Bike Share Pricing

By Samuel Pederson.

It should be no surprise that bike sharing has been on the rise for many years as we see them popping up more frequently in our towns and cities. But are these companies maximizing their potential profits? In this article, I will answer this question.

Fig 1: Boston Bike Share Station, credit: Unsplash (Kelly Sikkema)

While bike share programs have existed in various forms going back to the 1960s in places like the Netherlands (WRI, 2019), no one could have predicted what they have become. Since 2007, more than 1,500 new bike share companies have been started globally (WRI, 2019), with expected revenue growth of 10.9% leading to expected total revenue of 7.96 billion in 2022 (Statista, 2022). Needless to say, there is a lot of expectation about this rapidly growing market. Of these numerous new companies, many pricing schemes have been developed with the most common among these being a flat price with a low-value additional fee if your bike is returned late. While this has worked well for companies up till now, that does not mean it cannot be improved. Thus I asked myself how can I utilize data to create a pricing scheme that suggests a price based on how many people are expected to make use of a given bike share system. This question and more I plan on answering in this article.

Part 1: Data Wrangling

The dataset used for this project was collected from the UCI Machine Learning Repository (I strongly recommend checking them out as they have many great free datasets to choose from). The dataset had information about ridership for each hour taken over 2 years. I'll briefly describe the features of the dataset below, save for the index column:

- dteday : The given date
- season : The given season (represented as numbers 1-4)
- yr : The given year (0 = 2011, 1 = 2012)
- mnth : The given month (represented as numbers 1-12)
- hr : Hour of the day (represented as numbers 0-23)
- holiday : True if a given day is part of a list of holidays, false otherwise
- weekday : Day of the week (represented as numbers 0-6)
- workingday : True if the day is neither weekend nor holiday, false otherwise
- weathersit : Type of weather at time (represented as numbers 1-4 where 1 is best conditions and 4 is worst)
- temp : Normalized temperature in Celsius
- atemp : Normalized feels like temperature in Celsius
- hum : Normalized humidity
- windspeed : Normalized windspeed
- casual : Count of casual users
- registered : Count of registered users
- cnt : Total count of users

The data retrieved from the UCI repository was in a mostly clean state when I got it, so only a few changes were made to the initial dataset. Primarily what I did was change the dteday feature to a DateTime object and renamed the row to date while getting rid of the yr, mnth, and instance index columns as they were redundant.

Part 2: Exploratory Data Analysis

The key goal for the EDA section of my project was as it is during any project, trying to find out what features relate to each other. As my main goal was trying to figure out what changed the total count, I first looked at a correlation matrix to gather an idea of what features were highly correlative (seen below in figure 2).

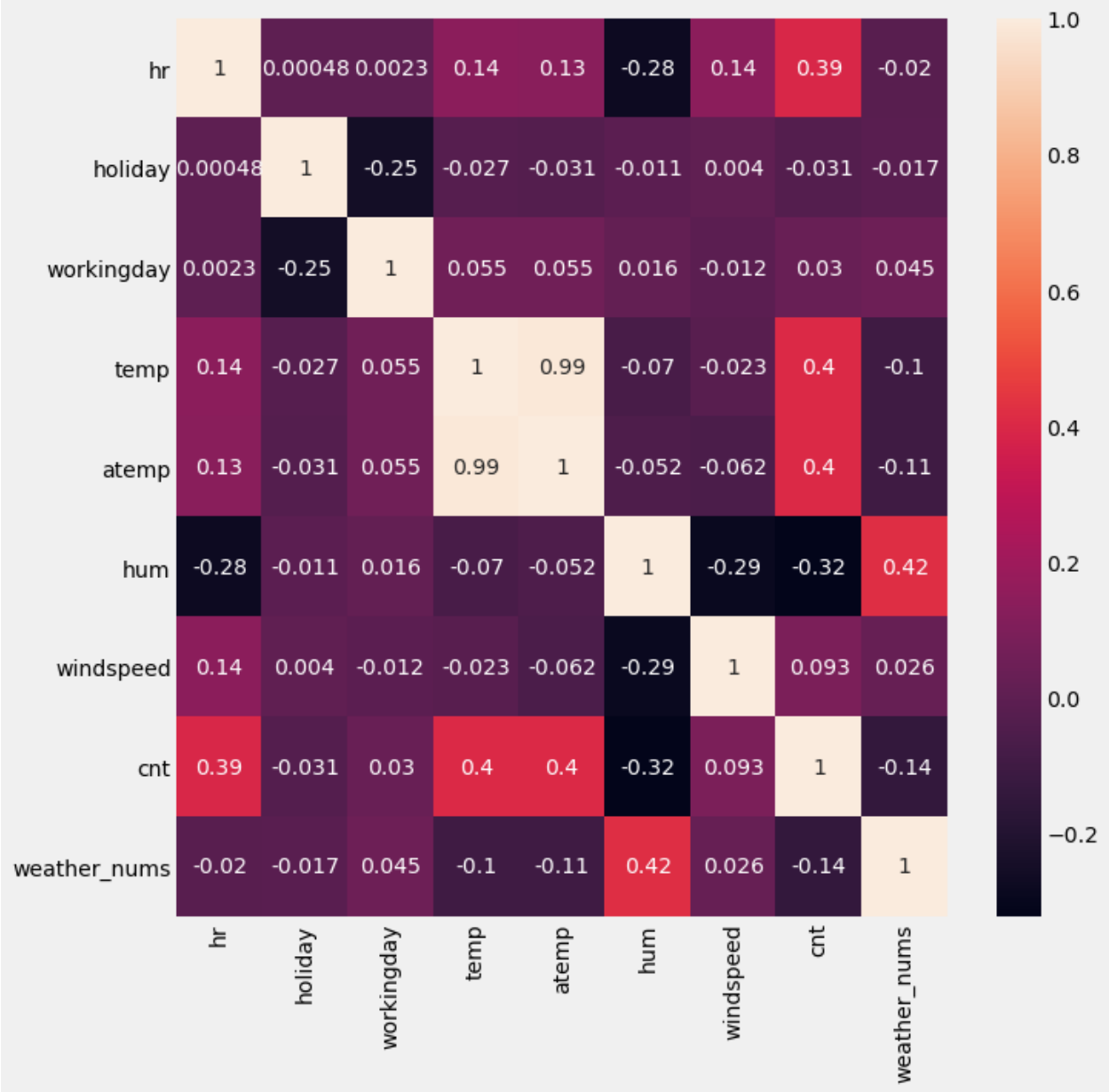

Fig 2: Correlation Matrix of Dataset Features

What this shows is that the key feature of the total count (cnt) has a high positive correlation with the hour of the day, the temperature, and the feels-like temperature (albeit they are mostly the same so this logic follows). Additionally, it has a high negative correlation with the humidity and weather situation columns (weathersit had been renamed to weather_nums before this was run). The correlation for each of these made a good amount of sense to me, for example, more people want to go out on a bike when it is warm and fewer people would be inclined to go out if it was very humid. The one thing that made me ponder was why none of the correlations were very high on their own. For the hour of the day I could understand not a super high correlation as while it would generally go up during the day it would end low which would skew the correlation. As I wanted to know more about these correlations I ended up graphing them all out to see what nuances there were as shown in figure 3a-3e below.

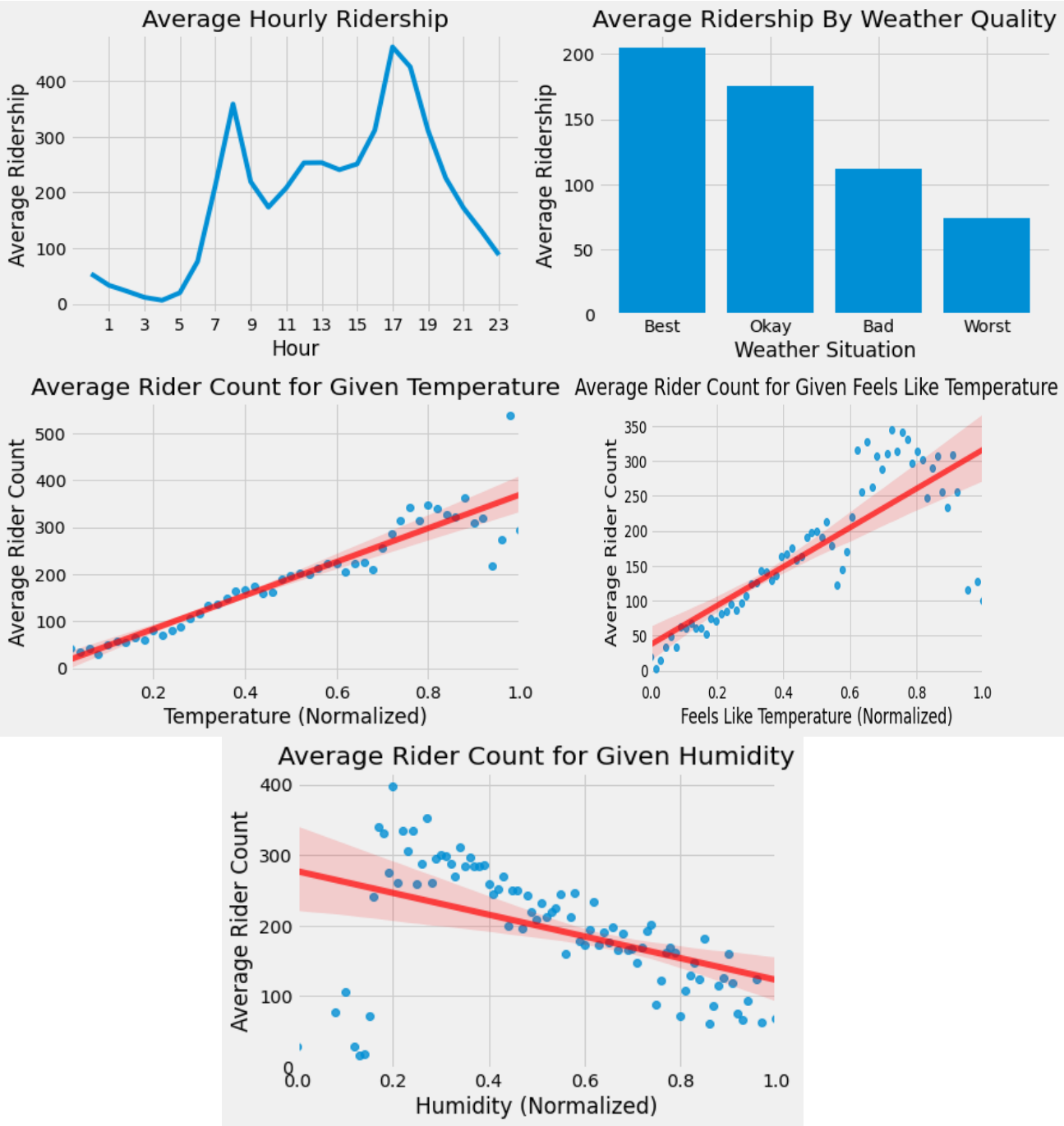

Fig 3: a.) Average Hourly Ridership, b.) Average Ridership by Weather Quality, c.) Average Rider Count by Given Temperature, d.) Average Rider Count by Feels Like Temperature, e.) Average Rider Count by Humidity

I could go over each of these graphs individually, but suffice it to say each shows strong patterns, which lead me to believe that a model like a random forest would work well in capturing these nuances.

Part 3: Pre-Processing and Modeling

I decided to combine both of these stages as the pre-processing stage was mostly completed for me, as the dataset that I initially collected from the UCI repository was already normalized, at least for the continuous values. For the categorical variables, I reassigned the numerical values to descriptive strings where applicable (not in the hour of the day for example) and used the pandas get_dummies method so that each feature was understandable. The result of this process was a dataset with 43 features in total, 1 of which (the count) would then be set as my y variable and the rest as my x variables for my train_test_split call. I decided to use a 75-25% split for this as I felt that would give my model enough to train on while still having a good amount of examples to test on. And with that done I moved on to modeling.

For the modeling, I initially chose to run 7 different models without touching any of the hyperparameters, those models being:

- Linear Regression
- Ridge Regression
- K Nearest-Neighbors Regression
- Radius Nearest-Neighbors Regression
- AdaBoost Regression
- Random Forest Regression
- Multi-Layer Perceptron Regression

Of these models, the Radius Nearest-Neighbors regression was not well enough suited to the dataset and was thrown out. The rest of the results of the comparison can be seen below in figures 4 and 5.

	model_name	model	train_r2	test_r2	train_rmse	test_rmse	time
0	Linear	LinearRegression()	0.627916	0.624570	111.049976	109.878036	0.043496
1	Ridge	Ridge()	0.627916	0.624411	111.049905	109.901374	0.018978
2	KNN	KNeighborsRegressor()	0.852828	0.768656	69.841021	86.253389	4.504978
4	AdaBoost	(DecisionTreeRegressor(max_depth=3, random_sta...	0.371221	0.362620	144.360033	143.167946	1.008274
5	Random Forest	(DecisionTreeRegressor(max_features='auto', ra...	0.974020	0.822423	29.343723	75.568310	10.910859
6	Multi-Layer Perceptron	MLPRegressor()	0.811825	0.806543	78.972959	78.874909	15.670602

Fig 4: Model Training Table with R^2 and RMSE results

Fig 5: RMSE and R^2 Train and Test Results for Each Model Visualized

From this, I figured that the best model for hyperparameter tuning would be the Random Forest Regression. While it may have taken slightly longer to finish the modeling, once it has fit the data it does not need to run through the dataset every time new data is added like a KNN model would and it outperformed every other model even if it did overfit to the dataset a bit. With that, I moved on to hyperparameter tuning through GridSearchCV and ended up with my best hyperparameters being min_samples_split=8 and n_estimators=500. This helped a lot with the overfitting and slightly raised the R^2 score (from .822 to .828) and slightly lowered the RMSE (from 75.6 to 74.3). With this, I was happy to have a model that predicted with reasonable accuracy.

Part 4: Pricing Suggestion

Now I can't forget the reason I did all of this was to create a dynamic pricing system for bike share programs. The way that I decided to do this was under the understanding of a currently flat pricing system, which is how many such systems currently operate. Using the New York City bike pricing as a baseline (they charge ~$4 per trip), I created three new pricing models that take the predicted number of riders for the next hour to separate into different pricing tiers, where when more people are expected to use the service the price gets higher and when low ridership is expected it lowers the price. The three plans I created I called the low, medium, and high-risk solutions. Each is based on setting the cutoff points on the 33rd and 66th percentile of ridership and assigning a new price based on if the predicted ridership for the next hour is in the lower, middle, or upper third of ridership. Pricing scheme 1 (low-risk) offers a 50-cent discount if predicted ridership is at or less than the 33rd percentile, the same ($4 in this case), and a 50-cent increase in price if predicted ridership is above the 66th percentile. The second pricing scheme is the same as the first without adding a discount for ridership below the 33rd percentile. This would not incentivize people to make use of the service as much at lower times of popularity, but that is outside the scope of this investigation. The final pricing model is a variation of the first with a lower floor and a higher ceiling, with the lower payment amount being $1 off ($3) and the higher tier being $1 extra ($5). While each of these suggestions would increase expected profits as shown below in figure 6, caution must

always be taken when introducing a new pricing plan, so I would suggest the low-risk option as then you can incentivize more people into using the program with lower prices sometimes, but I digress.

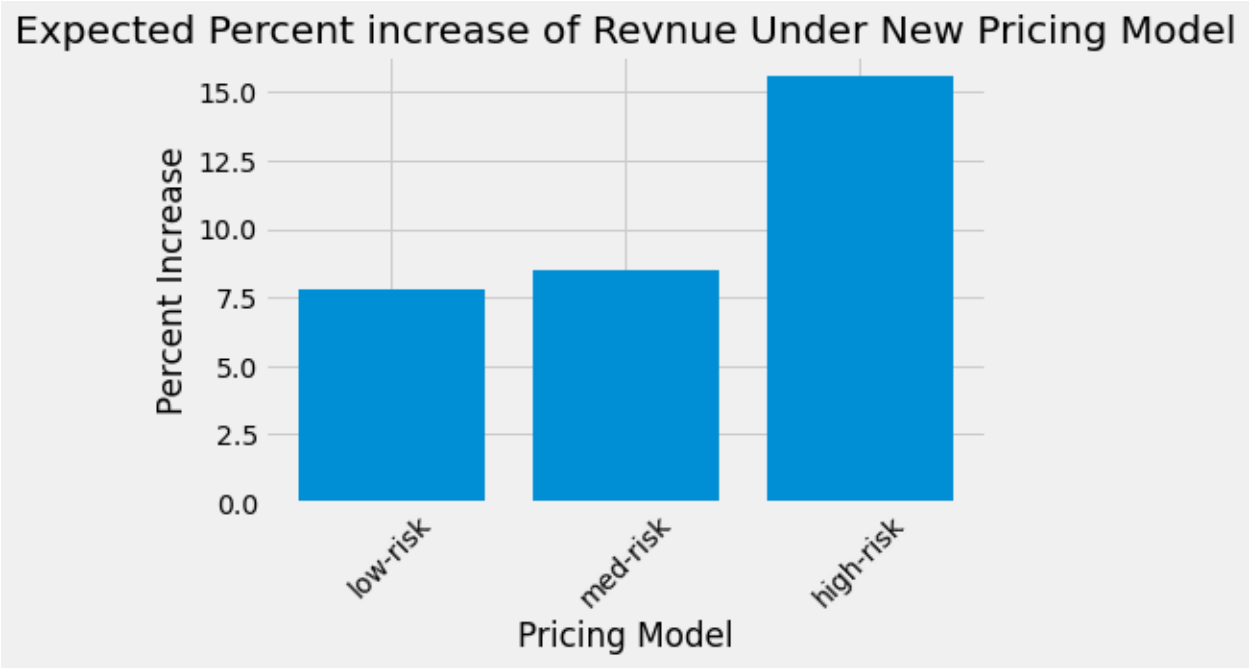

Fig 6: Percent Increase of Revenue under Each Pricing Plan

Part 5: Final Thoughts

While I am happy with the results of this study, that doesn't mean there isn't more that could be done. One idea I have for improving what I have done would be to create a pricing model that works on a scale rather than on break-off points, as this could lead to better revenue generation without alienating the audience (think Uber or Lyft's varied pricing models). While this could have been done here I was just trying to show the potential of a varied approach and doing so the way I did while not without flaws if far easier to visualize and explain. Further research into this problem is possible and I highly recommend it to all who are interested!

Pic Source:
https://unsplash.com/photos/pE7X4R7dW3o

Info Sources:

https://www.wri.org/research/evolution-bike-sharing#:~:text=Bike%20sharing%20began%20in%20the,expanded%20over%20the%20last%20decade.

https://www.statista.com/outlook/mmo/shared-mobility/shared-rides/bike-sharing/worldwide

https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset#