**Bike Renting**

***Tanishka***

***JULY 2019***

# Contents

1. **[Introduction](#_bookmark0) 3**
   1. [ProblemStatement](#_bookmark1) 3
   2. [Data](#_bookmark2) 3
2. **[Methodology](#_bookmark3) 4**
   1. [PreProcessing](#_bookmark4) 4
      1. Variable identification 5
      2. Visualization 5
3. Univariate Analysis 5
4. Bivariate Analysis 6
   * 1. Missing Value Analysis 9
     2. Outlier Analysis 9
     3. Feature Selection 9
5. Correlation 10
6. Anova 10
   * 1. Feature Scaling 13
   1. Modelling 13
      1. Model Selection 14
7. Decision Tree 14
8. Random Forest 15
9. Linear Regression 16
   * 1. Visualizing models 17
10. Prediction Plots 17
11. **Conclusion 18**
    1. Model Evaluation 18
       1. Mean Absolute Error(MAE) 18
       2. Mean Squared Error(MSE) 18
    2. Model Selection 18

**Appendix A -Imp Plots 19**

**Appendix B 21**

1. **Code 21**

**Python- Code 27**

**Chapter 1**

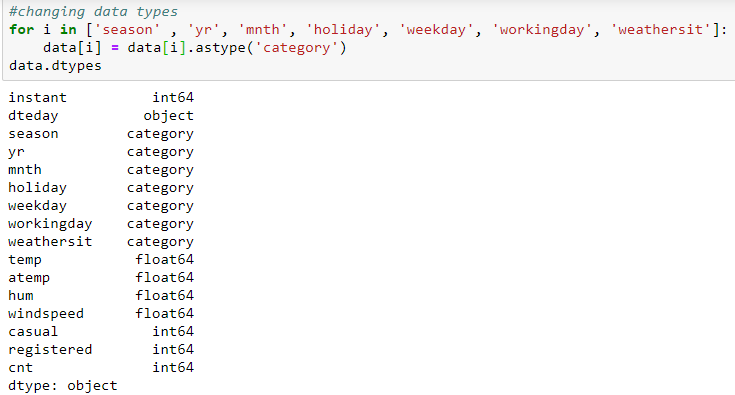
1. ***Introduction*** 
   1. ***Problem Statement:*** The objective of this Project is to Predict the count of the bikes to be rented on daily basis. This count will take environmental and seasonal settings from the historical data in account while forecasting the daily demand. We would be building a model that can successfully predict the count of rentals on relevant factors.
   2. ***Data:*** As the dataset given has dependent and independent values, it will come under supervise Machine learning. Our task is to build Regression models which will help us predicting the count of bikes which will get rented depending on the factors provided. Given below is a sample of the data set that we are using for our prediction.

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| instant | Daily customer index |
| dteday | Date index for both the years |
| season | Season (1:springer, 2:summer, 3:fall, 4:winter) |
| yr | Year (0: 2011, 1:2012) |
| mnth | Month (1 to 12) |
| holiday | weather day is holiday or not (extracted fromHoliday Schedule) |
| weekday | Day of the week |
| workingday | If day is neither weekend nor holiday is 1, otherwise is 0. |
| weathersit | (extracted fromFreemeteo) 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\_maxt\_min), t\_min=-16, t\_max=+50 (only in hourly scale) |
| hum (Humidity) | 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 | The number of registered users at a given day |
| cnt (Count) | Total Rentals with both casual and registered users |

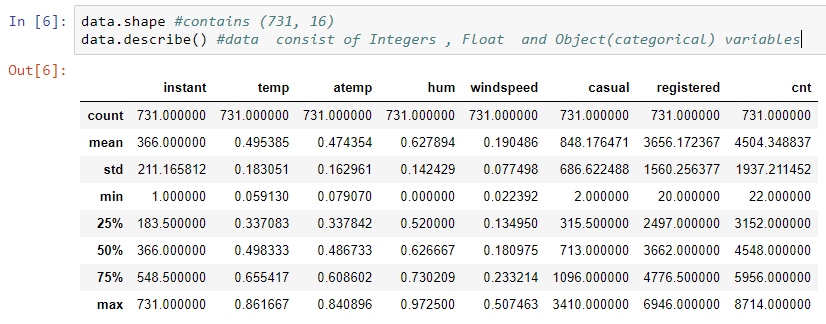
This dataset contains the rental count in between year 2011 and 2012 based on seasonal and environment etc... This is a new way for traditional bike rent. The whole process from registration to rental return back is automated. Here is our data explanation

**Chapter 2**

1. ***Methodology***
   1. ***Pre Processing***:Before we proceeding to create our model on top of the provided data. It is necessary to do Exploratory Data Analysis. EDA is very first and necessary step to take before proceeding further. As the result depends on the data, EDA makes sure the quality of input data is high which will lead to high quality results. We can perform EDA as follows:
      1. **Variable Identification:**In Order to understand the data, we need to first, Identifying Predictor (Input) and Target (output) variables. Then, Identifying the data type and category of the variables.
2. *Types of Variable*: Our Target Variable is ‘CNT’ , and Predictor variables are (dteday,season,yr,mnth,holiday,weekday,workingday,weathersit, temp, atemp,hum,windspeed,casual,registered) .

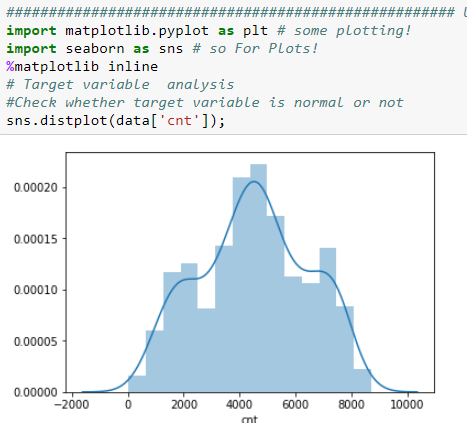
1. *Data Types*: Character(dteday), Numeric(instant,season,yr,mnth,holiday,weekday,workingday,weathersit,casual,registered,cnt ) ,factor(temp,atemp,windspeed).

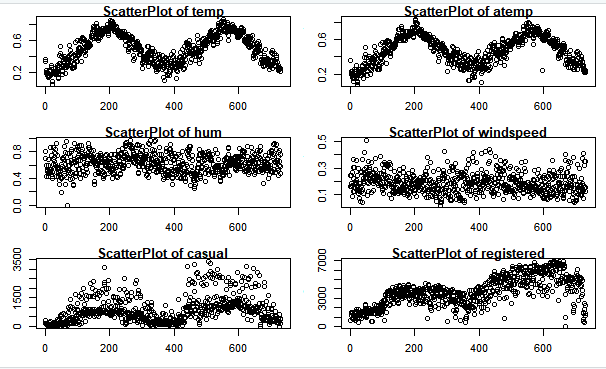
We have converted the data to category for categorical variable.

1. *Variable Categories*: Categorical (season, yr, mnth, holiday, weekday, workingday, weathersit), Continious (temp, atemp, hum, windspeed, casual,registered)

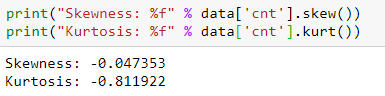
**2.1.2 Visualization:**Exploring Variables one by one to understand central tendency, spread of the variable, distribution of each category, association and disassociation between variables at a predefined significance level.

### A) Uni-variate Analysis: Checking the distribution of individual variables.





Checking for Skewness and Kurtosis: Skewness is usually described as a measure of a dataset’s symmetry – or lack of symmetry.   A perfectly symmetrical data set will have a skewness of 0. If the skewness is between -0.5 and 0.5, the data are fairly symmetrical If the skewness is between -1 and – 0.5 or between 0.5 and 1, the data are moderately skewed. If the skewness is less than -1 or greater than 1, the data are highly skewed.



Here Skewness is very less so target variable is normal distribution.

**B) Bi-variate Analysis:**We are checking relation of variables with each other to understand the relationship of variables with our target variable.

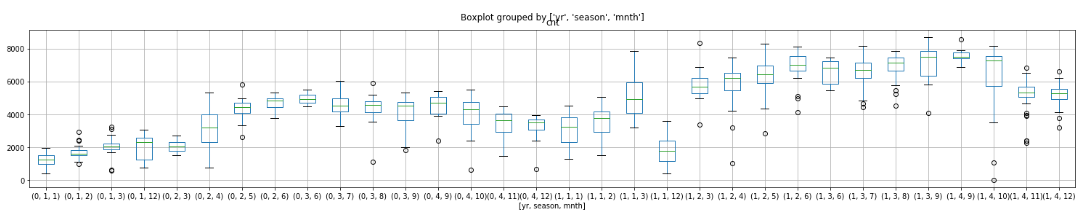


Fig 1.1 – Boxplot of count of bike rents grouped by year, season and month

From this, we can infer that the count of rental bikes have increased over the year 2012. In each particular year, month march, may, June, July, Sept have highest rentals. Also we see that rentals are higher in Season3 (fall) and least in Season1 (Spring).

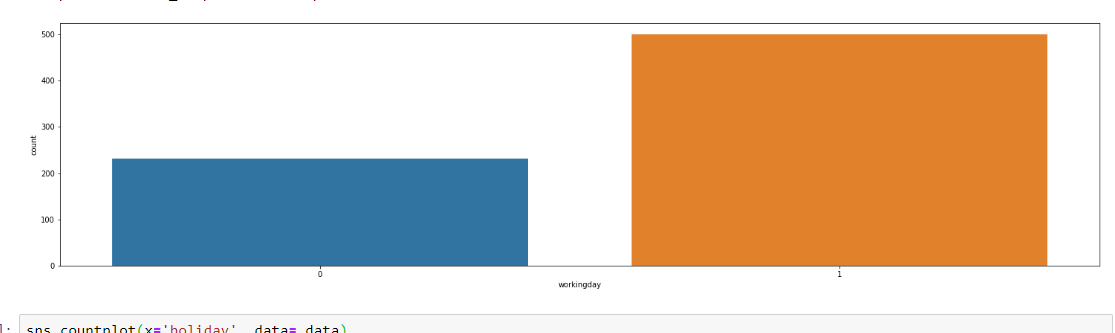


Fig 1.2 Bar graph of working day

From Fig 1.2 (Bar graph of working day) we can infer that people rent bikes more on working days as there may be chances of rental requirements to commute rather than on non-working days.

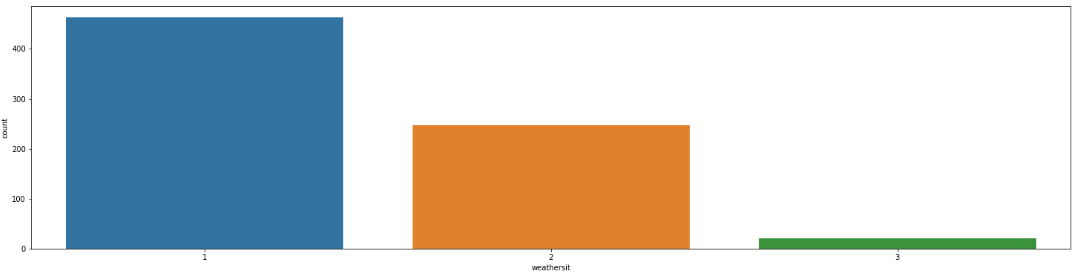


Fig 1.3 Bar graph of weather sit

From Fig 1.3 (Bar graph of weather sit) we can infer that on weatersit1 (clear weather), people prefer more to rent bikes rather on weathersit3(heavy rain).

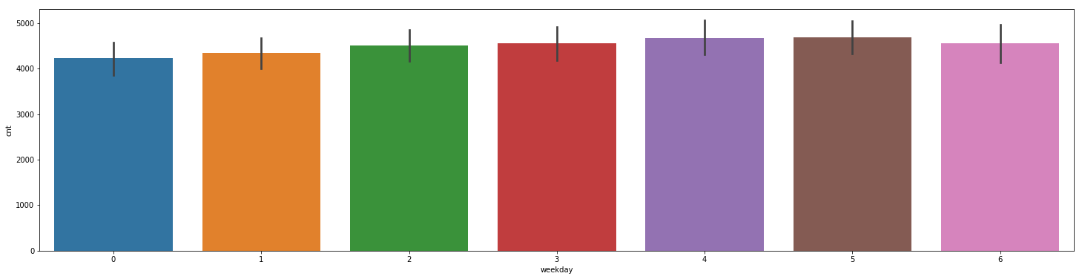


Fig 1.4 Boxplot of weekdays

From 1.4 (Boxplot of weekdays) we can clearly see that there is almost no difference in rentals on each day, it means that all the days have almost same rental counts.

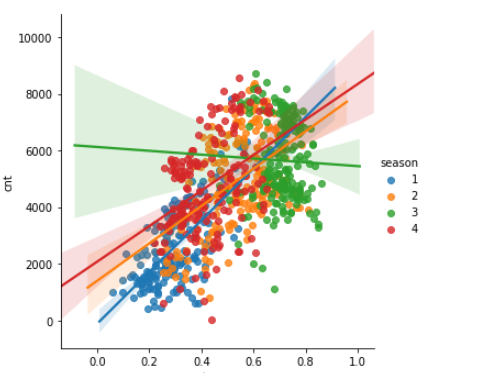


Fig 1.5 Linear plot of temp Vs count (season wise)

From 1.5(Linear plot of temp Vs count (season wise) shows that high temperatures are the main reason of people to rent out bikes more. Also it shows the variation of counts over temperature in each season.

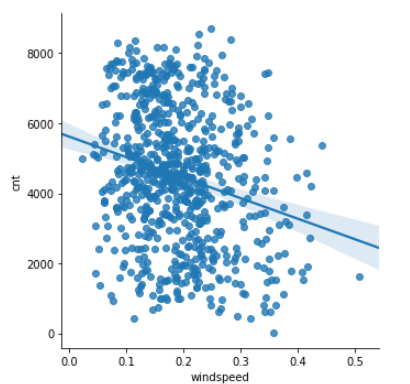
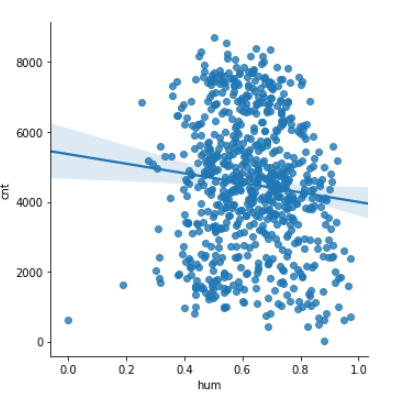
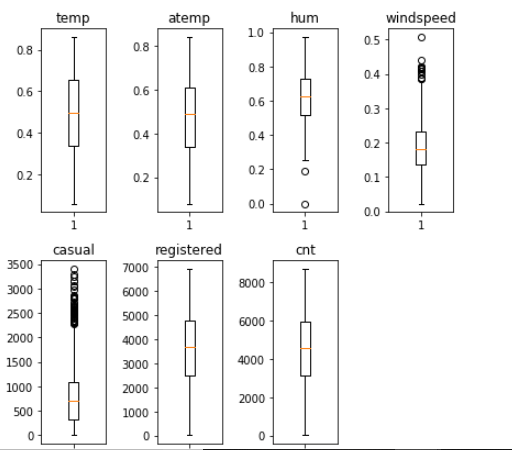


Fig 1.6 Linear plot of humidity vs count Fig 1.7 Linear plot of windspeed vs count

From 1.6(Linear plot of humidity vs count) and 1.7(Linear plot of windspeed vs count) it is clear that humidity and windspeed is inversely proportion with count, i.e. with increase in humidity, the rental counts decrease.

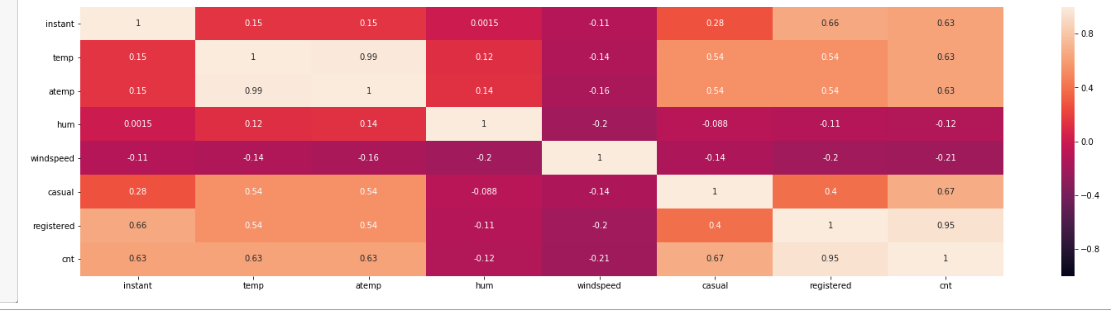
**2.1.3 Missing values treatment:**Missing values occur when no data value is stored for the variable in an observation. Missing values are a common occurrence, and you need to have a strategy for treating them. A missing value can signify a number of different things in your data. Perhaps the data was not available or not applicable or the event did not happen. It could be that the person who entered the data did not know the right value, or missed filling in. Typically, ignore the missing values, or exclude any records containing missing values, or replace missing values with the mean, or infer missing values from existing values. We check for missing values in our data and came to know that there are no missing values

**2.1.4 Outlier treatment:** An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. Outliers can drastically change the results of the data analysis and statistical modelling. There are numerous unfavourable impacts of outliers in the data set. It increases the error variance and reduces the power of statistical tests. If the outliers are non-randomly distributed, they can decrease normality. They can also impact the basic assumption of Regression, ANOVA and other statistical model assumptions. The boxplot for our data could be seen as follows:



The boxplot helps us to identify outliers in each column. In our data, outliers are found in humidity, windspeed and casual columns. Outliers in humidity were imputed with mean of that column. (only 2 outliers). Outliers in windspeed were imputed with season wise mean of windspeed. Outliers in casual is imputed by subtracting registration number from cnt, as I saw the casual counts are nearer to this equation.

**2.1.5 Feature Selection :**Variable selection is an important aspect of model building. It helps in building predictive models free from correlated variables, biases and unwanted noise. It helps in selecting a subset of relevant features (variables, predictors) for use in model construction and subset of a learning algorithm’s input variables upon which it should focus attention, while ignoring the rest.

* **Correlation Analysis :**We make heat map to understand the co relation of contiguous variable. A heatmap is a graphical representation of data where the individual values contained in a matrix are represented as colors. Here each numerical variable’s correlation is mapped with each other’s in a matrix which has been plotted in the following heatmap.

Our correlation matrix shows results as follows

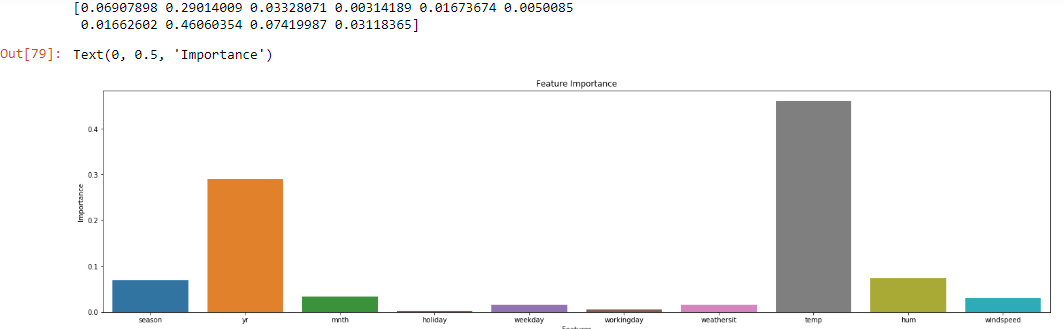
1. Temp and atemp are highly correlated 2. Registered and count are highly correlated (since count is a sum of registered and casual)
2. We can now remove one of the highly correlated variable so that our model can perform well with much accuracy.

So, we are dropping variables('instant','atemp','casual','registered'). Instant is unique for all observations hence has no significance, atemp is strongly correlated with temp, cnt is sum of casual and registration.

Let’s check the importance of our features to our target variable.

* **Feature Importance:** The concept is really straightforward: We measure the importance of a feature by calculating the increase in the model’s prediction error after permuting the feature. A feature is “important” if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction. A feature is “unimportant” if shuffling its values leaves the model error unchanged, because in this case the model ignored the feature for the prediction.

1. **Checking via Tree**



Here , we can see the the importance of Holiday(0.00344259) and workingday (0.0048055) is extremly low . We can remove these variables for our dataset as its importance is not much to be considred.

1. **ANOVA** (Analysis of Variances)

Analysis of variance (ANOVA) is a statistical technique that is used to check if the means of two or more groups are significantly different from each other. ANOVA checks the impact of one or more factors by comparing the means of different samples. As our target variable is numerical we will use ANOVA for feature selection technique to see whether any categorical variable is related to target variable. The higher the variance between the variables, the less likely that they are related (or correlated). The result of anova is as follows:

|  |
| --- |
| 1. For Seasons:  dfsum\_sqmean\_sq F PR(>F)  season 3.0 9.505959e+08 3.168653e+08 128.769622 6.720391e-67  Residual 727.0 1.788940e+09 2.460715e+06 NaNNaN  2. For Year:  dfsum\_sqmean\_sq F PR(>F)  yr 1.0 8.798289e+08 8.798289e+08 344.890586 2.483540e-63  Residual 729.0 1.859706e+09 2.551038e+06 NaNNaN  3. For Month  dfsum\_sqmean\_sq F PR(>F)  mnth 11.0 1.070192e+09 9.729021e+07 41.903703 4.251077e-70  Residual 719.0 1.669343e+09 2.321757e+06 NaNNaN  4. For Holiday  dfsum\_sqmean\_sq F PR(>F)  holiday 1.0 1.279749e+07 1.279749e+07 3.421441 0.064759  Residual 729.0 2.726738e+09 3.740381e+06 NaNNaN  5. For Weekday  dfsum\_sqmean\_sq F PR(>F)  weekday 6.0 1.765902e+07 2.943170e+06 0.782862 0.583494  Residual 724.0 2.721876e+09 3.759498e+06 NaNNaN  6. For Workingday  dfsum\_sqmean\_sq F PR(>F)  workingday 1.0 1.024604e+07 1.024604e+07 2.736742 0.098495  Residual 729.0 2.729289e+09 3.743881e+06 NaNNaN  7. For Weathersit  dfsum\_sqmean\_sq F PR(>F)  weathersit 2.0 2.716446e+08 1.358223e+08 40.066045 3.106317e-1  Residual 728.0 2.467891e+09 3.389960e+06 NaNNaN |

Ho = Categorical variable is Independent from the Target variable

Ha = Categorical variable is Dependent on the Target variable

--> If the p value of the categorical variable is less than 0.05 then we will consider that the target variable is dependent on the categorical variable for which we reject the null hypothesis.

--> From the above result we can see that only five variables are very much related to target variable hence we delete all the other variables.

|  |
| --- |
| season 3 950595868 316865289 436.234 < 2e-16 \*\*\*  yr 1 884008263 884008263 1217.030 < 2e-16 \*\*\*  mnth 11 187311622 17028329 23.443 < 2e-16 \*\*\*  holiday 1 3306975 3306975 4.553 0.03321 \*  weekday 6 15839061 2639843 3.634 0.00147 \*\*  weathersit 2 185659616 92829808 127.800 < 2e-16 \*\*\* |

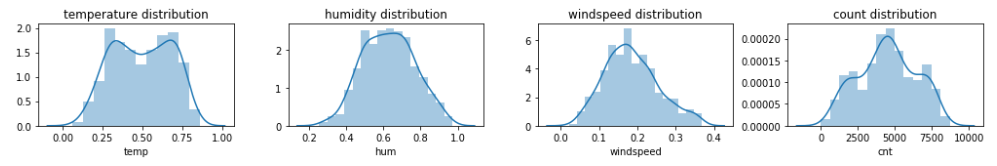
--> Therefore from both the correlation analysis and ANOVA we got some variable which we shouldn’t consider for further processing. The variables that could be deleted are as follows

Numerical: Instant, atemp, casual, registered

Categorical: dteday, holiday, workingday

Hence, We have deleted these Variables from our data set.

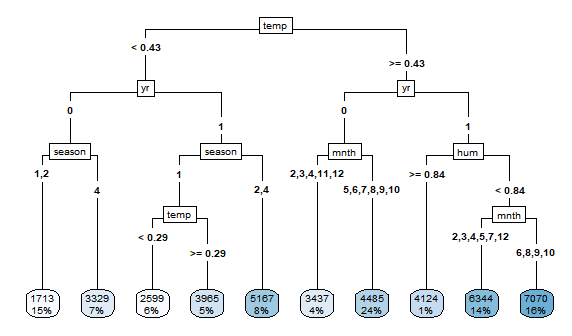
**2.1.6 FeatureScaling:**Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization. Normalization also called Min-Max scaling. It is the process of reducing unwanted variation either within or between variables. Normalization brings all of the variables into proportion with one another. It transforms data into a range between 0 and 1. All our continuous variables are already normalized except the target variable which we prefer not to scale because its variation is spread quite widely and after scaling, the difference between the number is diminishing.



***2.2 Modeling***

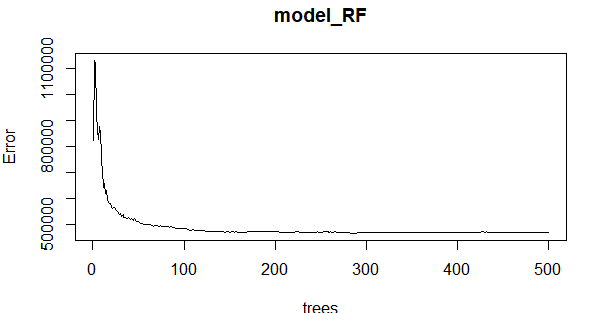
2.2.1 **Model Selection:**For modelling, we are going to use some famous models to our data-set and will conclude the result according to it.

1. **Decision Tree:** Decision tree is a rule. Each branch connects nodes with “and” and multiple branches are connected by “or”. It can be used for classification and regression. It is a supervised machine learning algorithm. Accept continuous and categorical variables as independent variables. Extremely easy to understand by the business users. Split of decision tree is seen in the below tree. Decision tree regression is as follows



**2) Random Forest:**Random Forest or decision tree forests are an ensemble learning method for classification, regression and other tasks. It consists of an arbitrary number of simple trees, which are used to determine the final outcome. In the regression problem, their responses are averaged to obtain an estimate of the dependent variable. Using tree ensembles can lead to significant improvement in prediction accuracy (i.e., better ability to predict new data cases). The goal of using a large number of trees is to train enough that each feature has a chance to appear in several model.

--> As we increase the number of trees the error count decrease until a point and then becomes constant. Error vs number of trees to be used graph is as follows:



--> We can also call our model and get below details

Call:

randomForest(formula = cnt ~ ., data = data\_train, importance = TRUE, ntree = 500)

Type of random forest: regression

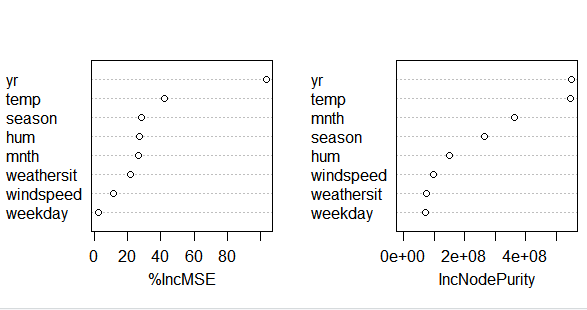
Number of trees: 500

No. of variables tried at each split: 2

Mean of squared residuals: 468080.9

% Var explained: 87.61

--> We can check the importance of our variables in Random Forest Model with (varImpPlot(MODELNAME))



The first graph shows that if a variable is assigned values by random permutation by how much will the MSE increase. Higher the value, higher the importance. On the other hand, node purity is measured by the Gini index which is the difference between before and after split on that variable.

**Linear Regression:**Linear regression is the most basic type of regression and commonly used predictive analysis. Linear regression is an approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression.

We have to convert all the categorical variable into Dummies because Machine learning algorithm require numbers as input. In regression analysis, a dummy variable (also known as an indicator variable, design variable, Boolean indicator, binary variable, or qualitative variable) is one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome

--> Following is the summary of the Linear model:

Call:

lm(formula = cnt ~ ., data = data\_train)

Residuals:

Min 1Q Median 3Q Max

-3541.5 -368.3 50.7 485.1 2955.6

Coefficients:

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

(Intercept) 1339.15 267.82 5.000 7.68e-07 \*\*\*

season2 979.10 200.28 4.889 1.33e-06 \*\*\*

season3 796.55 236.15 3.373 0.000795 \*\*\*

season4 1736.98 196.51 8.839 < 2e-16 \*\*\*

yr1 2080.53 64.46 32.277< 2e-16 \*\*\*

mnth2 109.74 155.14 0.707 0.479640

mnth3 651.92 182.54 3.571 0.000386 \*\*\*

mnth4 409.34 276.41 1.481 0.139199

mnth5 872.56 298.60 2.922 0.003616 \*\*

mnth6 763.84 314.64 2.428 0.015513 \*

mnth7 303.82 348.70 0.871 0.383960

mnth8 799.96 336.10 2.380 0.017642 \*

mnth9 1267.97 293.66 4.318 1.87e-05 \*\*\*

mnth10 577.38 267.11 2.162 0.031078 \*

mnth11 -136.81 253.38 -0.540 0.589456

mnth12 -138.31 193.21 -0.716 0.474377

weekday1 54.05 117.40 0.460 0.645397

weekday2 358.06 117.91 3.037 0.002504 \*\*

weekday3 391.12 120.99 3.233 0.001299 \*\*

weekday4 408.10 117.50 3.473 0.000554 \*\*\*

weekday5 450.23 120.25 3.744 0.000200 \*\*\*

weekday6 482.12 121.01 3.984 7.67e-05 \*\*\*

weathersit2 -516.11 85.94 -6.005 3.45e-09 \*\*\*

weathersit3 -2110.52 233.00 -9.058 < 2e-16 \*\*\*

temp 3823.23 463.06 8.256 1.09e-15 \*\*\*

hum -1173.54 340.61 -3.445 0.000613 \*\*\*

windspeed -2131.72 502.77 -4.240 2.62e-05 \*\*\*

---

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

Residual standard error: 761.1 on 557 degrees of freedom

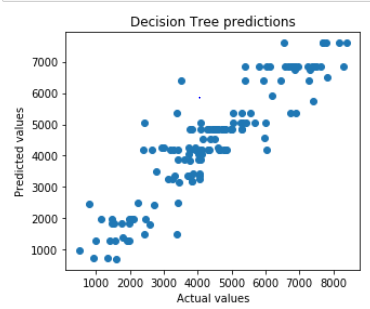
Multiple R-squared: 0.8538, Adjusted R-squared: 0.847

1. statistic: 125.1 on 26 and 557 DF, p-value: < 2.2e-16

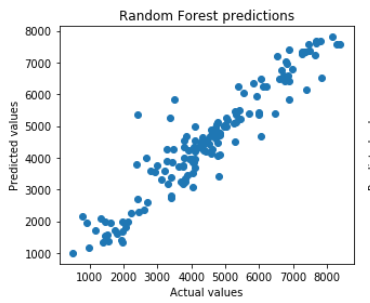
2.2.2 **Visualizing models** :We can see the plots of our predicted model to understand it better

A) Prediction Plots:

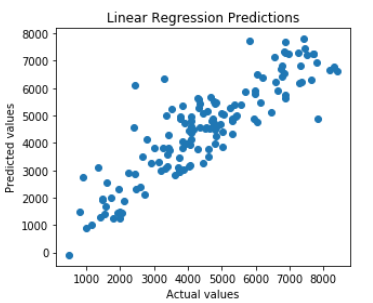
**Decision Tree:**



**Random Forest:**



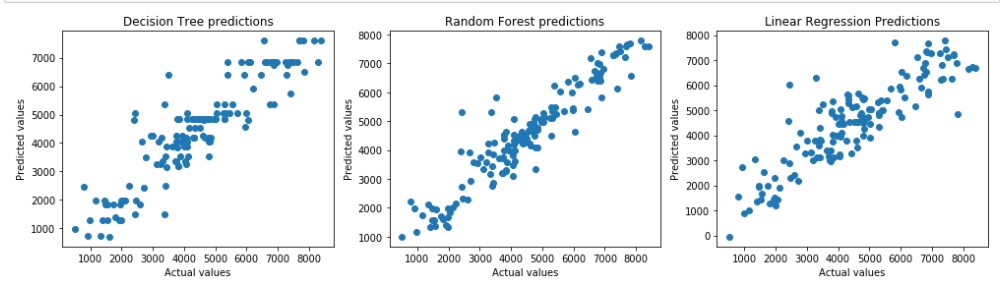
**Linear regression:**



1. ***Conclusion***
   1. ***Model Evaluation:*** Model evaluation is done on basis of evaluation metrics or error metrics. Evaluation metrics explain the performance of a model. An important aspect of evaluation metrics is their capability to discriminate among model results. Simply, building a predictive model is not our motive. But, creating and selecting a model which gives high accuracy on out of sample data. Hence, it is crucial to check accuracy or other metric of the model prior to computing predicted values. In our data as we applied regression models we have error metrics like Mean square error(MSE), MAPE, Root mean square error (RMSE), Mean absolute error (MAE).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Language/Model |  | **Python** |  | |  | **R** | |  |  |
| **MODELS** | MSE | RMSE | MAPE | R-SQ | MAE | | RMSE | MAPE | R-SQ |
| **Decision Tree** | 580795 | 762.09 | 16.6 | 0.8296 | 682.26 | | 971.37 | 25.98 | 0.74 |
| **Random Forest** | 376333 | 613.46 | 13.62 | 0.8896 | 512.64 | | 767.94 | 20.64 | 0.84 |
| **Linear regression** | 782200 | 884.42 | 19.25 | 0.7705 | 589.63 | | 810.13 | 18.29 | 0.82 |

* 1. ***Model Selection :***We can see that all models perform comparatively on average and therefore we select Random forest classifier models for better prediction.



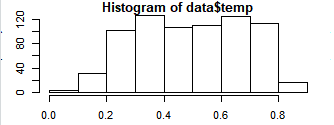
From the above plots of Actual Vs Predicted values, we can infer that values of Random forest falls on straight line indicating random forest fits better than the other three models.

Also amongst the three models, Random forest has best R-sq. (Coef. of determination). Hence we’ll fix Random Forest as our model.

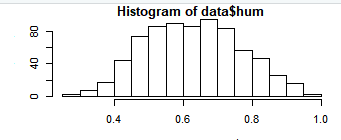
**Appendix A -Imp Plots**

**Distribution of data**

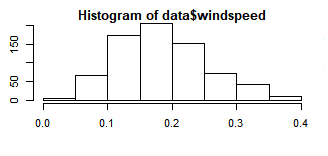
* + - 1. **Cnt Vs Temp**

****

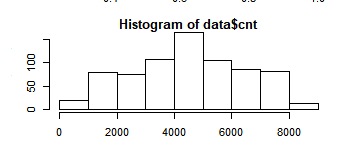
* + - 1. **Cnt Vs Hum**



* + - 1. **Cnt Vs windspeed**

****

* + - 1. **Cnt**

****

**Appendix B**

**R Code**

rm (list = ls()) # Cleaing the evironment

setwd('C:/Users/HP/Desktop/TanishkaProject1') #setting directory

getwd()

#Reading the data

data = read.csv('day.csv')

#knowing structure of our data

str(data) #We have 731 obs. of 16 variables

#Changing the data types of categorical variables

data$season= factor(data$season)

data$yr = factor(data$yr)

data$mnth = factor(data$mnth)

data$holiday = factor(data$holiday)

data$weekday = factor(data$weekday)

data$workingday = factor(data$workingday)

data$weathersit = factor(data$weathersit)

############################DATA PRE-POCESSING########################

######################################################################

#1.\*\*\*\*MISSING VLUE ANALYIS\*\*\*\*

#sum(is.na(data)) = 0

sapply(data, function(x) sum(is.na(x))) # We don't have any missing value

#2.\*\*\*\*VISUALIZATION\*\*\*\*

install.packages("ggplot2")

install.packages("scales")

library(ggplot2)

library(scales)

par(mar=c(3,3,1,1))

par(mfrow=c(3,2))

#A)Univariate

#variable quantity

#par("mar") - set it to par(mar= c(1,1,1,1)) to avoid margin error for plots

#par("mar")

plot(data$temp, main="ScatterPlot of temp")

plot(data$atemp, main="ScatterPlot of atemp")

plot(data$hum, main="ScatterPlot of hum")

plot(data$windspeed, main="ScatterPlot of windspeed")

plot(data$casual, main="ScatterPlot of casual")

plot(data$registered, main="ScatterPlot of registered")

plot(data$cnt, main="ScatterPlot of cnt")

#B)bi-variant

#B.1)categorical variables vs target variable

plot(cnt ~ season , data = data, main = 'season')# we see least rentals are in season 1 and most in season 3

plot(cnt ~ yr, data= data, main = 'yr')# #we see rental are high in 2012, this tells the rental is increasing with time

plot(cnt ~ mnth, data= data, main = 'mnth')#we see rental high from march to oct

plot(cnt ~ holiday, data= data, main = 'holiday')#we see rental high in weekdays

plot(cnt ~ weekday , data = data, main = 'weekday')#not much difference

ggplot(data , aes\_string(x=data$workingday)) + geom\_bar(stat = "count", fill = "DarkslateBlue") +

xlab("Working day") + ylab("Count") + ggtitle("Working day distribution") + theme(text = element\_text(size = 15))

#bikes are rented more on working days

plot(cnt ~ weathersit , data = data, main = 'weathersit')#we see rental are high with clear weather and low with rainy

#B.2)continuous variables vs target variable

reg1 = lm(cnt ~ temp, data = data)

with(data ,plot(temp, cnt, main = 'temp'))

abline(reg1) #rental counts increase with increase in temperature

reg2 = lm(cnt ~ atemp , data = data)

with (data, plot(atemp, cnt, main = 'atemp'))

abline(reg2)

reg3 = lm(cnt ~ hum, data = data)

with(data ,plot(hum, cnt, main = 'hum'))

abline(reg3) #rental count decrease with increase in humidity

reg4 = lm(cnt ~ windspeed , data = data)

with (data, plot(windspeed, cnt, main = 'windspeed'))

abline(reg4)

reg5 = lm(cnt ~ casual, data = data)

with(data ,plot(casual,cnt,main = 'casual'))

abline(reg5)

reg6 = lm(cnt ~ registered , data = data)

with (data, plot(registered, cnt, main = 'registered'))

abline(reg6)

#3.\*\*\*\*OUTLIER ANALYSIS\*\*\*\*

numeric\_index = sapply(data, is.numeric) # creating numerical value index

numeric\_data = data[,numeric\_index] # storing numeric data

cnames = colnames(numeric\_data) #storing numeric data column names

#Creating box-plot to analyze outliers

for (i in 1:length(cnames)){

assign(paste0("gn", i), ggplot(aes\_string(y = cnames[i], x = "cnt"), data = subset(data)) +

stat\_boxplot(geom = "errorbar", width = 0.5) +

geom\_boxplot(outlier.colour = "red", fill = "blue", outlier.shape = 18, outlier.size = 1, notch = FALSE) +

theme(legend.position = "bottom") + labs(y = cnames[i], x="count") + ggtitle(paste("Boxplot of count for", cnames[i])))

}

gridExtra::grid.arrange(gn2, gn3, gn4,gn5, gn6, gn7, ncol = 3, nrow = 3) # excludif gn1 as that is unique for each observation

#replace outliers with NA

for(i in cnames) {

print(i)

val = data[,i][data[,i] %in% boxplot.stats(data[,i]) $out]

print(length(val))

data[,i][data[,i] %in% val] = NA

}

#imputing NA values

data$hum[is.na(data$hum)] = mean(data$hum,na.rm = T) #closest value was from mean

data$casual[is.na(data$casual)] = data$cnt - data$registered # as cnt geg + casul

data$windspeed[is.na(data$windspeed)] = mean(data$windspeed, na.rm = T)#closest value was from mean

# creating copy of imputed data

copy1 = data

#4.\*\*\*\*FEATURE SELECTION\*\*\*\*

install.packages("corrgram") # for correlation graph

library(corrgram)

#A. Correlation check on continuous variable

round(cor(numeric\_data),2) #Correlation tablecolumn wise

corrgram(data[, numeric\_index], order = F, upper.panel = panel.pie, text.panel = panel.txt, main = "correlation plot") # temp & ateamp, cnt=registered+casual

#temp and atemp are strongle correlated

#B. Anova test on categorical variable

#creating sunset randomly for anova

data = subset(data, select=-c(instant, atemp, casual, registered, dteday))

anova\_test=aov(cnt~season + yr + mnth + holiday + weekday + workingday + weathersit, data = data)

summary(anova\_test)

data = subset(data, select=-c(holiday, workingday))

data

#5.\*\*\*\*FEATURE SCALING\*\*\*\*

hist(data$temp)

hist(data$hum)

hist(data$windspeed)

hist(data$cnt)

#Scaling categorical variable with dummies

install.packages("dummies") #for scaling

library(dummies)

#dummy.data.frame()

data\_new = dummy.data.frame(data, sep = '\_')

data\_new

#data['cnt'] = (data$cnt-min(data$cnt))/(max(data$cnt)-min(data$cnt))#no need of scaling target variable.

############################MODELING########################

#############################################################

#Sampling

#1. Non scaled data

set.seed(101)

train\_index = sample(1:nrow(data), 0.8\*nrow(data))

data\_train = data[train\_index,]

data\_test = data[-train\_index,]

#2. Scaled data

#set.seed(102)

train\_scaled = sample(1:nrow(data\_new), 0.8\*nrow(data\_new))

data\_train\_scaled = data\_new[train\_scaled,]

data\_test\_scaled = data\_new[-train\_scaled,]

#Function to calculate MAPE

mape = function(actual, predict){

mean(abs((actual-predict)/actual))\*100

}

##1. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Decision tree\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

par(mar=c(1,1,1,1))

par(mfrow=c(1,1))

install.packages("rpart.plot")

library(rpart.plot)

library(rpart)

#Model

set.seed(101)

model\_DT = rpart(cnt~. , data = data\_train, method = "anova")

summary(model\_DT)

plt = rpart.plot(model\_DT, type = 5, digits = 2, fallen.leaves = TRUE)

#?rpart.plot

#Predictions

DT\_Predict = predict(model\_DT, data\_test[,-9])

plot(data\_test$cnt, DT\_Predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'DT model')

#Evaluation statistics

install.packages("caret")

library(caret)

postResample(DT\_Predict, data\_test$cnt)#R-sq = 0.74

mape(data\_test$cnt, DT\_Predict) #25.98

##2. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Random forest\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

install.packages("randomForest")

library(randomForest)

library(inTrees)

#Model

set.seed(101)

model\_RF = randomForest(cnt ~. , data\_train, importance = TRUE, ntree = 500)

model\_RF

#Error plotting

plot(model\_RF) #my error i decreasing with higher number of trees

#Predict test data using RF model

RF\_predict = predict(model\_RF, data\_test[,-9])

plot(data\_test$cnt, RF\_predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'RF model')

#Evaluation statistics

postResample(RF\_predict, data\_test$cnt)#R-sq = 0.84

mape(data\_test$cnt, RF\_predict) #20.64

varImpPlot(model\_RF) #Check the importance of variables in our RT model

##3. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Linear regression\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#Model

set.seed(101)

model\_LR = lm(cnt ~. , data = data\_train\_scaled)

summary(model\_LR)

#Predictions

LR\_predict = predict(model\_LR, data\_test\_scaled[,-32])

plot(data\_test\_scaled$cnt, LR\_predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'LR model')

LR\_predict

#Evaluation statistics

postResample(LR\_predict, data\_test\_scaled$cnt)#R-sq = 0.82

mape(data\_test\_scaled$cnt, LR\_predict) #18.29

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Compare all three models\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

plot(data\_test$cnt, DT\_Predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'DT model')

plot(data\_test$cnt, RF\_predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'RF model')

plot(data\_test\_scaled$cnt, LR\_predict, xlab = 'Actual values', ylab = 'Predicted values', main = 'LR model')

#As per the above calculation Random forest is best suite for our Prediction.

**Python Code**

#!/usr/bin/env python

# coding: utf-8

# In[2]:

import os #Intraction local system directories

import pandas as pd #Data processing

import numpy as np #Linear algebra

os.chdir("C:/Users/HP/Desktop/TanishkaProject1")

data = pd.read\_csv('day.csv', sep = ',')

os.getcwd()

# In[3]:

data.shape

# In[4]:

data.head()

# In[5]:

data.dtypes #checking datatypes

# In[6]:

#changing data types

for i in ['season' , 'yr', 'mnth', 'holiday', 'weekday', 'workingday', 'weathersit']:

data[i] = data[i].astype('category')

data.dtypes

# In[7]:

#understanding the data

data.shape #contains (731, 16)

data.describe() #data consist of Integers , Float and Object(categorical) variables

# In[8]:

##################################################### Univariate Analysis##########################################

import matplotlib.pyplot as plt # some plotting!

import seaborn as sns # so For Plots!

get\_ipython().run\_line\_magic('matplotlib', 'inline')

# Target variable analysis

#Check whether target variable is normal or not

sns.distplot(data['cnt']);

# In[9]:

#Data Distribution for independent numeric variables

fig = plt.figure(figsize=(10,6))

#Check whether variable 'temp'is normal or not

sns.distplot(data['temp']);

#Check whether variable 'atemp'is normal or not

sns.distplot(data['atemp']);

#Check whether variable 'hum'is normal or not

sns.distplot(data['hum']);

#Check whether variable 'windspeed'is normal or not

sns.distplot(data['windspeed']);

fig.legend(labels=['temp','atemp','hum','windspeed'],)

plt.show()

# In[10]:

#Check whether variable 'casual'is normal or not

fig = plt.figure(figsize=(10,6))

sns.distplot(data['casual']);

#Check whether variable 'registered'is normal or not

sns.distplot(data['registered']);

fig.legend(labels=['casual','registered'])

plt.show()

# In[11]:

fig\_size = plt.rcParams['figure.figsize']

fig\_size[0] = 10

fig\_size[1] = 4

plt.rcParams['figure.figsize'] = fig\_size

data.boxplot(column=['temp','atemp','hum','windspeed'])

# it is clearly showing that there are outliers present in 'windspeed,hum' varible

# In[12]:

fig\_size = plt.rcParams['figure.figsize']

fig\_size[0] = 10

fig\_size[1] = 4

plt.rcParams['figure.figsize'] = fig\_size

data.boxplot(column=['casual','registered'])

# it is clearly showing that there are outliers present in 'casual' varible

# In[13]:

print("Skewness: %f" % data['cnt'].skew())

print("Kurtosis: %f" % data['cnt'].kurt())

#Here Skewness is very less so target variable is normal distribution

# In[14]:

fig\_size = plt.rcParams['figure.figsize']

fig\_size[0] = 25

fig\_size[1] = 6

plt.rcParams['figure.figsize'] = fig\_size

data.boxplot(column='cnt', by=['yr','season','mnth'])

#over the years count of rental bikes have increased

# In[15]:

sns.countplot(x='workingday', data= data) #working days have more count

# In[16]:

sns.barplot(x='weekday', y='cnt', data= data) #weekdays are almost same

# In[17]:

sns.countplot(x='holiday', data= data) #Rentals are high on holidays when compare to weekdays

# In[18]:

sns.countplot(x='weathersit', data= data) #weathersit 1 have highest count

# In[19]:

#sns.lmplot(x='temp',y='cnt',data=data)

sns.lmplot(x='temp',y='cnt',data=data, hue='season') #Increasing temp is increasing rental count

# In[20]:

sns.lmplot(x='hum',y='cnt',data=data)

# In[21]:

sns.lmplot(x='windspeed',y='cnt',data=data)

# In[22]:

########################################## missing values ##############################################

#missingvalue = data.isnull().sum()

#missingvalue

total = data.isnull().sum().sort\_values(ascending=False)

percent = (data.isnull().sum()/data.isnull().count()).sort\_values(ascending=False)

missing\_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])

missing\_data

# In[23]:

######################################### Outlier Analysis ####################################################

plt.subplot(2,4,1)

plt.boxplot(data['temp'])

plt.title('temp')

plt.subplot(2,4,2)

plt.boxplot(data['atemp'])

plt.title('atemp')

plt.subplot(2,4,3)

plt.boxplot(data['hum'])

plt.title('hum')

plt.subplot(2,4,4)

plt.boxplot(data['windspeed'])

plt.title('windspeed')

plt.subplot(2,4,5)

plt.boxplot(data['casual'])

plt.title('casual')

plt.subplot(2,4,6)

plt.boxplot(data['registered'])

plt.title('registered')

plt.subplot(2,4,7)

plt.boxplot(data['cnt'])

plt.title('cnt')

plt.tight\_layout()

#We have outlerss in hum, windspeed and casual

# In[34]:

#Deleting Outliers

cname = ['temp','atemp', 'hum','windspeed','casual','registered','cnt']

numeric\_data = data[cname]

for i in cname:

print(i)

q75, q25 = np.percentile(data.loc[:,i],[75,25])

iqr = q75- q25

min = q25 - (1.5\*iqr)

max = q75 + (1.5\*iqr)

data.loc[data.loc[:,i] < min , i] = np.nan

data.loc[data.loc[:,i] > max , i] = np.nan

#Deleting purposfully so that while imputing we can chek the results based on these values as we already kbow the values here

#data.iloc[40,11] = np.nan

#data.iloc[40,12] = np.nan

#data.iloc[40,13] = np.nan

# In[35]:

pd.isnull(data).sum() #after deletion we have 59 Missing values

# In[36]:

#Imputing Missing value

#1. Imputing hum

#Actual data.iloc[40,11] = 0.494783

#data['hum'] = data['hum'].fillna(data['hum'].median()) #=0.6283335

#data['hum'] = pd.DataFrame(KNN(k=3).fit\_transform(data[['hum']])) #= 0.0

data['hum'] = data['hum'].fillna(data['hum'].mean()) # = 0.41

#print(data.iloc[40,11])

# so we are imuting this value with medin since that is the closest value

# In[37]:

#2. Imputing windspeed

#Actual data.iloc[40,12] = 0.08645

data['windspeed'] = data['windspeed'].fillna(data['windspeed'].median()) #=0.179108

#data['windspeed'] = pd.DataFrame(KNN(k=3).fit\_transform(data[['windspeed']])) #= 0.0

#data['windspeed'] = data['windspeed'].fillna(data['windspeed'].mean()) # = 0.18641671227336115

#print(data.iloc[40,12])

# so we are imuting this value with medin since that is the closest value

# In[38]:

#3. imputing missing casual

#original = 43

#data['casual'] = data['casual'].fillna(data['casual'].median()) # 675

#data['casual'] = data['casual'].fillna(data['casual'].mean()) #732

#data['casual'] = pd.DataFrame(KNN(k=3).fit\_transform(data[['casual']])) #= 0.0

data['casual'] = data['casual'].fillna(data['cnt']-data['registered']) # we saw the values are closer this way that is 245

#print(data.iloc[40,13])

# In[39]:

############################################# feature selection #######################################

#drawing correlation matrix between all numeric variables and analyse what are the variables are important

colname = ['temp','atemp', 'hum','windspeed','casual','registered','cnt']

heat\_map = data[colname]

sns.heatmap(heat\_map.corr(), vmin=-1.00, vmax=1.00, annot=True)

# we have made heat map to understand the corelation of continious variable

# In[40]:

#Dropping variables

#data1 = data.copy()

data = data.drop(['instant','atemp','casual','registered','dteday'], axis=1)

#instant is unique for all observations hence has no significance

#atemp is strongly correlated with temp

#cnt = casual + registered

# In[41]:

data.head() # After Drop out dataset contains

# In[42]:

#feature importance from random forest

from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(n\_estimators = 100, random\_state = 24).fit(data.iloc[:,0:10],data.iloc[:,10])

print(rf.feature\_importances\_)

#Feature importance plotting

names=list(data)

names = names[0:10]

sns.barplot(x=names ,y=rf.feature\_importances\_) #removing holiday, working day since feature is extremly low

plt.title('Feature Importance')

plt.xlabel('Features')

plt.ylabel('Importance')

# In[43]:

#anova test

#since the target variable is continuous

import statsmodels.api as sm

from statsmodels.formula.api import ols

#from scipy import stats

# In[44]:

#loop for ANOVA test Since the target variable is continuous

#categorical\_vars = ['season','yr', 'mnth','holiday','weekday','workingday','weathersit']

#for i in categorical\_vars:

#f, p = stats.f\_oneway(data[i], data["cnt"])

#print("P value for variable "+str(i)+" is "+str(p))

# In[45]:

mod1 = ols('cnt ~ season', data = data).fit()

aov\_table1 = sm.stats.anova\_lm(mod1, type=2)

print(aov\_table1) #keep

# In[46]:

mod2 = ols('cnt ~ yr', data = data).fit()

aov\_table2 = sm.stats.anova\_lm(mod2, type=2)

print(aov\_table2) #keep

# In[47]:

mod3 = ols('cnt ~ mnth', data = data).fit()

aov\_table3 = sm.stats.anova\_lm(mod3, type=2)

print(aov\_table3) #keep

# In[48]:

mod4 = ols('cnt ~ holiday', data = data).fit()

aov\_table4 = sm.stats.anova\_lm(mod4, type=2)

print(aov\_table4) #remove

# In[49]:

mod5 = ols('cnt ~ weekday', data = data).fit()

aov\_table5 = sm.stats.anova\_lm(mod5, type=2)

print(aov\_table5) #remove

# In[50]:

mod6 = ols('cnt ~ workingday', data = data).fit()

aov\_table6 = sm.stats.anova\_lm(mod6, type=2)

print(aov\_table6) #remove

# In[51]:

mod7 = ols('cnt ~ weathersit', data = data).fit()

aov\_table7 = sm.stats.anova\_lm(mod7, type=1)

print(aov\_table7) #keep

# In[52]:

#from anova and feature importance plotting of random forest, we decided to drop holiday and workingday

data = data.drop(['holiday', 'workingday'], axis=1)

# In[53]:

data.head()

#dataScale = data.copy()

#dataScale.head()

# In[54]:

####################################### feature Scaling ######################################

#Scaling for Continious variable

plt.figure(figsize=(14,4))

plt.subplot(2,4,1)

sns.distplot(data['temp'])

plt.title('temperature distribution')

plt.subplot(2,4,2)

sns.distplot(data['hum'])

plt.title('humidity distribution')

plt.subplot(2,4,3)

sns.distplot(data['windspeed'])

plt.title('windspeed distribution')

plt.subplot(2,4,4)

sns.distplot(data['cnt'])

plt.title('count distribution')

plt.tight\_layout()

#All our continuous variables are already normalized except the target variable which we prefer not to scale because its variation are spread quite widely and after scaling, the difference between the number is diminishing.

# In[55]:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Modeling \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

##Sampling: dividing Test and train data using sklearn

from sklearn.model\_selection import train\_test\_split,KFold, cross\_val\_score, cross\_val\_predict

#Random sample selection

train, test = train\_test\_split(data, test\_size = 0.20, random\_state = 100)

train.to\_csv("TrainFile\_BikeRenting.csv", index=False)

test.to\_csv("TestFile\_BikeRenting.csv", index=False)

data.shape, test.shape , train.shape

# In[56]:

from sklearn import metrics

#function to check performance

def performance(actual, predict):

print('MSE:', metrics.mean\_squared\_error(actual, predict))

print('RMSE:', np.sqrt(metrics.mean\_squared\_error(actual, predict)))

print('MAPE:',np.mean(np.abs((actual-predict)/actual))\*100)

print('R-Sq:', metrics.r2\_score(actual, predict))

# In[57]:

#1. ############################################ Decision Tree ############################

from sklearn.tree import DecisionTreeRegressor

#1. K-fold for cross validation of our model

#k\_fold = KFold(n\_splits = 10, shuffle=True, random\_state=101).get\_n\_splits(train.iloc[:,0:8])

#rmse = np.sqrt(cross\_val\_score(dt1, train.iloc[:,0:8], train.iloc[:,8], cv= k\_fold))

dt1 = DecisionTreeRegressor(max\_depth =6, random\_state=123).fit(train.iloc[:,0:8],train.iloc[:,8])

prediction\_dt1 = dt1.predict(test.iloc[:,0:8])

#error matrix

performance(test.iloc[:,8],prediction\_dt1)

print(' ')

print('Perdicted Vs Actual value: ')

prediction\_dt1[6], test.iloc[6,8]

# In[58]:

#2. ############################################ Random Forest ########################################################

from sklearn.ensemble import RandomForestRegressor

rf1 = RandomForestRegressor(n\_estimators = 100, random\_state = 126).fit(train.iloc[:,0:8],train.iloc[:,8])

prediction\_rf1 = rf1.predict(test.iloc[:,0:8])

performance(test.iloc[:,8],prediction\_rf1)

print(' ')

print('Perdicted Vs Actual value: ')

prediction\_rf1[1], test.iloc[1,8]

# In[59]:

#3. ############################################ Linear Regression ############################

from sklearn.linear\_model import LinearRegression

ln\_model = LinearRegression().fit(train.iloc[:,0:8],train.iloc[:,8])

prediction\_lr1 = ln\_model.predict(test.iloc[:,0:8])

performance(test.iloc[:,8],prediction\_lr1)

print(' ')

print('Perdicted Vs Actual value: ')

prediction\_lr1[1], test.iloc[1,8]

# In[60]:

#Ploting to understand the spread of predicted data.

plt.figure(figsize=(14,4))

plt.subplot(1,3,1)

plt.title('Decision Tree predictions')

plt.scatter(test.iloc[:,8] , prediction\_dt1)

plt.xlabel('Actual values')

plt.ylabel('Predicted values')

plt.subplot(1,3,2)

plt.title('Random Forest predictions')

plt.scatter(test.iloc[:,8] , prediction\_rf1)

plt.xlabel('Actual values')

plt.ylabel('Predicted values')

plt.subplot(1,3,3)

plt.title('Linear Regression Predictions')

plt.scatter(test.iloc[:,8] , prediction\_lr1)

plt.xlabel('Actual values')

plt.ylabel('Predicted values')

plt.tight\_layout()

# In[62]:

#Hence we are finalising random forest

#saving results

test['DT Predictions'] = prediction\_dt1

test['RF Predictions'] = prediction\_rf1

test['LR Prediction'] = prediction\_lr1

train.to\_csv("TrainFile1onwhichModelistrained.csv", index=False)

test.to\_csv("TestFile1withresultsfromModel.csv", index=False)

# In[ ]:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **\*\*\*\*\*\*\*END OF DOCUMENT**