Bike Sharing Demand Prediction

Chetan Jadhav,
Meghana Ranganath,
Pooja Parsana,
Robin Rego
Data Science Trainees,
Alma Better, Bangalore

Abstract:

Seoul city provide rental bike to their people for certain amount for certain period of time. But, the demand for bikes fluctuates with different conditions and so they aren't able to keep up with it. Our model aims at predicting Rented Bike demand at any time, considering all conditions which will help them to manage the flow of bikes.

We will examine which machine leaning algorithms suits best with data along with selection of features which have major impact in rented bike demand.

Keywords: Data Science, Machine Learning

1. Problem Statement

Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time. Eventually, providing the city with a stable supply of rental bikes becomes a major concern. The crucial part is the prediction of bike count required at each hour for the stable supply of rental bikes.

The main objective is to make a predictive model, which could help them in predicting the bike demand proactively. This will help them in stable supply of bike wherever needed.

The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.

Attribute Information:

- Date - year-month-day
- Rented Bike count - Count of bikes rented at each hour
- Hour - Hour of he day
- Temperature-Temperature in Celsius
- Humidity - %
- Windspeed - m/s
- Visibility - 10m
- Dew point temperature - Celsius
- Solar radiation - MJ/m2
- Rainfall - mm
- Snowfall - cm
- Seasons - Winter, Spring, Summer, Autumn
- Holiday - Holiday/No holiday
- Functional Day - NoFunc (Non Functional Hours), Fun(Functional hours)

2. Introduction

Due to global warming, continuous pollution and depletion of sources of energy Main countries have been focused of using renewable energy which doesn't harm environment and can be reused as well. South Korea is one the country which has adapted to it and their mostly used service is rented bike in Seoul. But in order to avoid any difficulties such as waiting time it is necessary to have an estimate of future demand.

Our goal here is to build model that can predict bike sharing demand considering all the factors which have their effects.

3. Major Factors Affecting Bike Demand

A) Rainfall :

People tend to use rented bike quite frequently due to the fact that they can be easily rented from any place and can be dropped off any other place, cheap enough to rent daily, but conditions like Rainfall affects its renting count a lot. People don't rent bike during rainy season. So we can say rainfall is negatively co-related with rented bike count.

B) Snowfall :

Similarly as rainfall, snowfall negatively affects rented bike count as its hard to drive with snowy road.

C) Visibility :

At times when one can't see properly, it's natural for them to avoid driving, and this is what which affects the rented bike count. Although in Seoul, case of these are quite low.

D) Temperature :

It is seen that, people then to avoid renting bikes at low temperature. Seoul is place with an average temperature of 27 to 32 degree Celsius. So, when temperature becomes warm people tend to enjoy it which has an effect in renting bikes as well.

E) Working Day or Not :

Compared to an Off day, people rent bikes more on a working day. Reason behind this is being the same i.e. they can be easily rented from any place and can be dropped off any other place, cheap enough to rent daily.

F) Traffic :

Even though this isn't mentioned in data, traffic also supports renting bike count indirectly. If traffic is high or large people visiting nearby walk or rent a bike for purpose.

4. Steps Involved

A) Exploratory Data Analysis :

This analysis is important in order to gain useful insights within data and also wrt dependent variable. In EDA variables are compared using plots and meaningful insights are drawn. It gives us better idea of which feature behaves in which manner compared to target variable.

B) Data Cleaning :

Our Data contains some null values which might tend to disturb our accuracy hence we dropped them at the beginning of our project to get better result.

C) Encoding of Categorical Columns :

We used one Hot Coding to produce binary integers of 0 and 1 to encode our categorical features because categorical features that are in string format cannot be understood by machine and needs to be encoded.

D) Feature Scaling :

Feature scaling is essential for machine learning algorithms that calculate distances between data. If not scale, the feature with a higher value range starts dominating when calculating distances.

E) Fitting different Model :

At first we tried with basic **linear regression**, but soon realized we will need much more complex model and so we then used **Decision tree Regressor.** Model is then boosted and results are compared.

F) SHAP Values :

We have applied SHAP value plots on the Decision Tree Regressor model to determine the features that were most important while modeling and those who didn't supported as well.

5. Algorithms

A) Regression :

Regression searches for relationships among variables. For example, you can observe several employees of some company and try to understand how their salaries depend on the **features**, such as experience, level of education, role, city they work in, and so on.

This is a regression problem where data related to each employee represent one **observation**. The presumption is that the experience, education, role, and city are the independent features, while the salary depends on them.

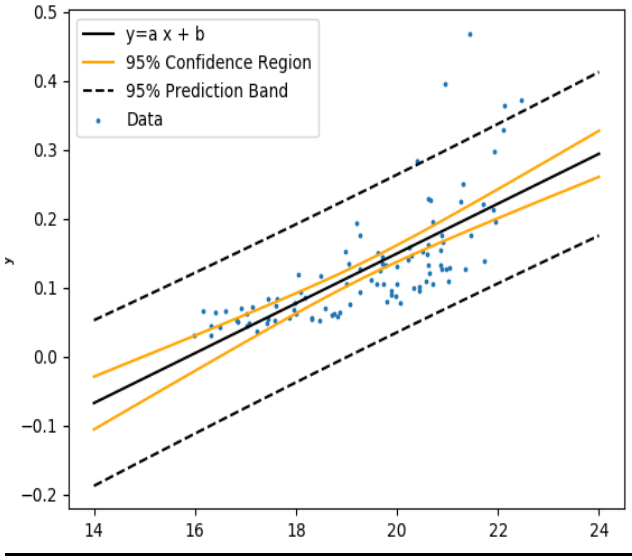
Similarly, you can try to establish a mathematical dependence of the prices of houses on their areas, numbers of bedrooms, distances to the city center, and so on.

Generally, in regression analysis, you usually consider some phenomenon of interest and have a number of observations. Each observation has two or more features. Following the assumption that (at least) one of the features depends on the others, you try to establish a relation among them.

In other words, you need to find a function that maps some features or variables to others sufficiently well.

The dependent features are called the **dependent variables**, **outputs**, or **responses**.

The independent features are called the **independent variables**, **inputs**, or **predictors**.



Regression problems usually have one continuous and un-bounded dependent variable. The inputs, however, can be continuous, discrete, or even categorical data such as gender, nationality, brand, and so on.

It is a common practice to denote the outputs with y and inputs with x. If there are two or more independent variables, they can be represented as the vector $\mathbf{x} = (x_1, \ldots, x_r)$, where r is the number of inputs.

B) Decision Tree Regressor :

Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs and utility.

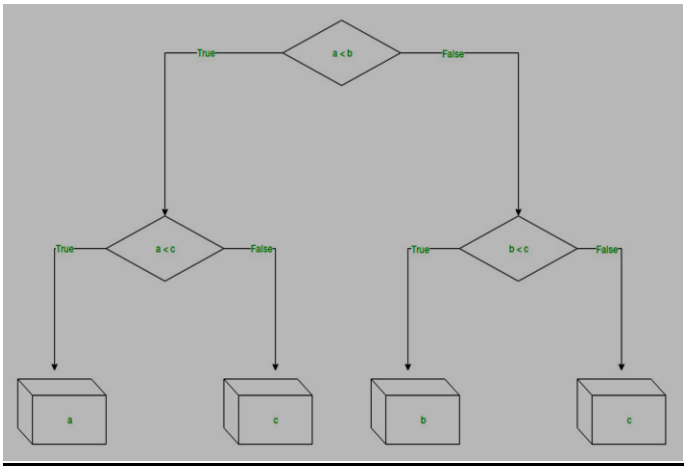
Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.

The branches/edges represent the result of the node and the nodes have either:

1. Conditions [Decision Nodes]

2. Result [End Nodes]

The branches/edges represent the truth/falsity of the statement and takes makes a decision based on that in the example below which shows a decision tree that evaluates the smallest of three numbers:



Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous out-put. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.

C) Gradient Boosting :

The term gradient boosting consists of two sub-terms, gradient and boosting. We already know that gradient boosting is a

boosting technique. Let us see how the term 'gradient' is related here.

Gradient boosting re-defines boosting as a numerical optimization problem where the objective is to minimize the loss function of the model by adding weak learners using gradient descent. Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. As gradient boosting is based on minimizing a loss function, different types of loss functions can be used resulting in a flexible technique that can be applied to regression, multi-class classification, etc.

Intuitively, gradient boosting is a stage-wise additive model that generates learners during the learning process (i.e., trees are added one at a time, and existing trees in the model are not changed). The contribution of the weak learner to the ensemble is based on the gradient descent optimization process. The calculated contribution of each tree is based on minimizing the overall error of the strong learner.

Gradient boosting does not modify the sample distribution as weak learners train on the remaining residual errors of a strong learner (i.e., pseudo-residuals). By training on the residuals of the model, this is an alternative means to give more importance to misclassified observations. Intuitively, new weak learners are being added to concentrate on the areas where the existing learners are performing poorly. The contribution of each weak learner to the final prediction is based on a gradient optimization process to minimize the overall error of the strong learner.

6. Model Performance

1. R-squared (R2) :

This is the proportion of variation in the outcome that is explained by the predictor variables. In multiple regression models, R2 corresponds to the squared correlation between the observed outcome values and the predicted values by the model. The Higher the R-Squared, the better the model.

2. Root Mean Squared Error (RMSE) :

This measures the average error performed by the model in predicting the outcome for an observation. Mathematically, the RMSE is the square root of the *mean squared error (MSE)*, which is the average squared difference between the observed actual outcome values and the values predicted by the model. So, MSE = mean((observed - predicted)^2) and RMSE = Sqrt(MSE). The lower the RMSE, the better the model.

3. Residual Standard Error (RSE) :

Also known as the *model sigma*, is a variant of the RMSE adjusted for the number of predictors in the model. The lower the RSE, the better the model. In practice, the difference between RMSE and RSE is very small, particularly for large multivariate data.

4. Mean Absolute Error (MAE) :

Like the RMSE, the MAE measures the prediction error. Mathematically, it is the average absolute difference between

observed and predicted outcomes, MAE = mean(abs(observed - predicted)). MAE is less sensitive to outliers compared to RMSE.

7. Hyper Parameter Tuning

Hyper-parameters are those sets of information that are used to control our parameters in order to get good results. We used Grid Search CV for hyper parameter tuning.

Grid Search CV :

It is the process of performing hyper parameter tuning in order to determine the optimal values for a given model. As mentioned above, the performance of a model significantly depends on the value of hyper parameters. Note that there is no way to know in advance the best values for hyper parameters so ideally, we need to try all possible values to know the optimal values. Doing this manually could take a considerable amount of time and resources and thus we use Grid Search CV to automate the tuning of hyper parameters.

Grid Search CV is a function that comes in Scikit-learn's(or SK-learn) model_selection package. So an important point here to note is that we need to have Scikit-learn library installed on the computer. This function helps to loop through predefined hyper-parameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyper parameters

8. Conclusion:

In project, after trying combinations of features with linear regression the model underfit. It seemed obvious because data is spread too much. It didn't seem practical to fit a line.

- With Decision tree we reached at the model R squared value of 0.86. We only fitted with minimum number of leaf hyper-parameter. With default parameters it overfitted and reached R-squared at 1 with train dataset but 0.86 with test.
- The Feature_importance is almost the same in both the tree based models. Gradient boost fine-tunes with error of the prior trees this is why it gets better accuracies.

9. References

- https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/
- https://www.mygreatlearning.com/blog/gradient-boosting/#sh2
- http://www.sthda.com/english/articles/38-regression-model-validation/158-re-gressionmodel-accuracy-metrics-r-square-aic-bic-cp-and-more/
- https://www.mygreatlearning.com/blog/gridsearchcv/#sh1