Seoul Bike Sharing Demand

Supervised Machine Learning: Regression - Course Project

Caio Mescouto Terra de Souza

June 8, 2021

Introduction and Main Objective

Nowadays rental bikes are a reality in the many urban cities in the world. One of the biggest concerns is maintaining a reliable supply of bicycles to meet public demand regardless of time and weather. However, the demand is not constant. It has, at least, seasonal fluctuation depending where the city is in addition to hourly and weather fluctuations. The maintenance schedule and the logistics of distribution of bicycles between locations are very important due to the fluctuation on demand stated before.

Regarding the main concern presented, the data set I worked on contains the count of public bikes rented at each hour in the Seoul Bike sharing System, besides weather and holiday information. The main objective of this analysis is to build a model highly interpretative where it will be possible to isolate the attributes that are mostly responsible for the demand fluctuation. Even if the goal is interpretation, we would like to avoid underfitting by simplicity and, of course, overfitting by high complexity. I believe that a highly interpretive model is more actionable because it leads to more straight actions that can change certain variables or use this variable in favor of something. In this specific case, a highly interpretative model can lead to improvements on maintenance schedule, for example.

Data Set Description

As presented before, the data set contains the count of public bikes rented at each hour in the Seoul Bike sharing System [1]. It has 14 attributes and 8760 observations. The data set contains one year of data from december, 2017 to november, 2018. The table below summarises the attributes (Table 1).

Exploratory Data Analysis (EDA)

Considering the main objective, the data exploration had 5 steps. First, data cleaning; second, to split the data into train and test data; third, to understand the distribution of the numerical variables; fourth, to analyse the correlation between than and fifth, to understand how the number of rented bikes varies through season and holiday.

Column	Dtype	Unit
Date	Datetime	year-month-day
Hour	int64	hour
Temperature	float64	Celsius
Humidity	int64	%
Windspeed	float64	m/s
Visibility	int64	10m
Dew point temperature	float64	Celsius
Solar radiation	float64	MJ/m2
Rainfall	float64	mm
Snowfall	float64	cm
Seasons	object	Winter, Spring, Summer, Autumn
Holiday	object	Holiday, No Holiday
Functional day	object	Non Functional Hours, Functional Hours
Rented Bike count	int64	Count of bikes rented at each hour

Table 1: Data Set Attributes

Before presenting the steps, some assumptions and limitations have already to be set. The data set is one year long, so, it's not possible to analyze properly if the demand is increasing or not because the seasonal effect will affect this analysis. Finally, it is assumed the demand for rented bikes is always fully met by the supply, meaning that we don't have a cut off by scarcity of bikes in any observation. This assumption is needed otherwise the model is biased.

Data Cleaning

The data set is almost clean, no null, no outliers, no mismatching. The only point is that the variable *Functional day* flags whether the service is operational or not and, as we are interested in the demand, we can drop all no operational observations (295 entries) that represent only 3% of the data set.

Data Stratified Sampling

To avoid hacking the model it is important to split the data between train and test data as early as possible. Just after the first view and the data cleaning, the data was split using a stratified split. The stratification followed the distribution of the target variable (*Rented Bike Count*) whitin 5 bins and the test set was set in 20% of the data. Below the *Scikit-Learn* implementation applied.

```
StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=0)
```

Numerical Variables

This subsection covers two steps of the EDA plan (third and forth). First the histograms of all numerical variables were plotted and two problems that can interfere in modeling were

spotted. First, some variables are far away from normal distributions, second the scale is also different between them. The scaling was tackled during the data preparation process that precedes the modeling process. The distribution of the target value is presented below as well as the data treatment (Figure 1). After analyzing the skewness of all numerical variables (Table 2) A log transformation was applied whenever skewness was greater than 0.4.

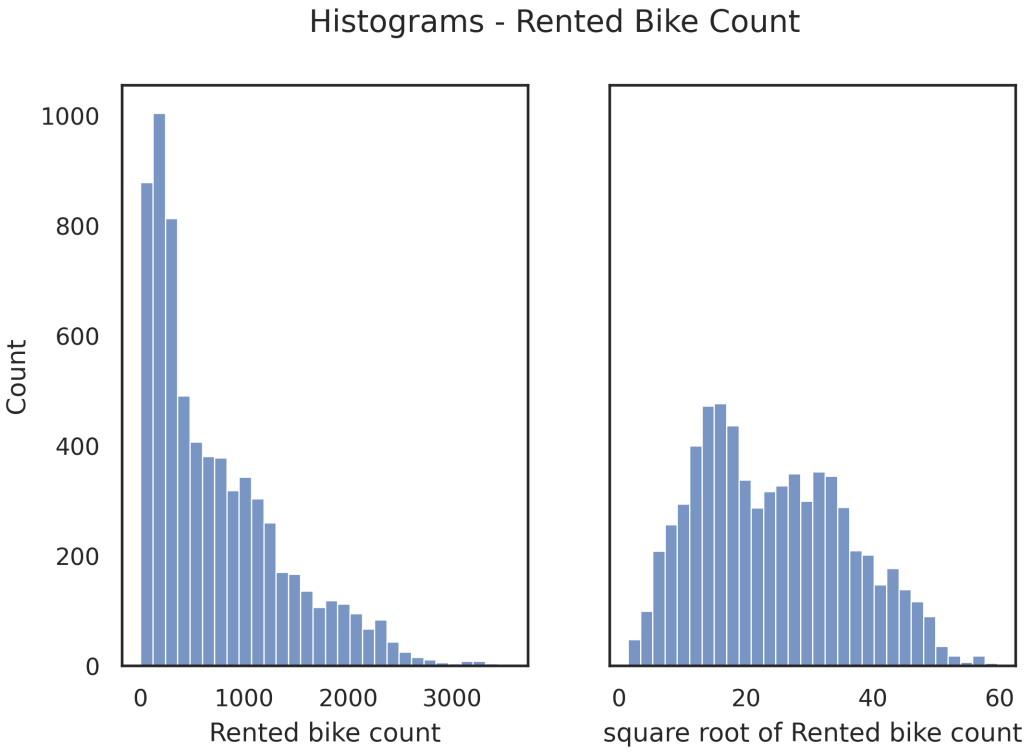

Figure 1: Target Histograms

After those transformations, the correlation between all variables (including the transformations) was carried out. In general was observed a improvement in correlation comparing the orignal variables and the transformations.

Variables that are highly correlated with each other should be avoided due to collinearity issues. As we can observe in the heatmap below (Figure 2), all variable transformations have highly correlation with the original one, as the correlation with the target was improved through the transformations, we dropped all originals in favor of the transformation. Finally, *Dew point temperature* has high correlation with *Temperature*, as the last one is more correlated with the target, it will be keeped and the former one will be dropped.

Categorical Variables

The last phase of the EDA is to analyze the relationship between the target value and the categorical variables. According to the violin plot (Figure 3) below we can easily observe that *Winter* is the season when the bikes are less rented and during this season, those who

Column	Skewness
Hour	−0.0025
Temperature	−0.1775
Humidity	0.0704
Windspeed	0.9126
Visibility	−0.6964
Dew point temperature	−0.3394
Solar radiation	1.5026
Rainfall	14.2689
Snowfall	7.8840

Table 2: Data Skewness

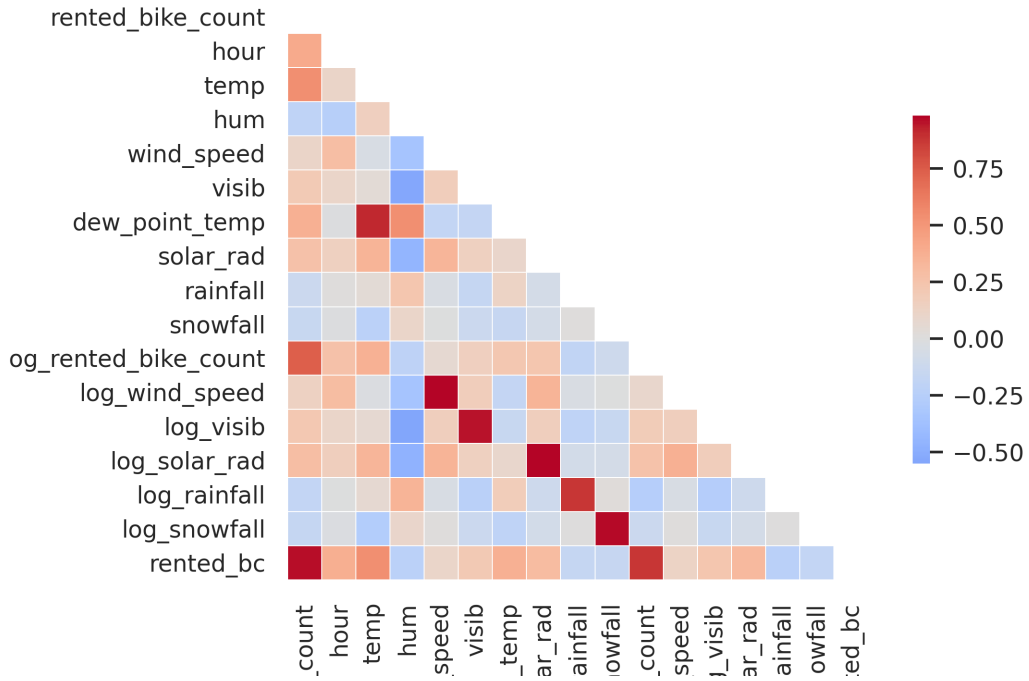

Figure 2: Heatmap of correlations

ride a bike do it more as means of transportation, because *No Holiday* has higher demand than *Holiday*. During the *Spring* the demand, in general, is higher than during *Winter*, but the same pattern is observed (*No Holiday*>*Holiday*). *Autumn* and *Summer* have the average greater than *Winter* and *Spring* and are quite similar between them both, with no strong pattern between *Holiday* and *No Holiday*. These patterns are expected due to the difference of temperature and rainfall/snowfall. But for me it was interesting to observe these differences between *Holiday/No Holiday*.

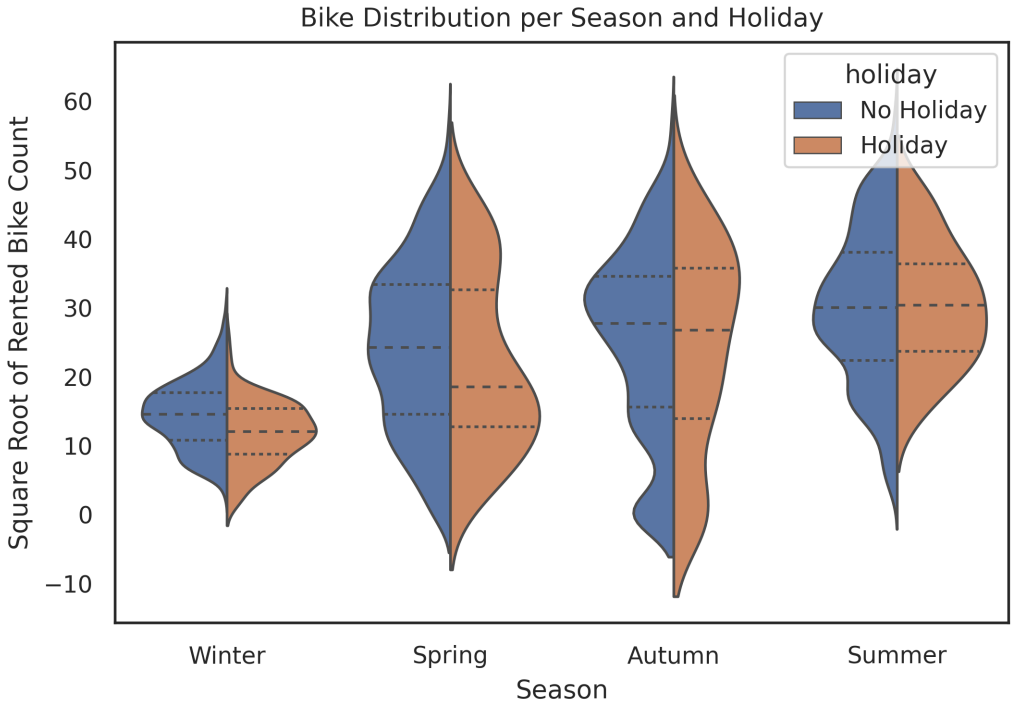

Figure 3: Bike Distribution per Season and Holiday

Data preparation and Feature Engineering

Some of the features engineering applied were already covered in the previous section. This section presents only the step by step used to prepare the data for modeling.

1. Split the data into train and test data set;
2. Filter the data only where *Functional day* is True;
3. Drop *Functional day, Date, Dew point temperature*;
4. Data log transformation in *Windspeed, Solar radiation, visibility, rainfall, snowfall*;
5. Divide data set into Features and Target;
6. Data Square root transformation in Target value;
7. Scale numerical variables with *Scikit-Learn*
   `StandardScale()` ;
8. Encode categorical variables with *Scikit-Learn*
   `OneHotEncoder(drop='first')` .

Model Selection

As stated in the first section, our main objective is to set up a model as simple as possible that fits the data, but where we can understand how the target is affected by the features. This model should avoid both under and overfit and have as few variables as possible. This problem is complicated because the number of variables is high related to under and overfitting. Therefore, first the R^2 was chosen as a parameter to compare models, as it is a measure of how well the model fits the data (or explains the phenomenon). The last concern is to ensure generalization (which is an overfitting concern as well), so a cross-validation model for training models should also be adopted. the Below is presented the step by step of this selection and further explanations.

0.1 Cross-Validation

All simulation during the model selection used the same set up for cross-validation:

```
GridSearchCV(<model>, n_jobs=-1, param_grid=<model parameters>,
scoring='r2', return_train_score=True)
```

0.2 Models

Four models were shortlisted as follow:

```
LinearRegression()
Lasso(random_state=0, max_iter=10000)
ElasticNet(random_state=0, max_iter=10000)
OrthogonalMatchingPursuit()
```

The most simple is the *Linear Regression* and it is likely to underfit the data. To avoid underfitting the data we first used this model to set the optimum degree for the polynomial features. As high polynomial degrees, on the other hand, tend to overfit the data, the cross validation was applied to avoid the other extreme as well. The *Polynomial Features* was optimized as follow. It's also import to highlight that small degree is consistent with more interpretability.

```
PolynomialFeatures(degree=3, include_bias=False)
```

After the optimization three other models were selected to improve the fitness through regularization. The *Lasso* regression throughout ℓ_1 penalization tends to force some coefficients to zero and this is more prominent with ℓ_1 penalization than ℓ_2 penalization (*Ridge*). Despite the computational superiority of the *Ridge regression*, the data set is relatively small and we are more interested in interpretability, so we picked *Lasso*. However, as we are also interested in a goodfit, we also set up a *ElasticNet* that combines both penalties (ℓ_1 *and* ℓ_2) and we can weight these penalties. The last model is the *Orthogonal Matching Pursuit (OMP)* implements the OMP algorithm for approximating the fit of a linear model with constraints imposed on the number of non-zero coefficients (ie. the ℓ_0 pseudo-norm) [2]. As we are interested in the fewest possible Features and this model is highly efficient in computational terms, it is a good candidate.

0.3 Cross-validation results

Best Estimator	R^2
LinearRegression(normalize=True)	0.7943
Lasso(alpha=0.0060, max_iter=10000, random_state=0)	0.8153
ElasticNet(alpha=0.0060, l1_ratio=1.0, max_iter=10000, random_state=0)	0.8153
OrthogonalMatchingPursuit(n_nonzero_coefs=155)	0.8144

Table 3: Cross-Validation Results

Following this result it is possible to infer that *Lasso* fits better than *Ridge* in this data set, as the best estimator with *ElasticNet* is in reality the same of *Lasso* ($\ell_1 ratio = 1$). In brief all estimators had achieved good scores. However, as we are interested in interpretability we dropped the linear regression and will further investigate *Lasso* and *OrthogonalMatchingPursuit*.

Model Interpretation and Key Findings

Two simple ways to start analysing complexity is to understand the weights and the number of non-zero coefficients. In addition to the R^2, Lasso is a also more simple model than OrthogonalMatchingPursuit (Table 4).

Model	**Coefficients** $= 0$	$\sum$ **|coefficients|**
Lasso	168	28	126.0955
OrthogonalMatchingPursuit	168	13	154.2423

Table 4: Models Coefficients

Although these measures of complexity, interpretation means to understand which variables are more responsible and in which direction they affect the target value. Below (Table 5) are presented the 10 more important variables with their respective coefficients for each model and the key insights. For sanity it's not repeated log all the time but *Rainfall and Solar Radiation* are in fact, log transformed.

Key Findings

- First of all, It is interesting that all variables are the same, with differing intensities but Lasso always has smaller coefficients than OrthogonalMatchingPursuit, meaning that Lasso is probability more regularezed than OrthogonalMatchingPursuit;
- Almost half of the most important variables has degree equal 3, meaning that the polynomial transformation played a important role in fitting the model;
- Only five of the eleven variables are listed in the top 10 (*Hour, Solar Radiation, Precipitation, Temperature, Season*);

Lasso	**Coef.**	**OrthogonalMatchingPursuit**	**Coef.**
$hour^2 \times solarRad$	10.5964	$rainfall$	-12.4247
$hour \times solarRad$	-10.3316	$hour^2 \times solarRad$	11.1473
$rainfall$	-10.2103	$hour \times solarRad$	-10.9250
$hour^2$	8.3205	$hour^2$	8.8413
$temperature$	6.8650	$temperature$	7.3074
$winter$	-6.2208	$winter$	-6.3295
$solarRad^2$	-3.8585	$solarRad^2 \times rainfall$	4.7297
$hour^3$	-3.6222	$solarRad^2$	-4.1108
$spring$	-3.4217	$hour^3$	-3.7492
$solarRad^2 \times rainfall$	3.3945	$spring$	-3.6187

Table 5: Main Features

- *Rainfall* has a negative effect on the target, but combined with the power of *Solar Radiation*, this effect is positive. As the power itself has a negative effect, Usually days with a little bit of rain but sunny are normal in the summer and the temperature is high.
- *Hour* has different effects on the target, below it is a plot of the mean demand per hour that can help understand this erratic behavior (Figure 4).

Test set

The last phase before choosing a model and presenting the conclusion is that both models were tested against the test data set and the result is presented below (Table 6).

Model	R^2
Lasso	0.8156
OrthogonalMatchingPursuit	0.8118

Table 6: Models performance

Model selection

Both models had performed as good as during the cross-validation, meaning that under and overfitting were both avoided. Finally, The *Lasso* regression is suggested to be adopted. The performance is better and interpretation-wise there is no difference in terms of top 10 variables, but has small coefficients and more zero-coefficients.

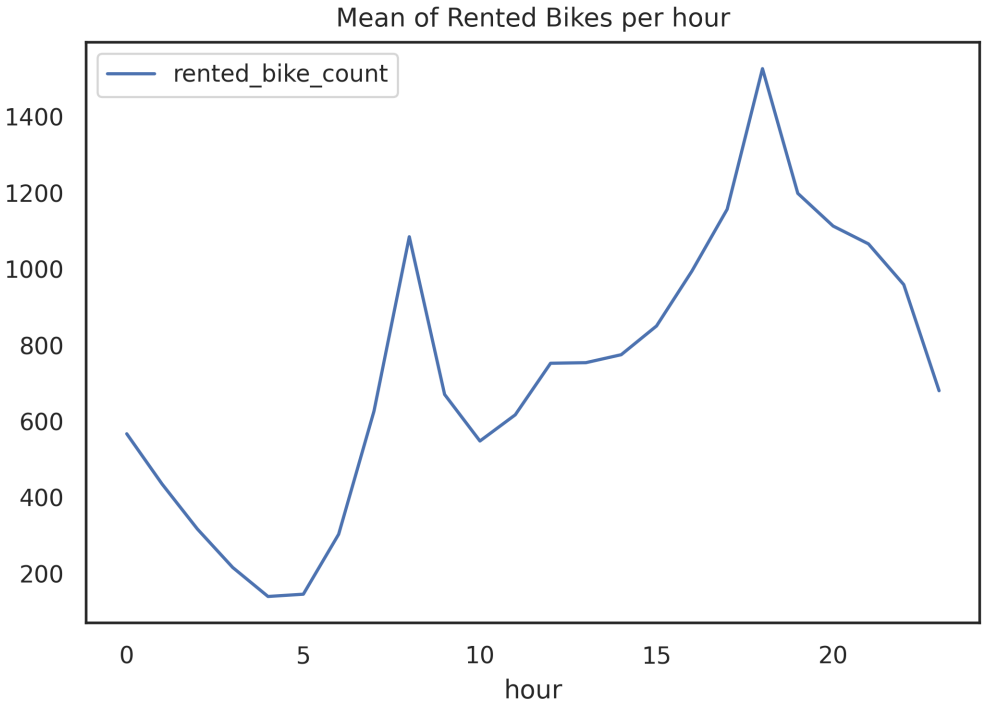

Figure 4: Mean of Rented Bikes per hour

Next steps and Improvements

The main flaw of the model is the granularity of the data. It is not a microdata that has information like localization of the bike or the time spent in the ride. This information would be used to improve the model, for example, connecting the flux of bikes during the period of the day or, understanding if some bikes spots are overloaded. These data should have granularity about weather as well, for example sparsity of the rain or solar radiation in different locations.

References

[1] Seoul Bike Sharing Demand Data Set,
`https://archive.ics.uci.edu/ml/datasets/Seoul+Bike+Sharing+Demand`

[2] 1.1.9. Orthogonal Matching Pursuit (OMP),
`https://scikit-learn.org/stable/modules/linear_model.html#omp`