# Wait Time Machine Learning Models for Expedition Everest

*Steven Chen*

*June 10, 2019*

## Introduction

Walt Disney World in Orlando, Florida USA is a theme park resort owned and operated by the Walt Disney Company. Consisting of four major theme parks and 34 resorts and hotels on property, Walt Disney World has over 58 million visitors per year.

Expedition Everest (full name "Expedition Everest - Legend of the Forbidden Mountain") is a high speed steel roller coaster thriller attraction (ride) at Walt Disney World's Animal Kingdom that takes guests through simulated Himilaya mountains while encountering a feared Yeti creature. Expedition Everest was built in 2006 and is reported to be the most expensive roller coaster ever built at an estimated cost of $100 million USD. The attraction experience takes three minutes and covers over 4,000 feet of track with the coasters reaching speeds of 50 miles per hour. It is one of the more popular attraction at Animal Kingdom.

Expedition Everest attraction wait time data are used as the basis of this project with the goal of using machine learning to determine a method of predicting accurate wait times.

## Obtaining the Data Set and Data Prep

TourinngPlans.com is a company founded by Len Testa about 20 years ago as a school project to minimize wait times at Disney theme parks. Len collected data and write the first algorithms to optimize the theme park experience. The company currently offers custom tour plan subscriptions to Disney and Universal theme parks as well as Disney Cruise lines and Washington DC and has over 140,000 subscribers.

Theme park attraction wait times since 2012 have been made available for key attractions at the Touring-Plans.com website and consist of a rich data set of actual measured attraction wait times along with an extensive "meta data" set of ancillary information (e.g. theme park open and close times, day of week, time of year, etc.)

After downloaing the files, I merged the Expedition Everest actual wait time data (discarding the "posted" wait time data) and merged it with the daily metadata based on the DATE field. I then cleaned the data by removing missing rows of data, cleaning date/time fields, and removing columns that had data specific to other Walt Disney World theme parks (i.e. Magic Kingdom, Epcot, and Hollywood Studios) but kept all the Animal Kingdom data. I also simplifed the data set by removing columns pertaining to regional school district hollidays and removing columns with non-binary categorical data. The final data file (named EverestData) is a compact 1.2MB file with 29 variables and 6940 observations.

## Data Exploration and Visualzation

The EverestData file consists of 29 variables with 6940 observations. A few basic summary statistics are provided here:

```
## [1] "Wait Time Summary Statistics"
```

```
## # A tibble: 5 x 2
##   Stat   Value
##   <chr>  <dbl>
## 1 Mean    15.5
## 2 Median  12
```

Figure 1: Expedition Everest Attraction

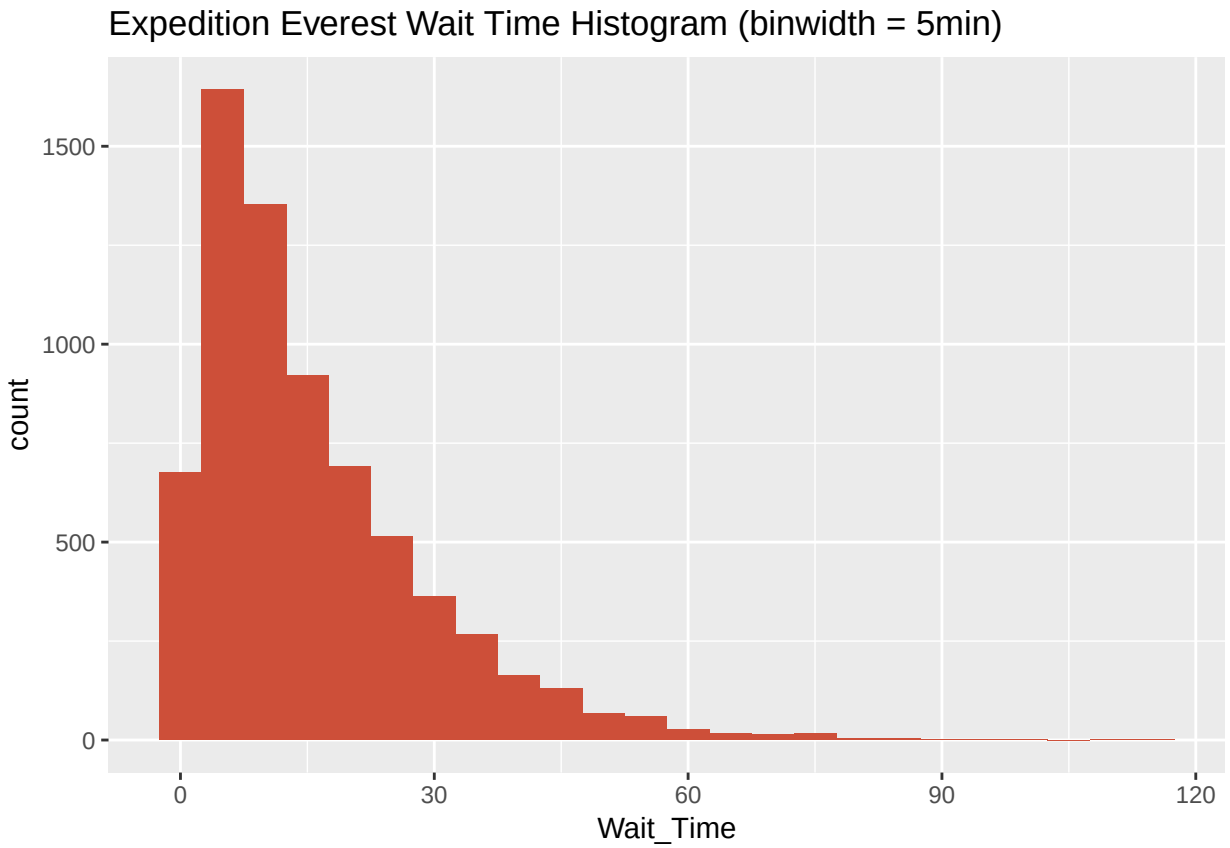
```
## 3 SD      13.5
## 4 Min      0
## 5 Max    114
```

```
## [1] "Wait Time Quantile"
```

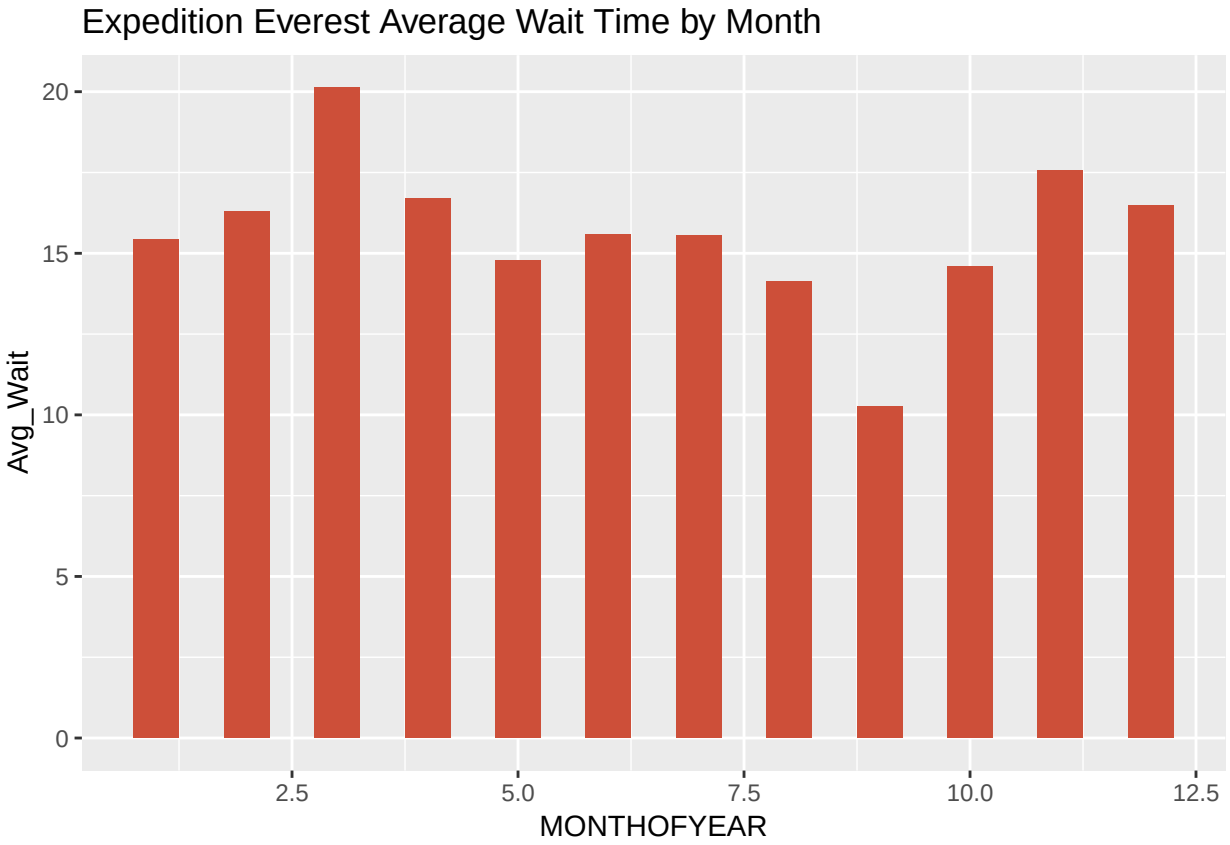
```
##   0%  25%  50%  75% 100%
##    0    6   12   22  114
```

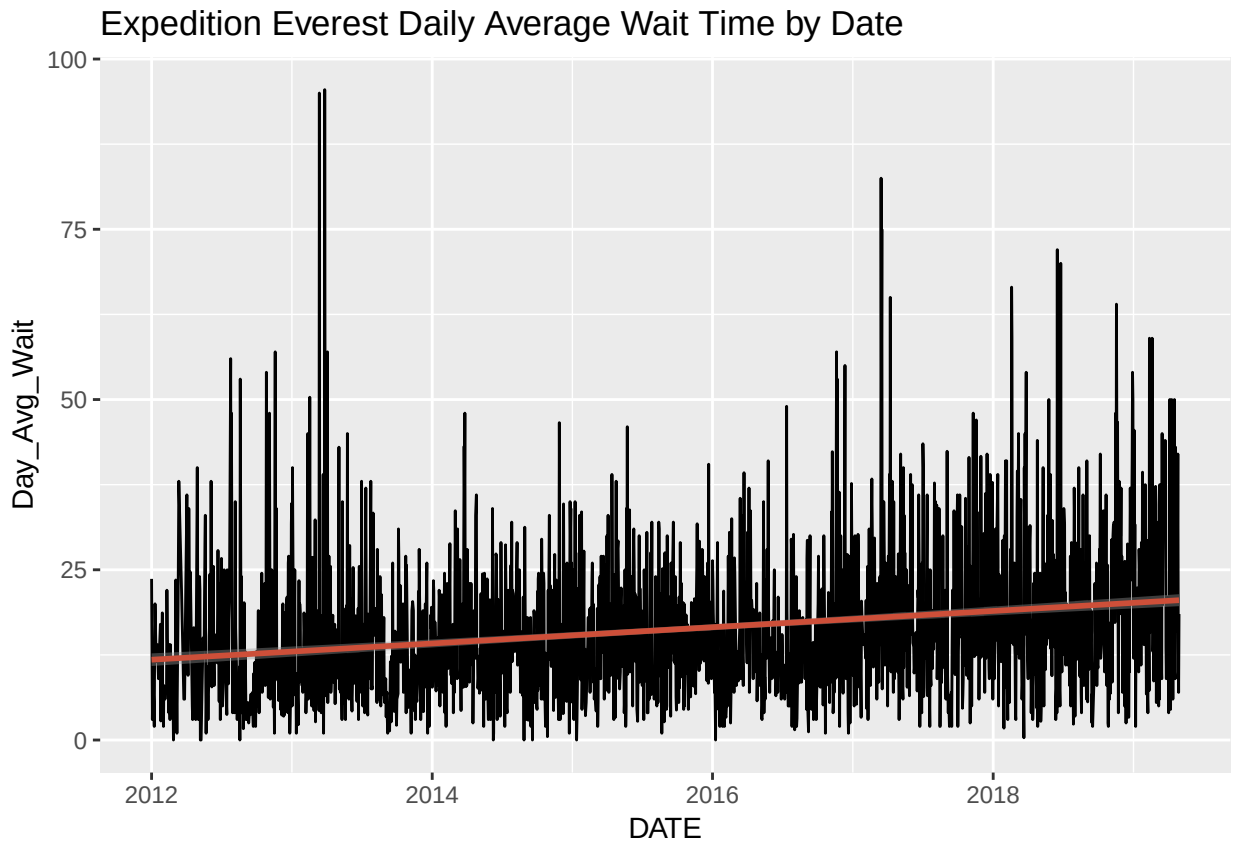
It is interesting to see that the standard deviation is quite close ot the mean wait time. This indicates a relatively broad distribution of wait times at the attraction.

Visualizing the datat will make this more apparent.

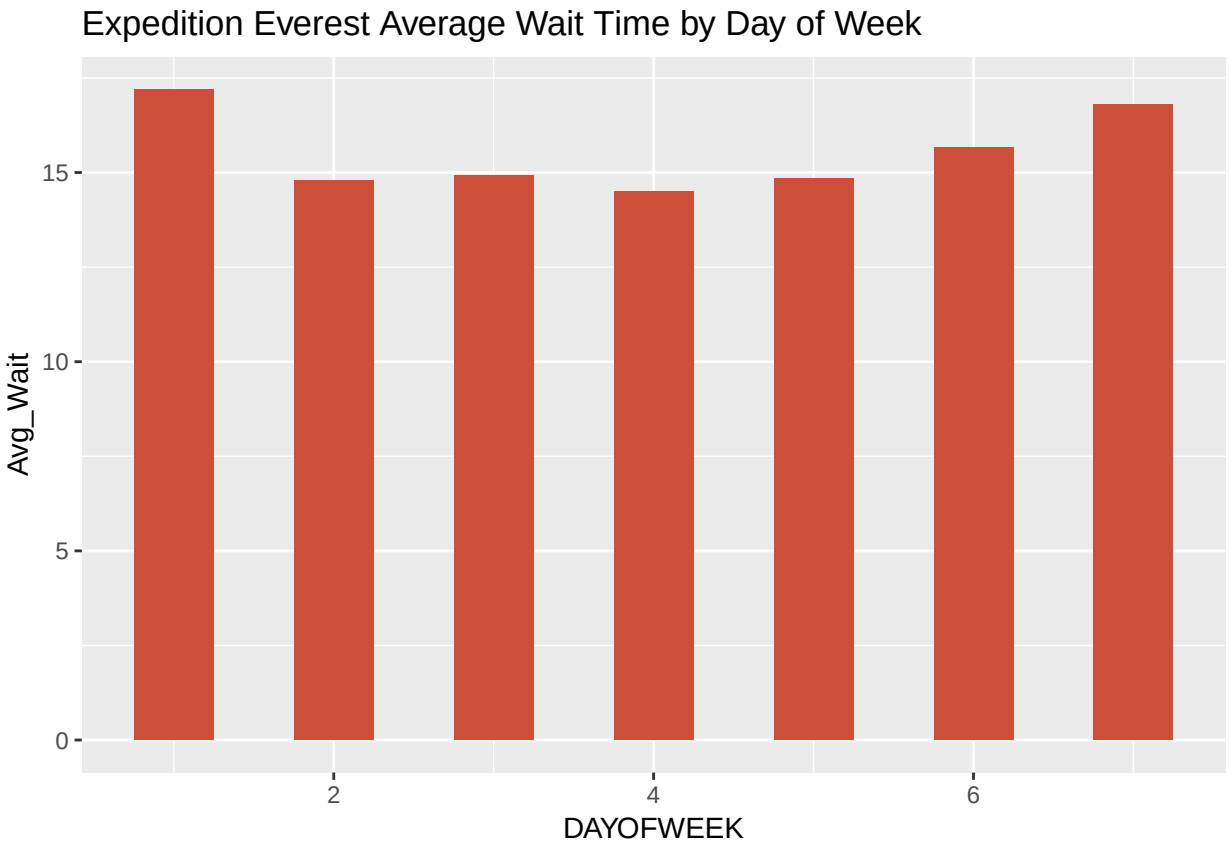



From the historgram we see a large number of short wait times with a long right hand tail that goes to 114 minutes. While a popular ride, wait times appear to be relatively short.

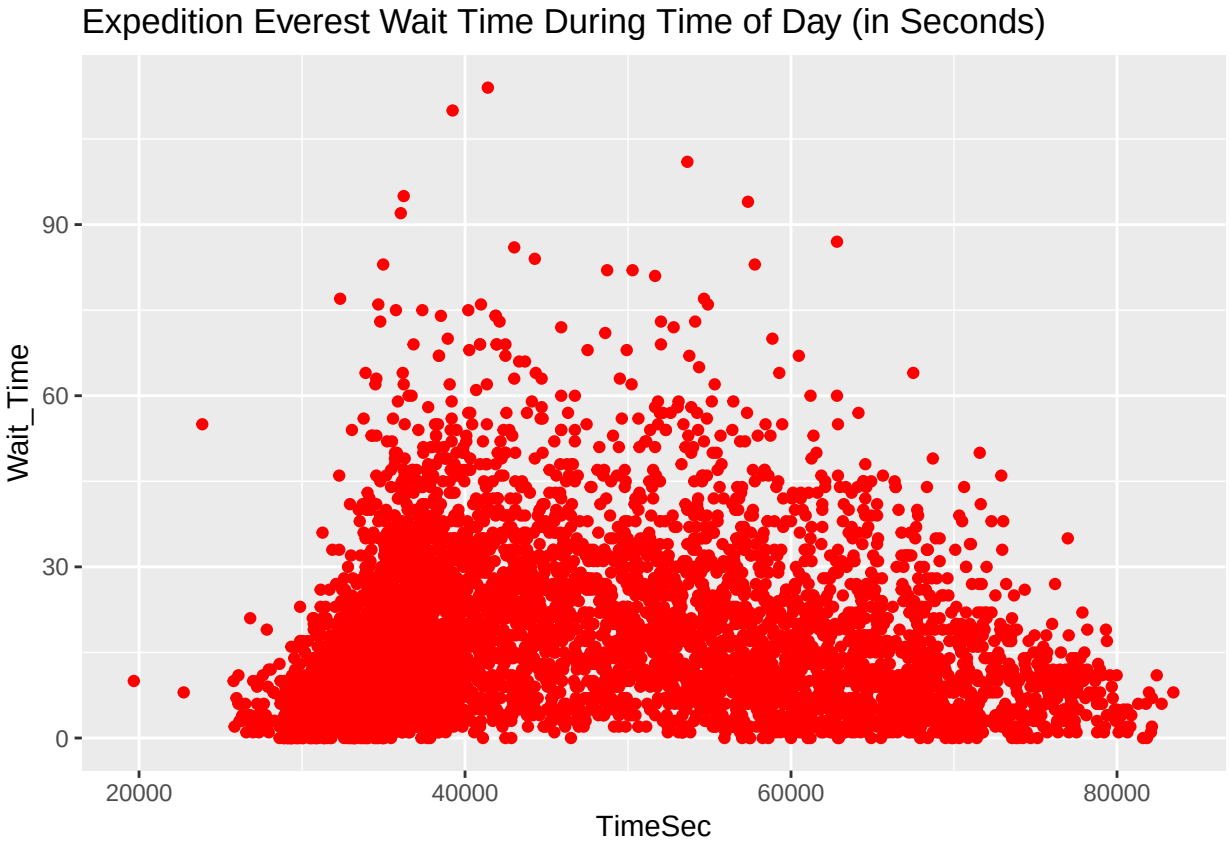

Expedition Everest Average Wait Time by Month
Avg_Wait
20
15
10
5
0
2.5
5.0
7.5
10.0
12.5
MONTHOFYEAR




Plots of average wait time by month and wait time by day indicate there is some sort of seasonality happening. Longest wait times occur around March (possibly due to Spring Break holiday) as well as the November and December timeframe (possibly Thanksgiving and Christmas holidays?) The lowest average wait occurs in the month of September. The average wait time by date plot shows us how "spiky" the data can be with variations within a day. The trend appears to be for the wait times to be increasing over time.




Plotting average wait times by day of week clearly show higher wait times on the weekends and Fridays with lower wait times Monday through Thursday.




Finally, plotting wait times against the time of day clearly show a Gaussian (bell) distribution with an increase in times towards the begining of the day and and decrease in the evening hours.

From the visuals, it appears riding Expedition Everest in the month of September, on a Monday - Thursday, either early morning or late evening would result in the shortest wait times. But can we actually use machine learning to predict this? Are there other factors that would influence wait time?

## Machine Language Approaches

Before applying machine language approaches to fit the datat, training and testing (validation) data sets were created from the EverestData. A partition was used so 20% of the data was in validation set while the remainder was in the training (Everest_Train) set. 20% was chosen to speed the training algorithms. Finally, Root Mean Squared Error is chosen as an accuracy measure so that is coded as well.

Once that was done, an array of machine language approaches was applied to the data.

For reference, a "Naive" model is used to determine a baseline for improvement. The naive model uses the average wait time as the prediction variable

```
## # A tibble: 1 x 2
##   method          RMSE
##   <chr>          <dbl>
## 1 Naive Approach  13.9
```

Not surprisingly, the RMSE is quite large given the distrbution of the data.

## Linear Regression Model Approach Results

A simple linear regression model was created using factors such as DAYOFWEEK, DAYOFYEAR, TimeSec, etc. as used in the visualzations.

```
## # A tibble: 2 x 2
##   method                 RMSE
##   <chr>                 <dbl>
## 1 Naive Approach         13.9
## 2 Linear Model Approach  13.7
```

The linear approach improved the predictive power of the model slightly.

## K Nearest Neighbor Approach Results

A K Nearest Neighbor model was created using the Caret package.

```
## # A tibble: 3 x 2
##   method                        RMSE
##   <chr>                        <dbl>
## 1 Naive Approach                13.9
## 2 Linear Model Approach         13.7
## 3 K Nearest Neighbor Approach   11.7
```

KNN improved the RMSE by almost 1.2 minutes.

## Regression Tree Approach Results

A K Nearest Neighbor model was created using the Caret package.

```
## # A tibble: 4 x 2
##   method                        RMSE
##   <chr>                        <dbl>
## 1 Naive Approach                13.9
## 2 Linear Model Approach         13.7
## 3 K Nearest Neighbor Approach   11.7
## 4 Regression Trees Approach     11.6
```

Regression Tree improved the RMSE slightly.

## Random Forest Approach Results

The randomForest package is used to apply a random forest algorithm to the data.

```
## # A tibble: 5 x 2
##   method                        RMSE
##   <chr>                        <dbl>
## 1 Naive Approach                13.9
## 2 Linear Model Approach         13.7
## 3 K Nearest Neighbor Approach   11.7
## 4 Regression Trees Approach     11.6
## 5 Random Forest Approach        11.6
```

There is no improvement in RMSE with Random Forest approach over Regression Tree. Perhaps the Regression Tree is already optimized?

## Ensemble Approach Results

Finally, and ensemble approach averaging KNN and Random Forest was applied to the data.

```
## # A tibble: 6 x 2
##   method                                 RMSE
##   <chr>                                 <dbl>
## 1 Naive Approach                         13.9
## 2 Linear Model Approach                  13.7
## 3 K Nearest Neighbor Approach            11.7
## 4 Regression Trees Approach              11.6
## 5 Random Forest Approach                 11.6
## 6 Ensemble (KNN + Random Forest) Approach  10.1
```

This led to an improvement in the RMSE reducing RMSE to 10.3 seconds.

## Conclusions

Using a KNN and Random Forest ensemble approach resulted in the most accurate wait time prediction model. The RMSE of 10.3, while an improvement of 3.8 minutes over the Naive model, still seems relatively large. In the future it would be good to include the categorical data (converting to binrary or other numerical values) as well as data from other theme parks to see if this has an impact on the RMSE. What's not clear is how to acutally use the model to predict lower times. Perhaps this can be determined by applying the y_hat to the validation data to see what factors are the greatest attribute.

## References

Reference materials for this project.

Touring Plans website - https://touringplans.com/

Touring Plan dataset - https://touringplans.com/walt-disney-world/crowd-calendar#DataSets

Business Insider Interview with Len Testa and Touringplans.com - https://www.businessinsider.com/touringplans-disney-world-len-testa-interview-2019-5