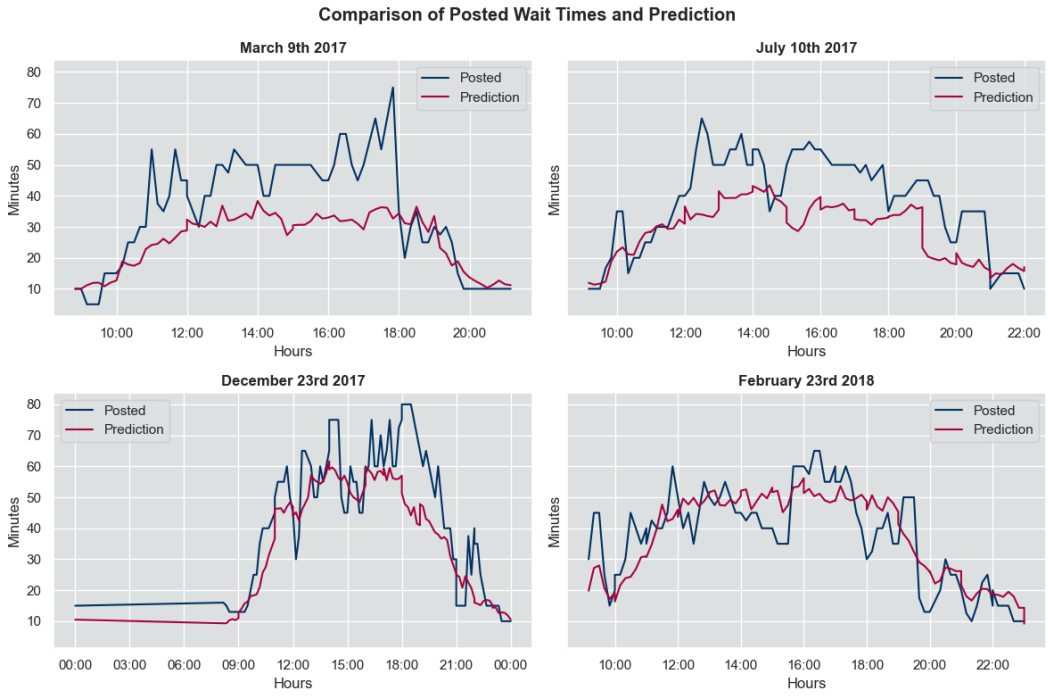

Figure 4: Comparison of Posted Wait Times and Prediction with Random Forest
Source: Own Figure

We then moved on to Gradient Boosting Trees, which do not build each tree independently but combine results along the way and can therefore result in better performance (Glen, 2019). With a mean RMSE of 12.869 it predicted the posted wait times slightly better. A plot of the same type as the one above showed a similar performance on our data.
Additionally, we fitted a simple linear regression model which, as expected, performed considerably worse with an RMSE of 68.957 and plotted only straight lines.

3.2 Periodic Time Feature Engineering

In an attempt to improve the previous models and take the cyclicality of the time series into account, we performed some engineering on the periodic time features, like MONTHOFYEAR or HOUROFDAY. The first approach was a sine and cosine transformation, through which the time features are encoded into two separate features that simultaneously represent the time point without any information loss (Lewinson, 2022). A major advantage of this transformation is that it preserves the continuity of the time variables, meaning there are no abrupt jumps between the first (e.g. HOUROFDAY: 0) and the last value (e.g. HOUROFDAY: 23) of the range.

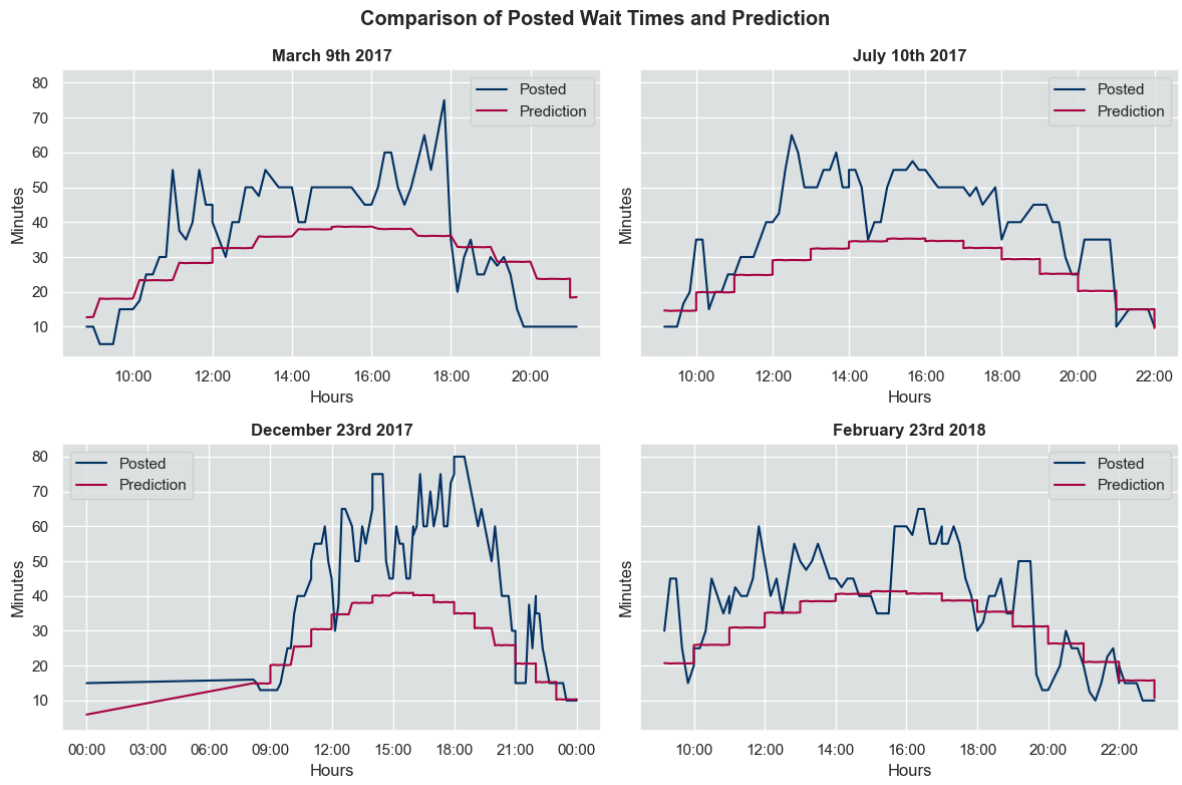

Figure 5: Posted Wait Times and Predictions for Linear Regression with Sine/Cosine Transformation
Source: Own Figure

As can be seen by Figure 5, the performance of our linear regression model improved drastically, with the mean RMSE dropping from 68.96 to 17.32. The model is not a straight line anymore and able to pick up the general trend, though it is also majorly underestimating the peaks throughout the day. On the other hand, there was a deterioration for the tree-based models (rise in RMSE of about 2.8-3.0). This can be explained by the fact that these models split a node based on one (best) feature among a subset. Trigonometric encoding, however, requires the sine and cosine features to be considered together when identifying a distinct timestamp.

For the second time feature transformation approach we used so-called splines. Essentially, the data is divided into sections and for each piece a separate function is fitted (Michelen, 2020). The greater the degree of division, the smoother the outcome. Here the number of sections is matched to the number of values a specific time feature can adopt (e.g. MONTHOFYEAR: 12). In comparison to the trigonometric encoding, this led to a further improvement for the linear model, with the RMSE declining by another 0.418.

The third approach consisted of one-hot-encoding the time features, which yielded the best results for the linear regression (RMSE of 16.631) but has the disadvantage of creating a lot more variables and neglecting their order. Again, with both the second and the third approach, there was a deterioration for the tree-based algorithms. Even without these techniques, though, they were still better performing models.

3.3 Introducing a Lag Feature

The next step was to introduce a new feature, a lag feature, enabling the models to learn from previous wait times. Choosing the right one consisted of a tradeoff between predictive power and context validity. Even though the observations right before, at t-1 (10 min before), showed the highest autocorrelation and would have yielded a lower RMSE, using them wouldn't have fit our use case because visitors and Disney need reasonable time to react to the crowd levels. Therefore, we selected the observations at t-6 (1 hour before).

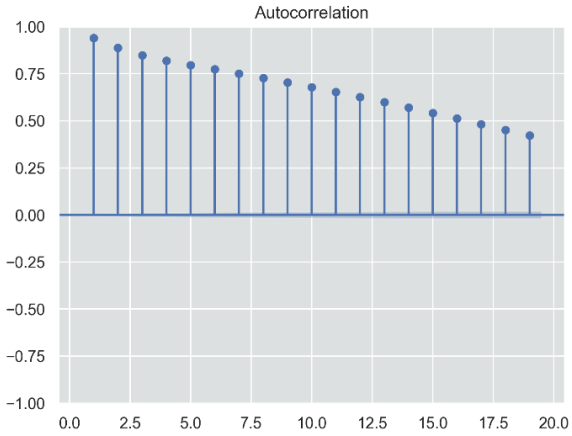

Figure 6: Autocorrelation of different lag features from t-1 to t-19
Source: Own Figure

Cross-validation indicated improved RMSEs of 12.096 for the Random Forest and 11.470 for the Gradient Boosted Trees. In contrast, the linear regression performed worse than with the one-hot-encoded features exclusively. Figure 7 illustrates that the greatest prediction errors were made with higher wait times.

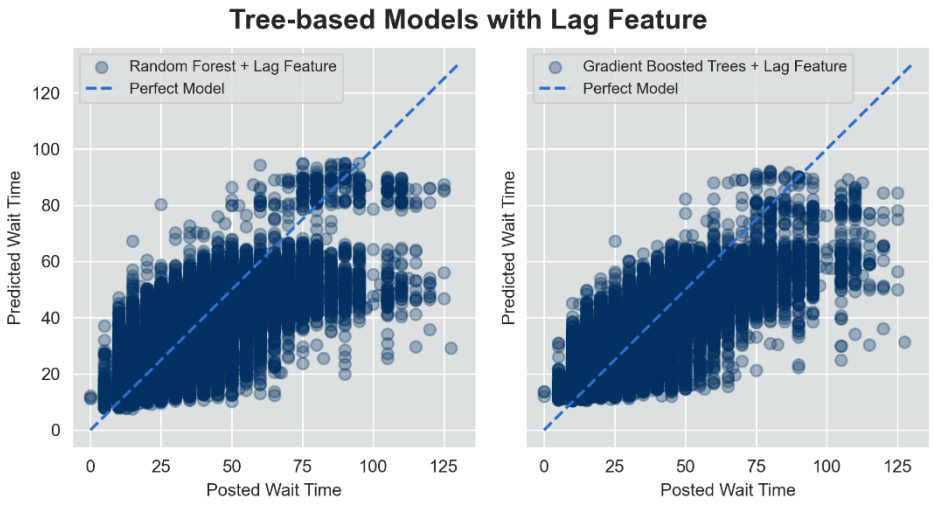

Figure 7: Tree-based models with lag feature prediction accuracy
Source: Own Figure

4 Concluding Discussion

Gradient Boosted Trees with the lag feature proved to be the best predictive model. It performed well with lower wait times up to roughly 50 min but underestimated the higher wait time events. As already mentioned, a limit of our project design is that we predict wait times posted by Disney instead of actual ones, which might differ significantly. This is confirmed by the following two boxplots.

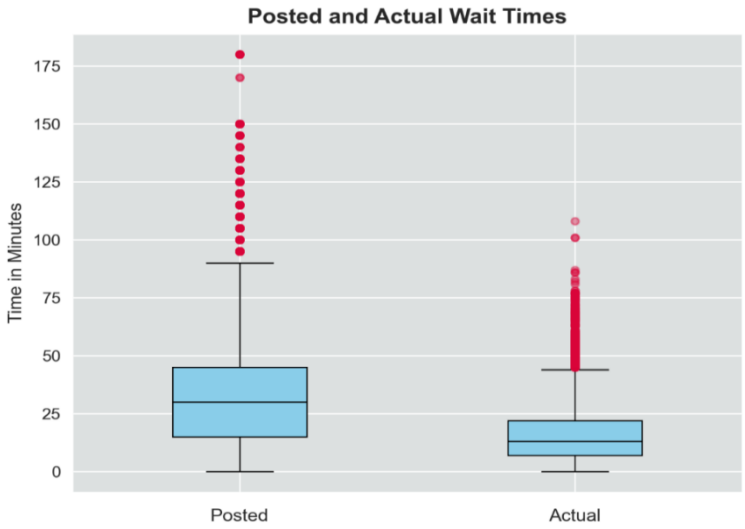

Figure 8: Boxplot of posted and actual wait times
Source: Own Figure

As can be seen, the posted wait times are significantly higher, founding the suspicion that Disney is artificially inflating wait times. In light of this, the described performance issue of our model might actually prove to be advantageous from a customer perspective and provide a forecast closer to the actual times.

One thing that should be kept in mind regarding the use case of crowd management is the effect publishing the expected wait times would have on the behavior of the visitors. It needs to be observed whether the published predictions would lead to more even wait times or just reverse the trend. If the latter outcome is the case, adjustments to the model need to be made in order to influence the behavior of the visitors in the desired way.

References

Glen, S. (2019, July 28). *Decision Tree vs Random Forest vs Gradient Boosting Machines: Explained Simply*. Retrieved from datasciencecentral.com: https://www.datasciencecentral.com/decision-tree-vs-random-forest-vs-boosted-trees-explained/

Mikulski, B. (2019, August 16). *Forecasting time series: using lag features*. Retrieved from mikulskibartosz.name: https://www.mikulskibartosz.name/forecasting-time-series-using-lag-features/

Touringplans (2022). *SAVE TIME AND MONEY*. Retrieved from touringplans.com: https://touringplans.com

Lam, B. (2015, January 28). The Logic of Long Lines. The Atlantic. https://www.theatlanticTouringplans.com/business/archive/2015/01/the-logic-of-long-lines/384870/

Lewinson, E. (2022). Three Approaches to Encoding Time Information as Features for ML Models. nvidia developer. https://developer.nvidia/blog/three-approaches-to-encoding-time-information-as-features-for-ml-models/

Michelen, R. (2020). Using B-Splines and K-means to Cluster Time Series. Towards Data Science. https://towardsdatascience: https://touringplans.com/using-b-splines-and-k-means-to-cluster-time-series-16468f588ea6

Data Sources

“metadata.csv”, Disney World Ride Wait Time Datasets, TouringPlans.com, June 2018, https://touringplans.com/walt-disney-world/crowd-calendar#DataSets, Accessed 11 November 2022

“haunted_mansion.csv”, Disney World Ride Wait Time Datasets, TouringPlans.com, June 2018, https://touringplans.com/walt-disney-world/crowd-calendar#DataSets, Accessed 11 November 2022