Bike Share Modeling Project

Ivan Novikov

DS 270

# Intro

I did my Machine Learning Project with regressions on a Bike Sharing Data set from the UCI Machine Learning Repository. The idea of this model is to predict how many people will rent bikes given certain conditions. It is useful for a city to have this information in order to make urban planning decisions and to know how well the bike sharing system is working. This is interesting to me because in my younger years I used to be a big biker and hopefully a model like this could do something to further bike friendly policy.

**Potential Urban planning use for a model created with these intentions**

With a model that predicts well we could see what the predicted value would be for the conditions in a given day. If the actual value was very far off or if there was a pattern of the actual value being very far off, we could see that there is an anomaly in how much the bikes are being used. If this anomaly was that there were more bikes rented than expected, this could be used as a justification for bike friendly policy. For example, additional bike lanes/routes could be argued for with analysis from a model such as this one. The opposite could be true, but the status quo of our culture is typically enough to build new roads and expand existing ones.

Another use for this model and data is to gain insight into the biking behaviors of Washington DC tourists vs residents. The target variable of the regression, cnt or count, is the total number of bikes rented at a given hour. Count is the addition of “casual” and “subscriber” meaning there are people who can choose to rent the bike once, and people who subscribe to the bike share system and get a discount on the rentals. It can be safely assumed that 100% of the people that are subscribers are residents of Washington DC or the surrounding area. We don’t know the proportion, but since DC is a heavy tourism destination, it can be assumed that a substantial number of the casual riders are tourists. In this analysis I focus heavily on the total, but it could be easily redone with “casual” and “subscriber” as the target variables to gain some sort of marketing insight for the DC tourism industry.

# Model Explanations

**Linear Regression**

Linear regression is a supervised machine learning prediction method which assigns variables with weights based on how correlated they are with the target feature. In a simple case with a single variable, linear regression creates a line of best fit on a plane where y axis represents values for the target variable, and x axis represents the values of the independent variable. The line is created so that the total distance between every data point and the line is minimized. The distance between the line and each individual data point is squared and then added together, this is called the sum of squared residuals. Reducing this number as low as it can go is how the slope and intercept of the line are decided. Dividing this number by the number of data points also gives us an important value, the mean squared error or MSE. The MSE is how the accuracy of the model is judged. The higher the MSE, the less the ability of the model to accurately predict. The Square root of the MSE lets us know how far the average point is away from our data points. The Square MSE is essentially how good the model is in terms of the data points that we are dealing with. This is how the model from this project works but instead of being on a 2 dimensional plane, our model and line is on a many dimensional plane

**Gradient Descent**

Gradient Descent is a computerized method for finding the weight of a variable that would create the line with the lowest error. In a regression we do it with the sum of squared residuals but in gradient descent we do it by having a computer algorithm create many lines of best fit, each one with a progressively smaller MSE. The computer algorithm does not do this randomly. There is an imaginary parabola where the y axis is our function and the x axis represents potential weights for the dependent variable. The algorithm travels down this parabola towards the point where the derivative of the parabola = 0. In gradient descent, we decide how big the steps are towards the bottom the parabola, how many steps we take, and how many data we use to inform the direction of the proceeding steps.

# Data set explanation

*Bike Sharing* is a system of bike rental in urban areas. The way it works is there are bike racks around a city that hold bulky looking bicycles. One can pay a little bit of money to rent a bike and return it to the same bike rack or one at another location. The idea of bike sharing is that it would decrease congestion and car use because biking would be made more convenient. On paper it is a good idea because there are many ways in which it makes biking more convenient. One does not have to own a bike, one does not have to haul a bike up the stairs in their apartment, one does not have to have a bike on them in the city if they feel the desire to ride one. In this bike sharing program, one can rent a bike casually or one can sign up to do so regularly and receive a discount. Downtown Kansas City has a bike sharing program.

This data set is from when Washington D.C. set up a bike sharing system in 2010. There is data for 2011 and 2012. The data includes weather features, time features, how many people rented a bike casually, how many subscribers rented a bike. There are two data sets, one breaks down conditions and riders hourly, one summarizes the totals for the day.

Features:

**Instant:** This is just a record/row id

**Dteday:** the date

**Season:** Spring/summer/fall/winter

**yr**: Year 2011=0 2012 =1

**mnth:** mnth total

**hr:** hour 1-24

**holiday:** cell=1 if day was a holiday

**workingday:** cell =1 if the day is not a weekend day or a holiday

**weahthersit:** 1: Clear, Few clouds, Partly cloudy, Partly cloudy  
- 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist  
- 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds  
- 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog  
**temp** : Normalized temperature in Celsius. The values are derived via (t-t\_min)/(t\_max-t\_min), t\_min=-8, t\_max=+39 (only in hourly scale)  
**- atemp:** Normalized feeling temperature in Celsius. The values are derived via (t-t\_min)/(t\_max-t\_min), t\_min=-16, t\_max=+50 (only in hourly scale)  
**- hum:** Normalized humidity. The values are divided to 100 (max)  
**- windspeed:** Normalized wind speed. The values are divided to 67 (max)  
**- casual:** count of casual users  
**- registered:** count of registered users  
**- cnt:** count of total rental bikes including both casual and registered

The daily data set has the same features but does not include the hour feature.

The data set is very complete, no missing values. I imagine this is because the ride sharing program was designed to collect data and because there is always intricate weather data.

The original data set data set is a singular column separated by ‘**;**’ . I put the data set through KNIME with the separator argument of, ‘;’ in order to make a csv file with individual columns that was much easier to work with in excel. Playing with the features/feature engineering was done in excel.

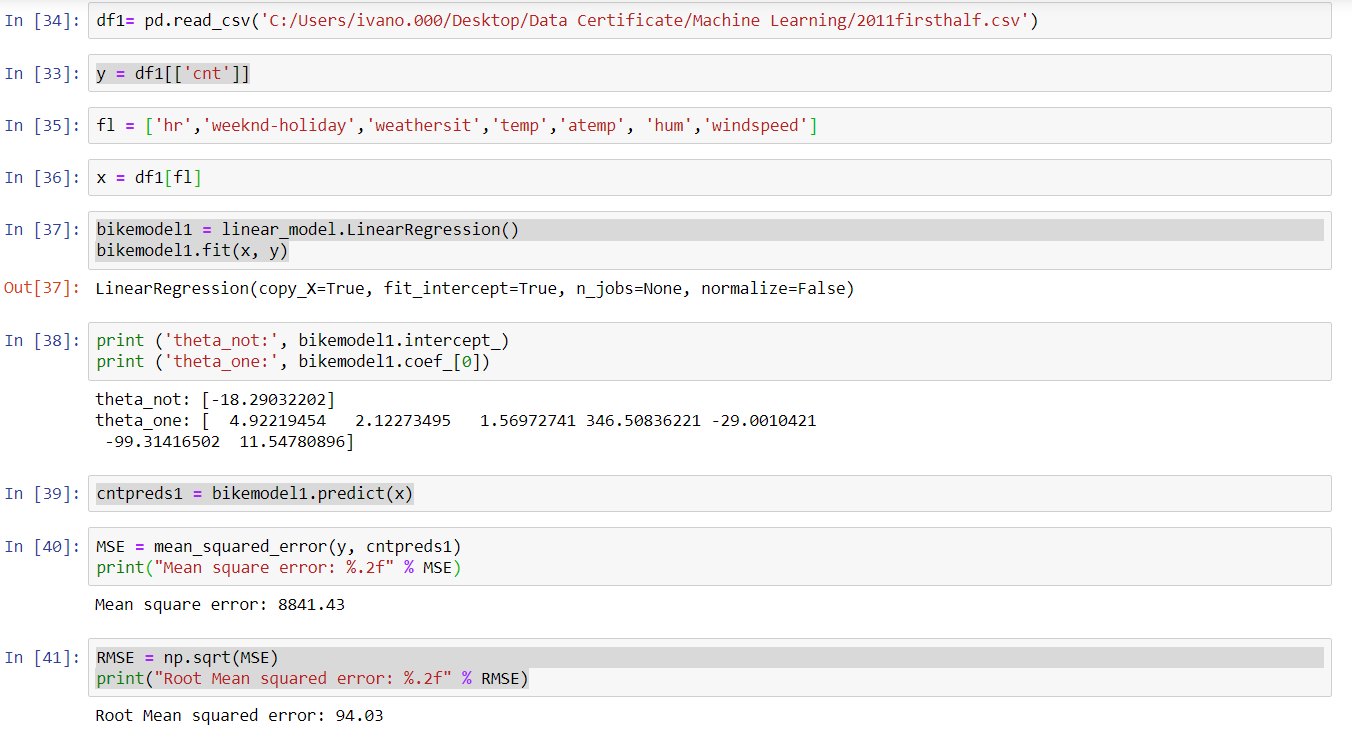
There are 17378 records and 16 features in the hourly data set.

There are 730 records and 15 features in the daily data set.

# Time as a confounding variable

This data was taken in the begining years of the Bike Sharing Program’s existence therefore time is a large and difficult to calculate confounding variable. When the program is initially set up, the weather and time of day are not the only variables that decide how many people will ride. People need to learn of the program’s existence, and to entertain the thought of using it. What this means is that in the first year at least, the increase in riders reflects more people trying the bike share program as much as it reflects the weather and time of day.

In order to exhibit this, I took the hourly data set, isolated the data from 2011, split it into the first half of the year and the second half of the year.





The error is higher for the second half of the year. This could mean many things, but I interpret this to mean that how many people bikeshare is much more variable the more people know about the program’s existence. This was against my expectations, I thought that the features would become more predictive as more people learned about the bike sharing system. This is also exhibited at the end of the Feature Engineering Workbook and in the visualizations towards the end of this paper.

This is the main caveat for this data analysis. There is an undefined variable for each given day that’s how much of the population is aware of the bikeshare program and considering the use of it. In the time period of this data, I assume that it is steadily increasing. It would be much more ideal to have data for a time period where we can reasonably assume that this variable is somewhat constant.

# Feature Engineering

The main part of this project is feature engineering in order to create a more accurate regression model. The model was fit and run on the entirety of the hourly data set. I am going to explain the features that I created and attempt to explain why they reduce the error. I created features based off the weather conditions and as well as based on the time features. I did the feature engineering by creating conditional formulas applied to newly created columns in Excel. This allowed me to take actions more complicated than were displayed in the feature engineering competition lab as I do not yet have a mastery over python syntax yet.

Most of my created features revolve around less than/greater than thresholds. I would

1. play with the values in the formula in Excel
2. save the workbook
3. plug the csv into the workbook
4. run the regression with the new value thresh holds
5. Evaluate the error of the model with means squared error
6. The value thresh holds that decreased error the most would be the ones kept in the engineered features.

Weather features:

**Cold:** If the ‘temp’ features is less than .5, this features equals 1. If temp is less than greater than .5, the feature equals 0. Mean Square error was reduced by **45.87**. The logic behind creating the feature is that people do not want to ride bikes when it is cold. I do not think that this logic is necessarily validated because it moved the mean squared error relatively little.

**Hot:** This feature is based off temperature and humidity, temp and hum features. If temp is higher than .6 and hum is above .7, this feature equals 1. If these conditions are not met, the feature equals 0. Apparently heat alone does not affect biking behavior in Washington D.C. Error did not decrease when a feature was made with a greater than temp condition alone. Mean Squared Error was reduced by **163.35**. People do not want to ride bikes when it is hot and humid. I do not know error was decreased enough to validate this feature.

**Windy:** This feature is based off the windspeed feature. If windspeed is greater than .45, the feature equals one. If it’s less than .45 it equals 0. Error was reduced by **31.49**. The logic behind this feature is that it is unpleasant to bike when it is windy, but the error was not reduced enough to validate this logic.

Overall, the weather features did very little to decrease error. This is how much the error was reduced when the value thresholds were optimized. I imagine that their predictive power is diluted by the hour time feature. It does not matter how nice it is outside, people are not going to go biking at 3 am.

If they were more powerful, they could be used to model bike use during a season. Averages from the past could be used as a prediction for future weather conditions in a given season. From these predictions, the regression could be run, and a prediction could be made.

More importantly, the model could be run on a time period in the recent past to see if there has been an anomaly in bike usage.

Time Features:

**Seasonflip:** There is a season feature where winter = 1, spring =2 , summer =3, fall = 4. I changed it to winter = 1summer = 2, fall and spring = 3. Mean Squared Error was decreased by **18.27**. The logic is that since fall and spring have the nicest biking weather, they should be weighed the most heavily.

**Daylight:** My most powerful and successful feature. If it is 7 am through 8 pm, this feature equals 1. Error was decreased by **4853.73**. It is dangerous to ride bikes at night, and rarely do people have time and desire to do so.

Overall, the square root mean squared error decreased from **141.84** to **122.50**. This is not very good because the average value for the cnt variable is about 189.46.

When the model is fit and run on just the data from 2012, the MSE increases significantly likely because of the explained confounding variable.

# Gradient Descent

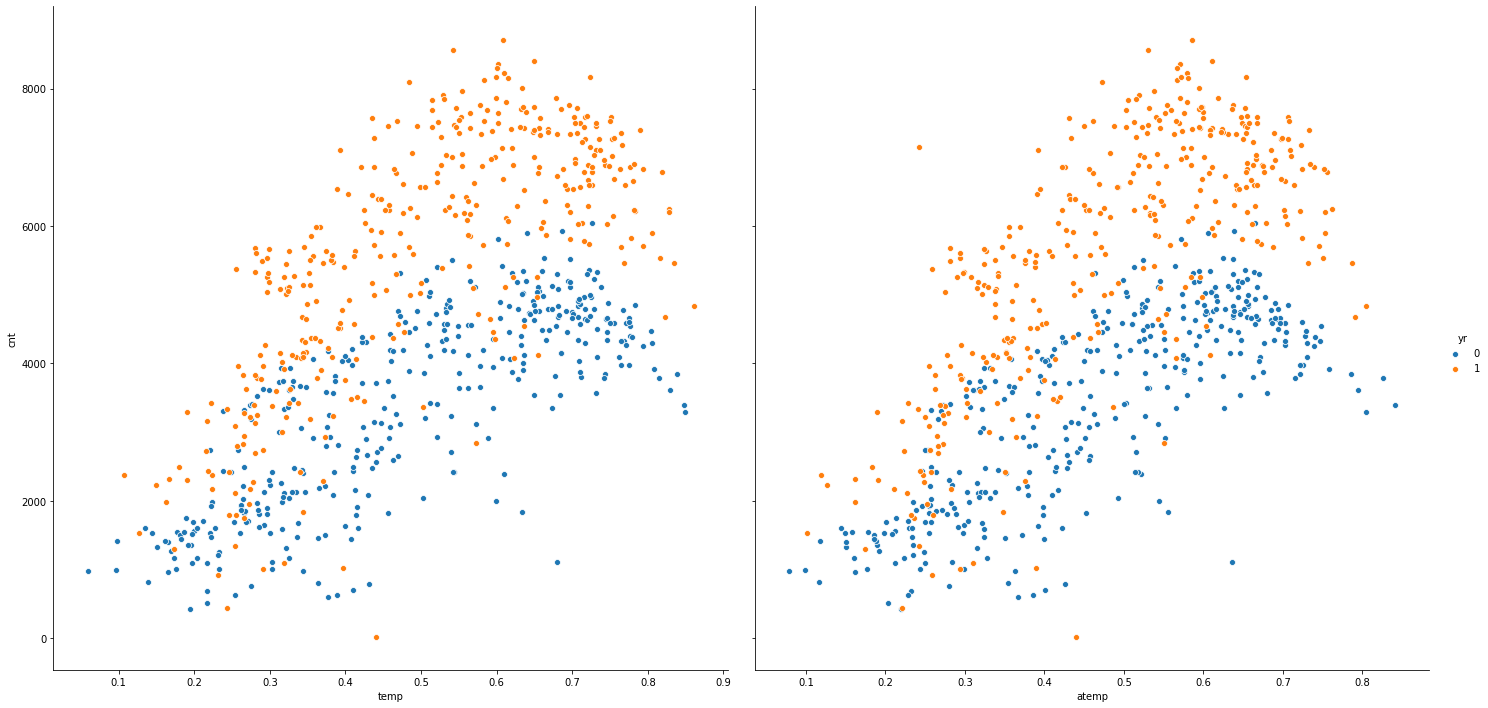
There are two temperature features and I wanted to find out which one is a better predictor. The first feature, temp, is the normalized literal temperature. The second feature is atemp, this is the temperature that represents what it “feels like” outside. I ran the gradient descent on each of them with ‘cnt’ being the target variable. I wanted to see which one would converge with a smaller error. The gradient descent was ran on the daily data set rather than the hourly data set on which the regression was run. I ran a few models playing around with the step size and number of iterations, these are the values around which we “started going up in the parabola.”

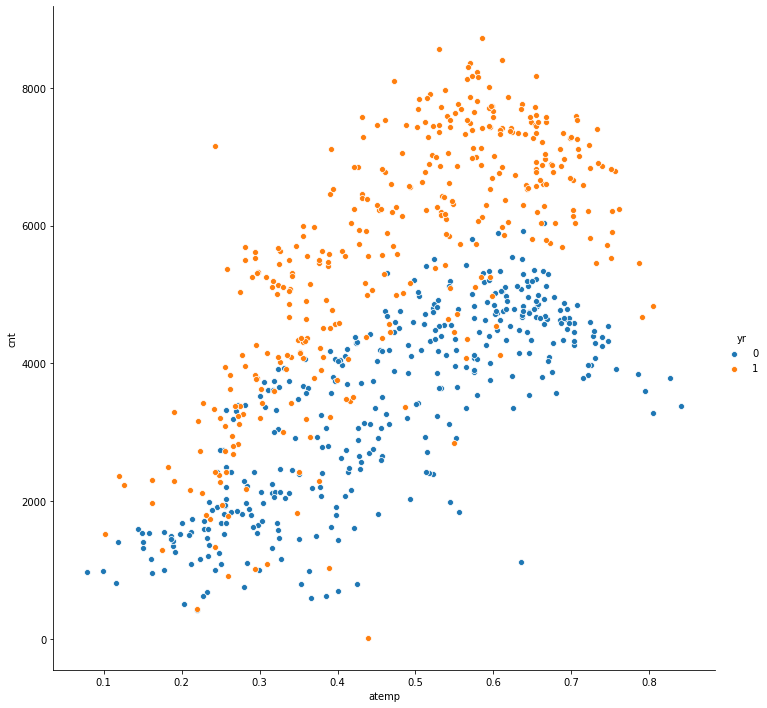
The optimal weight value for temp which yielded an error somewhere around **13776721.374.**

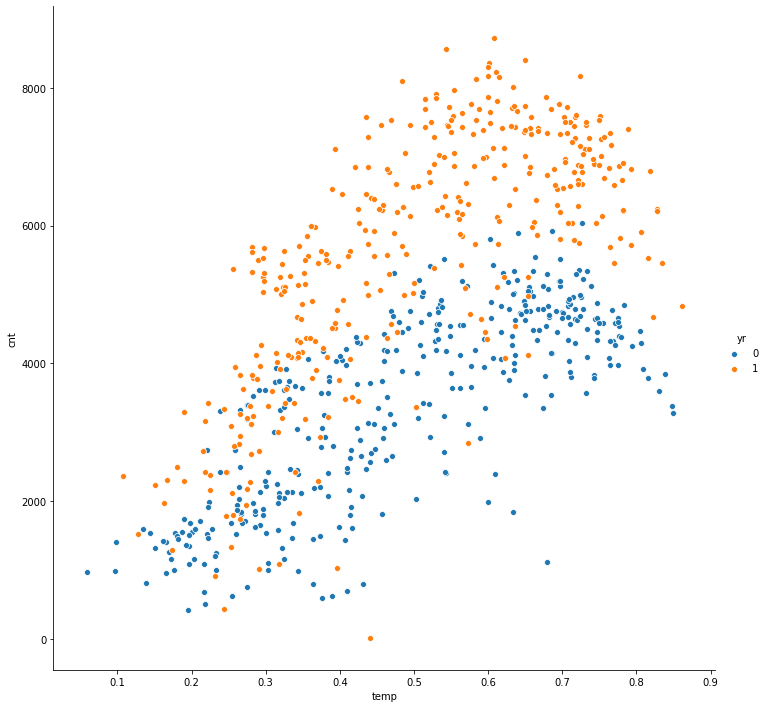
The optimal weight value for temp which yielded an error somewhere around **13786354.901**.

A difference of **9633.537**.

Temp converged with a smaller error than atemp, suggesting that temp is the better predictor.







For both visualizations, notice how the data for 2012 almost exclusively exhibits higher values and is more scattered than the data from 2011.

Code for the visualizations

Cnt by temp:

sns.pairplot(df, x\_vars ='temp' , y\_vars = 'cnt', hue = 'yr', height = 10)

cnt by atemp

sns.pairplot(df, x\_vars ='atemp' , y\_vars = 'cnt', hue = 'yr', height = 10)

both side by side

sns.pairplot(df, x\_vars =('temp' , atemp’) , y\_vars = 'cnt', hue = 'yr', height = 10)

# Conclusion and Thoughts about future work

Since we had data from the begining years of this program’s existence, it was difficult to create an accurate predictive model. This is because we have an unmeasurable confounding variable in the mix that decreases the predictive power of the features as time goes along. The confounding variable is how much of DC population knows about bike sharing and has entertained the thought of using it in their head. In 2012 this variable had a higher value than it did in 2011, and the data for 2012 is considerably more scattered.

Regardless of that, it has been shown that weather and time do have some power in predicting how many bikes will be rented. With the weather, it has been shown that people do not like to bike when it is hot and humid. With time, it has been shown that people like to bike primarily in the day time. It has been shown that atemp or “what it feels like outside” is likely less predictive of whether people ride bikes than the objective temperature.

I would like to redo this analysis for data in recent years when the bike sharing programs are something that everyone has heard of. I would also like to create a graph that tells me which hours of the day are the busiest and then create a feature weighing the busiest times the most.