Bike Rent Project

**Introduction**

Bike is a most common means of transport now a days. A school going children will love to ride bikes and youngsters go for racing . If you want to reach a destination as soon as possible individually or with your partner or friends every ones first point of choose is to go on bike.It doesn’t mean that you can take your own bikes every time out of station. It doesn’t even mean that everyone can afford a own bike.Here comes a picture to have a rental based bike for an individual to use as per their needs.So this project is to predict bike rental count on daily basis as per some important attributes.

We need a software to predict the future cases as how much will be the rental count of bikes based on season,time and date etc.Machine learning algorithms plays a vital role for right predictions as we all now.So,lets choose a best machine learning algorithm for prediction of bike rental count for future analysis.This project mostly diverges to linear regression so lets proceed with basic understanding of given data.

**Problem Statement :**

The objective of this Case is to Prediction of bike rental count on daily based on the environmental and seasonal settings.

**Data Set :**

So we have a very important data attributes from past experience where bike rental count is mostly dependent on. As from our problem statement we have to predict the rental count of bikes as per environmental basis which is a continuous variable. Where season ,time,year plays a vital role. So, in the data set we have those attributes in a required way.

In this data set we have only training data no test cases have been provided. So train data have these attributes:

R code:

> str(bike)'data.frame': 731 obs. of 16 variables:

$ instant : int 1 2 3 4 5 6 7 8 9 10 ...

$ dteday : Factor w/ 731 levels "2011-01-01","2011-01-02",..: 1 2 3 4 5 6 7 8 9 10 ...

$ season : int 1 1 1 1 1 1 1 1 1 1 ...

$ yr : int 0 0 0 0 0 0 0 0 0 0 ...

$ mnth : int 1 1 1 1 1 1 1 1 1 1 ...

$ holiday : int 0 0 0 0 0 0 0 0 0 0 ...

$ weekday : int 6 0 1 2 3 4 5 6 0 1 ...

$ workingday: int 0 0 1 1 1 1 1 0 0 1 ...

$ weathersit: int 2 2 1 1 1 1 2 2 1 1 ...

$ temp : num 0.344 0.363 0.196 0.2 0.227 ...

$ atemp : num 0.364 0.354 0.189 0.212 0.229 ...

$ hum : num 0.806 0.696 0.437 0.59 0.437 ...

$ windspeed : num 0.16 0.249 0.248 0.16 0.187 ...

$ casual : int 331 131 120 108 82 88 148 68 54 41 ...

$ registered: int 654 670 1229 1454 1518 1518 1362 891 768 1280 ...

$ cnt : int 985 801 1349 1562 1600 1606 1510 959 822 1321 ...

So we have 731 observations and 16 variables.

We have ‘dteday’ variable which represents the date.

We have most important variables where we have to concentrate on. Those variables are:

Season : (1:springer, 2:summer, 3:fall, 4:winter) which have only four values where 1 is spring season , 2 means summer season , 3 means Rainy season and 4 means winter season.

Year : Even this variable is also on 2 levels i.e.. 0 means 2011 and 1 means 2012. So by this we can understand that whole data is in between 2011 and 2012 years.

There are some extracted variables which explains whether it was a holiday, working day and so on.

If you could see there is a variable called ‘weather sit’ : which again in four levels as defined below:

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

Which is the most important aspect of this project target variable.As we are calculating based on seasonal settings which is explained by this variable.

Next comes ‘temp’ variable which is another most important variable for the data analysis.Which explains about 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)

And then,

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)

Wind speed: Normalized wind speed. The values are divided to 67 (max)

We are going to see more about clear view of variables in EDA. Till now the variables we observed are predictor variables.

And most finally we have count variable which explains count of total rental bikes including both casual and registered which is our **TARGET** variable.

**Exploratory Data Analysis :**

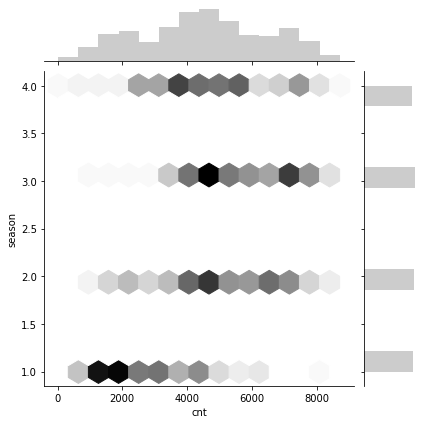
Here, this is essential part of any data scientist where he can clearly understand the data and think which algorithm will be useful for his predictions.

So we know that data is mostly explaining about some basic attributes like season , temperature, etc..First of all I have removed variable ‘instant’ which is of no use as it is just an index. Next comes with ‘dtedy’ variable as it explains date and day of the data but we already have other variables like ‘yr’ , ‘mnth’ ,’weekday’, which gives same information so even this variable doesn’t carry much information hence I removed this variable. Let me be clear, the data which we received is mostly pretty good where lot of information is clearly distributed as per the records they maintained which gives a clear picture of our data.

Next comes with season variable.Rather than my words lets see visually how data is distributed according to 4 seasons we have :

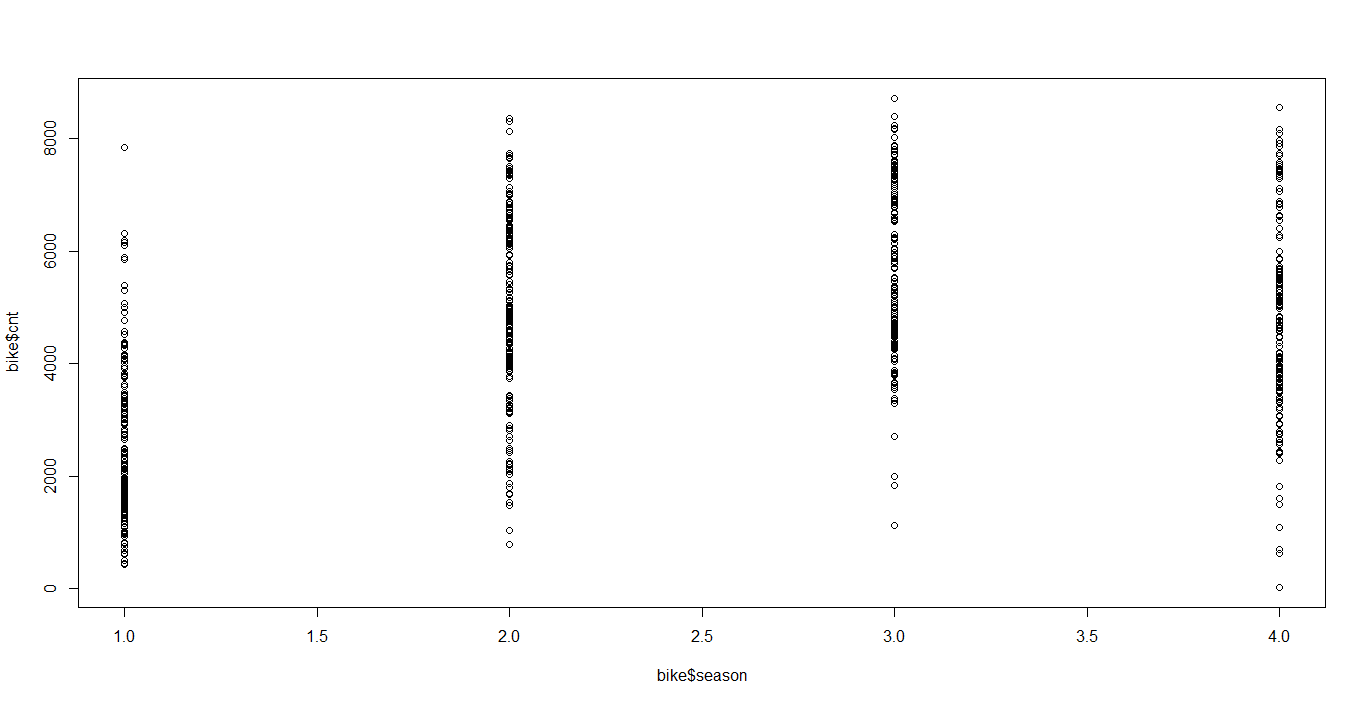
Here comes python seaborn library help and could see the data using jointplot command:

*sns.jointplot(x = 'cnt',y='season',data = bike,kind ='hex',color='grey')*



So we can understand data is distributed normally on every season as per count comparision where we have more number of counts on season 2 and season 4 as summer and winter which is obvious.

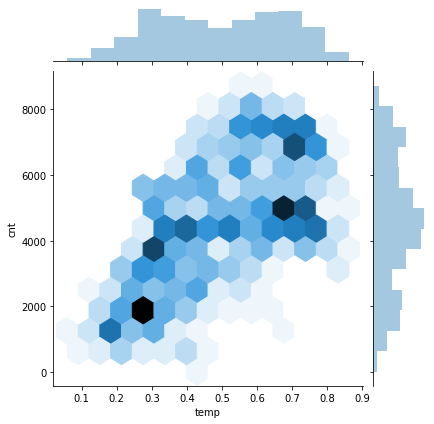
The comparison is mostly done on count variable as it is target variable.We can have a simple view from R :



We have got an idea how data is distributed based on seasonal settings as per problem statement.

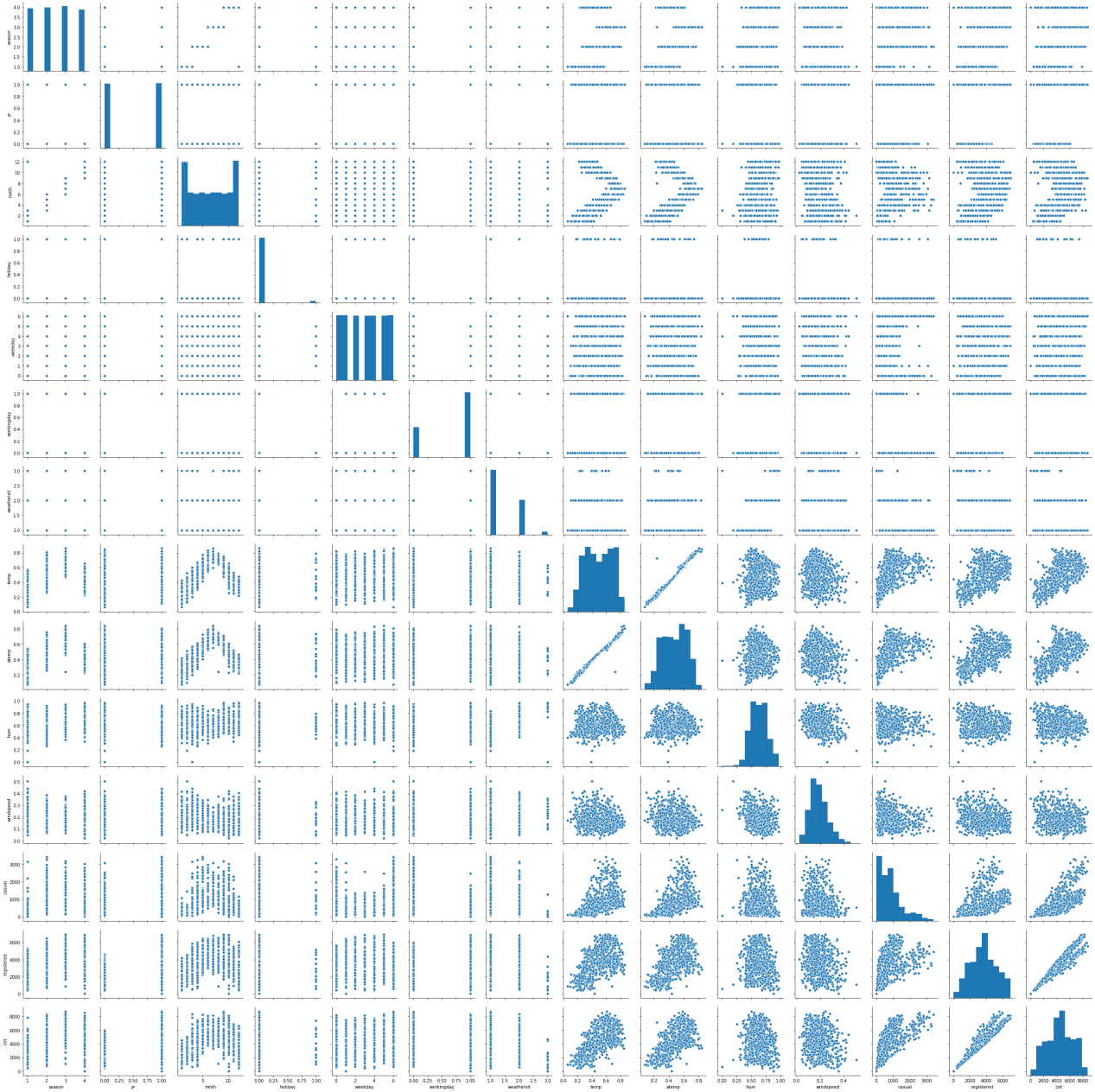
Next lets see how temperature is effecting the count of bike from python seaborn library by plotting a joint plot again.

*sns.jointplot(x = 'temp',y='cnt',kind='hex',data = bike)*



Which gives us a pretty good understanding as how temperature of the day variable is linearly dependent on cnt variable. That’s good.

Are you still having a doubt on how variables are distributed or lets try analyzing the data in pretty good manner by using a pair plot which gives a clear picture of the data given.



Isnt it great! We can clearly see how data is distributed on respective variables.

Now comes the next part of exploratory data analysis which is important that is outlier detection.

**OUTLIER ANALYSIS:**

Now its the time to check for outliers as we have dimension-ed the data .So is there any outliers in the data which might impact model performance or even model development . I have plotted the box plot and for all variables in python as shown below which confirms that there are outliers in the some variables.Lets see :



From above picture its clearly visible that there are some variables which have outliers but most of the variables don’t have outliers at all but the variables which have outliers are :

1. Holiday
2. Humidity
3. Windspeed
4. Casual.

All outliers have been removed from the data and now we have 655 observations after removing outliers.

**MULTICOLLINEARITY and CORRELATION ANALYSIS :**

So now after removing outliers its time to check whether there is any correlation between independent variables which results to multicollinearty and to have a out-view of how each variable is col-linear with target variable.

So by using R I have plotted eclipse view of correlation of all variables from eclipse library which gave analysis of correlation variables with each other.

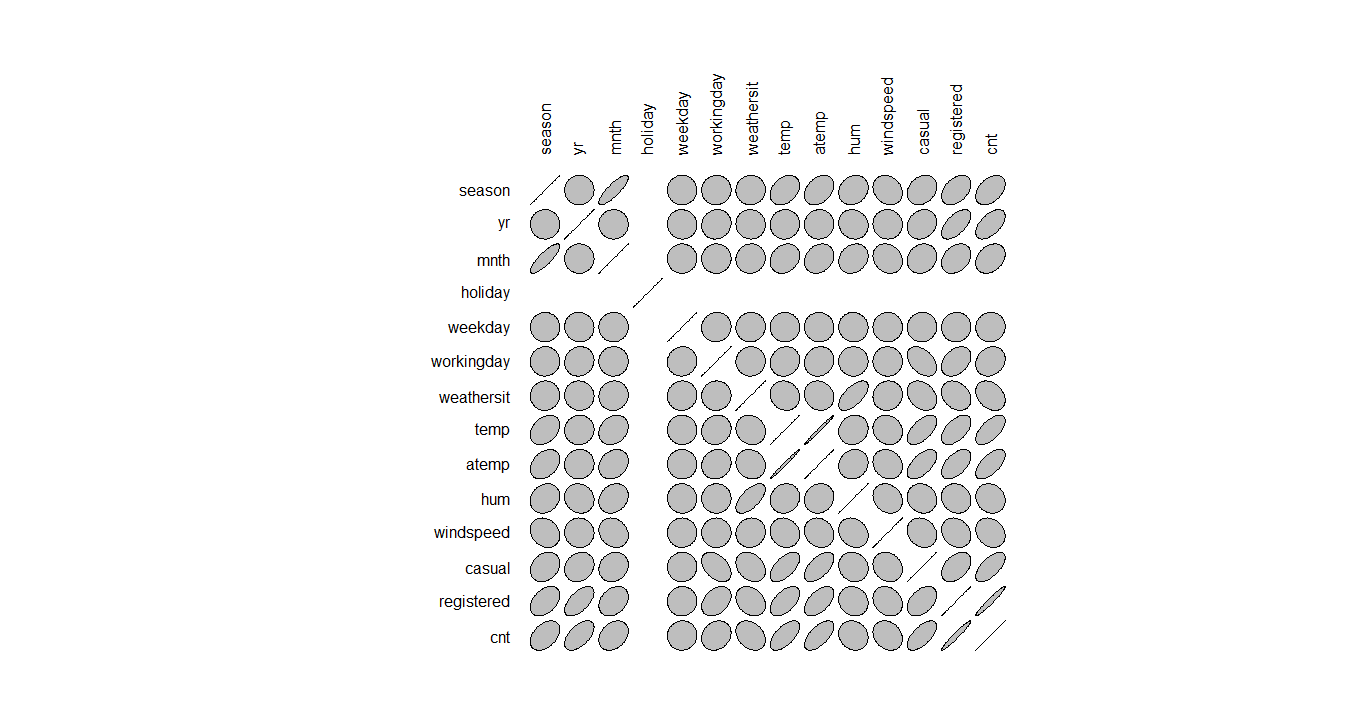
R code :

*library(ellipse)*

*ctab = cor(bike)*

*round(ctab,2)*

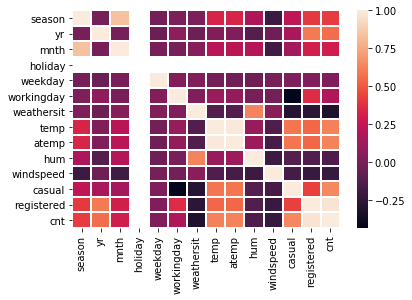
*plotcorr(ctab, mar = c(0.1, 0.1, 0.1, 0.1))*



In Python by using heatmap function we could see a similar view like this .

Python code :

*sns.heatmap(bike.corr(),lw=1)*



Isnt it a great view how data is related to each variable. I know what you are observing on both plots you could see holiday variable is no where related only because that whole variable is carrying only one value I.e. is ‘0’ after removing outliers. Even though holiday doesn’t matter much of bike rental count as we all know maximum in a year we will have 24 holidays. So I have removed ‘holiday’ variable from the data. We could see that ‘registered’ , ‘temp’ ,’atemp’, variables are carrying most information to explain target variable.In other words, they are correlated towards target variable.Which is good.

But have you observed ‘season’ variable is not highly correlated with target variable.Where as ,’yr’ variable is related to target variable pretty good.

**VIF factor :**

But before applying linear regression don’t you think that its better to check VIF factor after all these analysis which is essential part for a perfect fit of data.

R code :

> vifcor(bike[,-13], th = 0.9)1 variables from the 12 input variables have collinearity problem:

atemp

After excluding the collinear variables, the linear correlation coefficients ranges between:

min correlation ( temp ~ weekday ): -0.0001699624

max correlation ( mnth ~ season ): 0.8314401

---------- VIFs of the remained variables --------

Variables VIF

1 season 4.102798

2 yr 2.724922

3 mnth 3.355287

4 weekday 1.040958

5 workingday 2.992456

6 weathersit 1.910225

7 temp 2.461189

8 hum 1.931269

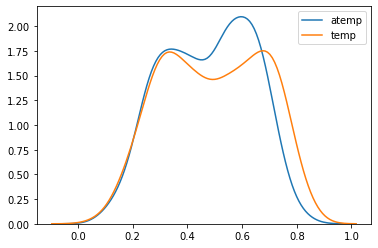
9 windspeed 1.224505

10 casual 3.493932

11 registered 5.959593

So, from above information it is obvious that temp and atemp have multicollinearity issue with target variable.Hence I removed ‘atemp’ variable . And all other variable values range below 10 hence no issues .

You can see how they are related to each other as shown below : where you can understand mulitcollinearity problem .



So the data is now dimension-ed in a proper way and its time to implement a model.

**MODEL SELECTION:**

So we have implemented two models so that to have a better understanding and not to loose any important information.

**MULTIPLE LINEAR REGRESSION :**

Linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables).Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a \*X + b

R code:

By using lm function I have implemented Linear regression on data.

Call:

lm(formula = cnt ~ ., data = bike)

Residuals:

Min 1Q Median 3Q Max

-1.816e-11 -1.858e-13 -2.590e-14 1.832e-13 1.753e-11

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -2.006e-12 3.730e-13 -5.378e+00 1.05e-07 \*\*\*

season -5.277e-13 9.069e-14 -5.819e+00 9.32e-09 \*\*\*

yr -1.429e-12 1.649e-13 -8.669e+00 < 2e-16 \*\*\*

mnth 9.485e-14 2.655e-14 3.572e+00 0.00038 \*\*\*

weekday -4.841e-14 2.607e-14 -1.857e+00 0.06376 .

workingday 5.881e-13 1.965e-13 2.993e+00 0.00287 \*\*

weathersit 9.235e-13 1.305e-13 7.075e+00 3.92e-12 \*\*\*

temp 9.674e-13 4.388e-13 2.205e+00 0.02784 \*

hum -6.008e-13 5.073e-13 -1.184e+00 0.23678

windspeed 2.244e-12 7.536e-13 2.978e+00 0.00301 \*\*

casual 1.000e+00 1.825e-16 5.479e+15 < 2e-16 \*\*\*

registered 1.000e+00 8.031e-17 1.245e+16 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

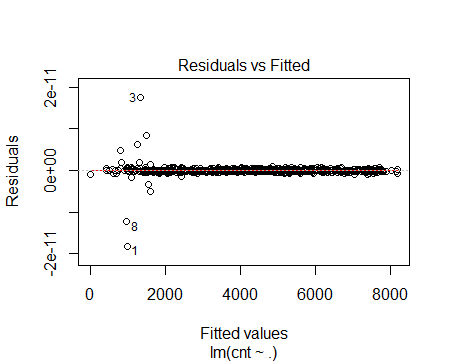
Residual standard error: 1.267e-12 on 643 degrees of freedom

Multiple R-squared: 1, Adjusted R-squared: 1

F-statistic: 1.274e+32 on 11 and 643 DF, p-value: < 2.2e-16

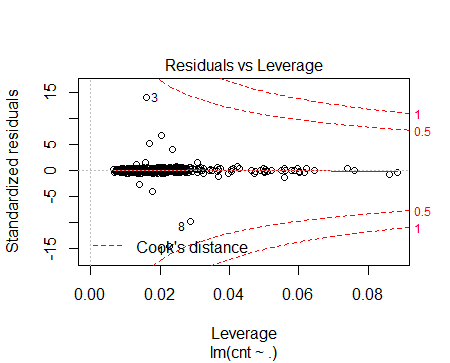
*Wonderful! We got R-squared value as 1*.R-squared is a statistical measure which explains how well data is fitted to the regression line. That means our model is a perfect fit as target variable is pretty well explained by the given set of variables.

Visuals :









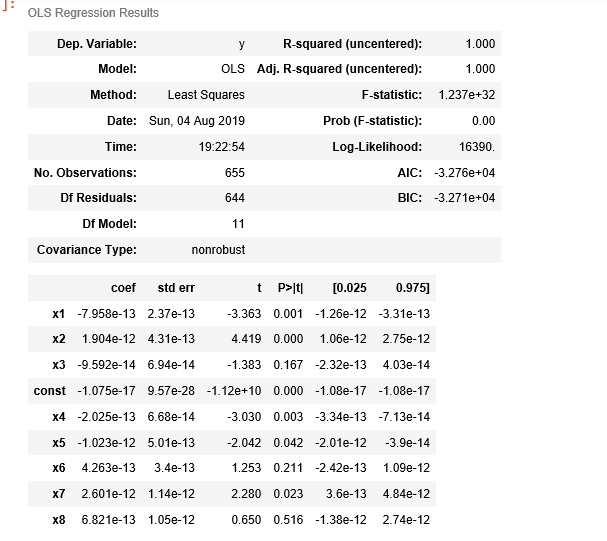
In Python:

As we know,by statsmodel library imported OLS least square method to develop linear regression model.

*import statsmodels.api as sm*

*Linearmodel = sm.OLS(Y,X).fit()*

*Linearmodel.summary()*



Wonderful! R-square is 1 and adjusted R-square is also 1. So without a clue we can easily consider this perfect fit model.

So now I have predicted the values by using ‘predict’ function above models on both R and Python.

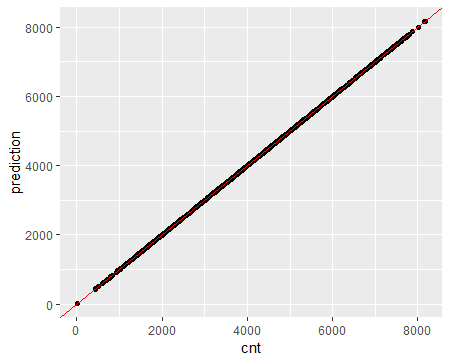
Python:

*predictions = Linearmodel.predict(X)*

*R:*

*prediction=predict(lmmodel,bike[,-12])*

In R, I have plotted linear regression model which was very good view of predicted values on the data by using ggplot function.



Is it looking so nice where all your values have perfectly fitted on given test data.

In python by using lm plot from seaborn library.



Even in python its pretty good fit model.

Its pretty much clear that linear regression model is perfect fit for prediction of bike rental count for new test cases.

As mentioned before just for comparison implemented decision tree model .Lets go!

**DECISION TREE:**

Decision tree is a rule. Each branch connects nodes with “and” and multiple branches are connected by “or”

R code:

In R we use ‘Rpart’ library for model evaluation and we select method as ANOVA as it is a regression analysis.

*library(rpart)*

*library(rpart.plot)*

*decisiontree = rpart(cnt~., data = bike, method = 'anova')*

After implementing decision tree model we can see the rules in R easily as shown below:

> decisiontreen= 655

node), split, n, deviance, yval

\* denotes terminal node

1) root 655 2250327000 4397.757

2) registered< 3905.5 392 625346400 3223.906

4) registered< 2294 141 59652420 1796.390

8) registered< 1377.5 50 7967350 1138.140 \*

9) registered>=1377.5 91 18116750 2158.066 \*

5) registered>=2294 251 116955600 4025.817

10) casual< 651.5 123 31012140 3582.748 \*

11) casual>=651.5 128 38594460 4451.578 \*

3) registered>=3905.5 263 279745500 6147.376

6) registered< 5411 159 67462910 5460.069

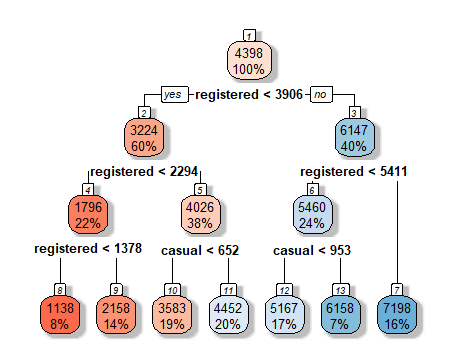
12) casual< 952.5 112 23101520 5167.250 \*

13) casual>=952.5 47 11873890 6157.851 \*

7) registered>=5411 104 22340470 7198.163 \*

I know everyone likes visuals even I like too.So,lets see decision tree by using rpart.plot library and function by shown below code:

*rpart.plot(decisiontree, box.palette="RdBu", shadow.col="gray", nn=TRUE)*

**

*In python.*

Decision tree is implemented by using scikit learn module and DecisionTreeRegressor function.

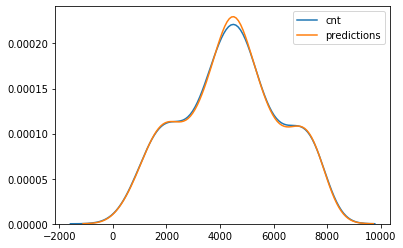
Python code :

*from sklearn.tree import DecisionTreeRegressor*

*fit\_DT = DecisionTreeRegressor(max\_depth=7).fit(X,Y)*

*So,its time to see in python how well the decision tree have fitted the values.*

*By using kdeplot I could see :*

**

*Which is a good mo*del I say.So lets go to the evaluation part and see which model is best and which model helps us to predict it in a right way.

**Error metrics:**

So, we have completed predicting values .Now its time to calculate error metrics in both R and python.Key component of most measures is difference between actual y and predicted y (“error”).

As It is a regression analysis i have calculated

MSE(Mean Squared error) :basically measures average squared error of our predictions.

MAE(Mean Absolute error) : In MAE the error is calculated as anaverage of absolute differences between the target values and thepredictions.

RMSE(Root Mean Squared error) : is just the square root of MSE.MAPE(Mean Absolute Percentage error) : MAPE can also be thought as weighted versions of MAE.

R code:

*regr.eval(bike[,13],prediction,stats=c('mape','mse','rmse','mae'))*

Python code :

*from sklearn import metrics*

*metrics.mean\_squared\_error(Y,predictions)*

*metrics.mean\_absolute\_error(Y,predictions)*

*np.sqrt(metrics.mean\_squared\_error(Y,predictions))*

*Below is* the tabulated form of error metrics calculations which help us to understand the error in the model that we implemented.

Multiple Linear Regression :

|  |  |  |
| --- | --- | --- |
|  | R | Python |
| MAE | 1.078061e-12 | 2.8618648434074273e-1 |
| MSE | 1.893733e-24 | 5.4623800118681445e-24 |
| RMSE | 1.376130e-12 | 2.3373660414890112e-12 |
| MAPE | 3.230626e-16 | 8.107248027584175e-14 |

Decision Tree :

|  |  |  |
| --- | --- | --- |
|  | R | Python |
| MAE | 3.894721e+02 | 43199.29123116646 |
| MSE | 2.335978e+05 | 162.3738488067641 |
| RMSE | 4.833196e+02 | 207.8443918684516 |
| MAPE | 1.927673e-01 | 8.107248027584175e-14 |

So as its a regression model which we are calculating we know MSE is the average value of difference of actual and predicted values. So I’ll choose MSE as my error metric and when I compare the values for both models and it clearly shows Linear regression is the best fit model for my prediction.Hence I proceed with Linear regression model for this data set.

**CONCLUSION** :

In statistical modeling, regression analysis is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

So thats all , we have completed with the prediction of values so we can deploy the model in any application and use it for our future test cases.This how a machine learning algorithm works and helps us to make our job so easy for spontaneous results.

**THANKYOU**