**Predicting Bike Rental Count**

**Preeti Chauhan**

**30 November 2018**

1. INTRODUCTION
   1. **Problem Statement**

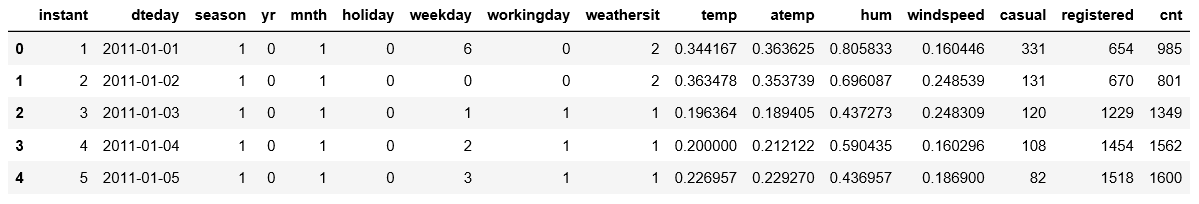
The objective of this Case is to predict of bike rental count on daily basis on the environmental and seasonal settings.

The aim of the whole project is to predict the total bike rental count for a day based on environment and seasonal settings so that the bike rental company can be ready to serve the customer and meet the demand proactively.

* 1. **Data**

Data provided with the problem is **day.csv**.

Let’s have a look at sample of dataset:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Expected Values** | **Value Meaning** |  |
| Instant | Record index | Unique numeric identifier |  |  |
| Dteday | Date | Any valid Date |  |  |
| Season | Season | 1  2  3 | 1:springer, 2:summer, 3:fall, 4:winter |  |
| yr | Year | 0  1 | 0: 2011,  1:2012 |  |
| Mnth | Month | 1  2  3  .  .  .  12 | 1: January,  2: Feb,  3: March,  .  .  .  12: Dec |  |
| Holiday | Indicates weather day is holiday or not(extracted from Holiday Schedule) | 0  1 | 0: Not a Holiday  1: Holiday |  |
| Weekday | Day of the week | 0  1  .  .  6 | 0: Saturday  1: Sunday  .  .  6: Friday |  |
| workingday | If day is neither weekend nor holiday is 1, otherwise is 0 | 0  1 | 0: Not a workingday  1: workingday |  |
| Weathersit | Weather situation(extracted fromFreemeteo) | 1  2  3  4 | 1: Clear, Few clouds, Partly cloudy, Partly cloudy  2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist  3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds  4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog |  |
| Temp | Normalized temperature in Celsius. | The values are derived via (t-t\_min)/(t\_max-t\_min), t\_min=-8, t\_max=+39 (only in hourly scale) |  |  |
| Atemp | Normalized feeling temperature in Celsius. | The values are derived via (t-t\_min)/(t\_max- t\_min), t\_min=-16, t\_max=+50 (only in hourly scale) |  |  |
| Hum | Normalized humidity | The values are divided to 100 (max) windspeed: Normalized wind speed. The values are divided to 67 (max) |  |  |
| Casual | count of casual users |  |  |  |
| Registered | count of registered users |  |  |  |
| Cnt | count of total rental bikes including both casual and registered |  |  |  |

Here Target variable(dependent variable), which needs to be predicted is: **‘cnt’**

And predictors(independent variables), on basis of which target needs to be predicted are : **(dteday,season,yr,mnth,holiday,weekday,workingday,weathersit,temp,atemp,hum)**

Let’s explore further to use this data to build a predictive model.

1. PROJECT APPROACH

To execute this project, I’ll go along with tried and tested CRISP- DM process

* 1. **CRISP-DM process**

CRISP – DM process

2

Data understanding

1

Business Understanding

Data preparation

Data

3

Deployment

6

Modelling

4

Evaluation

5

Above process, shows phases of data science projects, each phase having its importance.

* 1. **CRISP - DM In a nutshell**
* **Get a decent understanding of business domain.** Its very important to relate to problem before you figure out the solution . e.g. , if you want to build a model for predicting stock prices, you should have a basic understanding of stock market.
* **Get a precise Business Objective.** Sometimes you get directly from client, else you have to derive and get it verified.
* **Get the Data to analyze** and find stats, patterns, existing trends. This will help you toward a better judgement.
* **Categorize the problem statement.** We need to put the problem in a problem category(all the problem categories are described in answer to Question 3). \this is very important as specific machine learning algorithm is used for specific category.

**Eg.** Unsupervised learning problem will use any of the unsupervised learning algorithm, which may not be suitable for a prediction problem.

* **Know the prerequisites of the selected data models** and see if the data given to you is meeting that requirement. eg. If for a banking problem, decision tree machine learning model is selected to address a problem statement, then we should know that Decision tree does not allow missing values as input.
* **You have to prepare the data as per selected model.** eg. If you have selected decision tree then, you know that it doesn’t take missing values, then you need to deal with missing value before feeding it to the model.
* **Build the Model(s)** using the machine learning algo(s) you have selected.
* **Compare and Evaluate the models**: Compare various models on their performance matrices and choose the best one fitting your requirements.
* Deploy the selected Model to prod.

1. DETAILED PROJECT IMPLEMENTATION

Below are the different phases of the implementation of project

* 1. **Define the Project ROADMAP**

I have sketched the plan to implement the project in different phases, using CRISP-DM process. Some steps in CDM process have been left out which are out of the scope of this project report, like deployment.

The whole project is divided in 7 phases (and further subphases). Below are the phases defined.

* Define and categorize problem statement
* Gather the data
* Prepare data for consumption
* Perform Exploratory Data Analysis
* Modelling
* Evaluate and compare Model performances and choose the best model
* Hypertune the selected model
* Produce sample output with tuned model
  1. **Implement the Project ROADMAP**

As per the above roadmap, let’s start the project, exploring each phase.

* + 1. **Categorize Problem**

The problem statement is “to predict of bike rental count on daily basis on the environmental and seasonal settings”

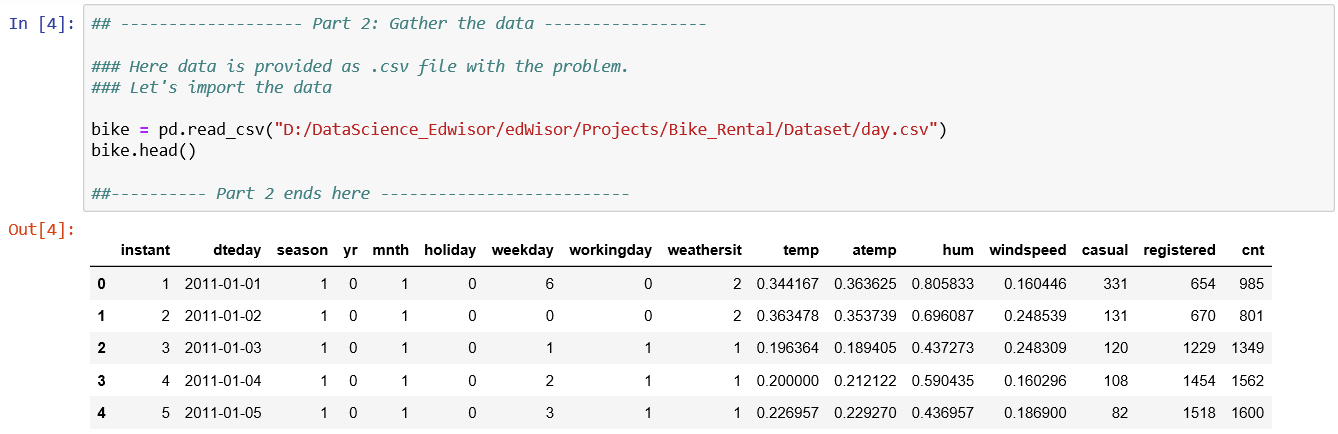
It is evident from the problem statement above that based on the predictors values (input, both numerical and categorical), the output dependent value(numerical) needs to be predicted.

So, clearly this problem is of category – **Supervised Machine Learning Regression Problem.**

* + 1. **Gather the Data**

The data is given to us on platter. Simply import the data and have a glance.

Note: Data is well described in section 1.2 . Use this section to understand the data better.



**Achievements of this step:**

* + The data set required for analysis has been imported.

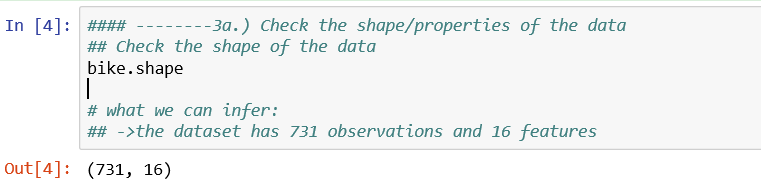
**Learnings of this step:**

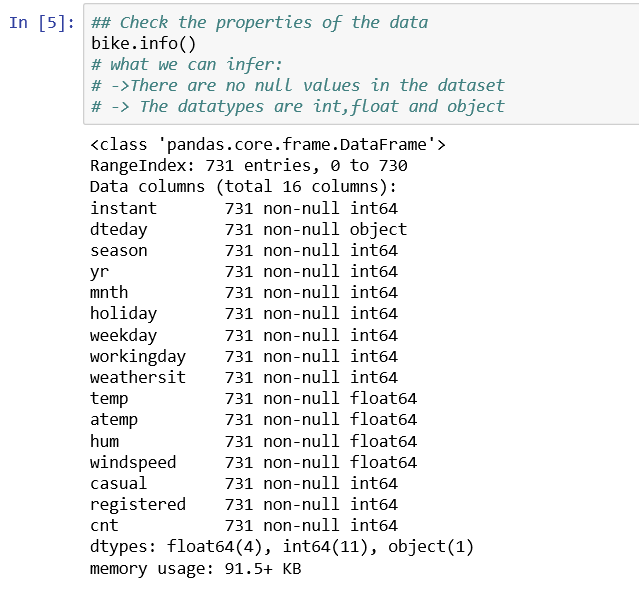
* + There is only 1 data file which has 16 columns
    1. **Prepare Data**

Next step is to prepare the data for consumption. In this case, the compiled data set is given to us in 1 file. We do not need to join different sources to prepare data for the analysis.

What we need to do here is more of data cleaning activity and make it ready for EDA and modelling. I performed following steps to achieve this.

* + - 1. **Check the shape/properties of the data**





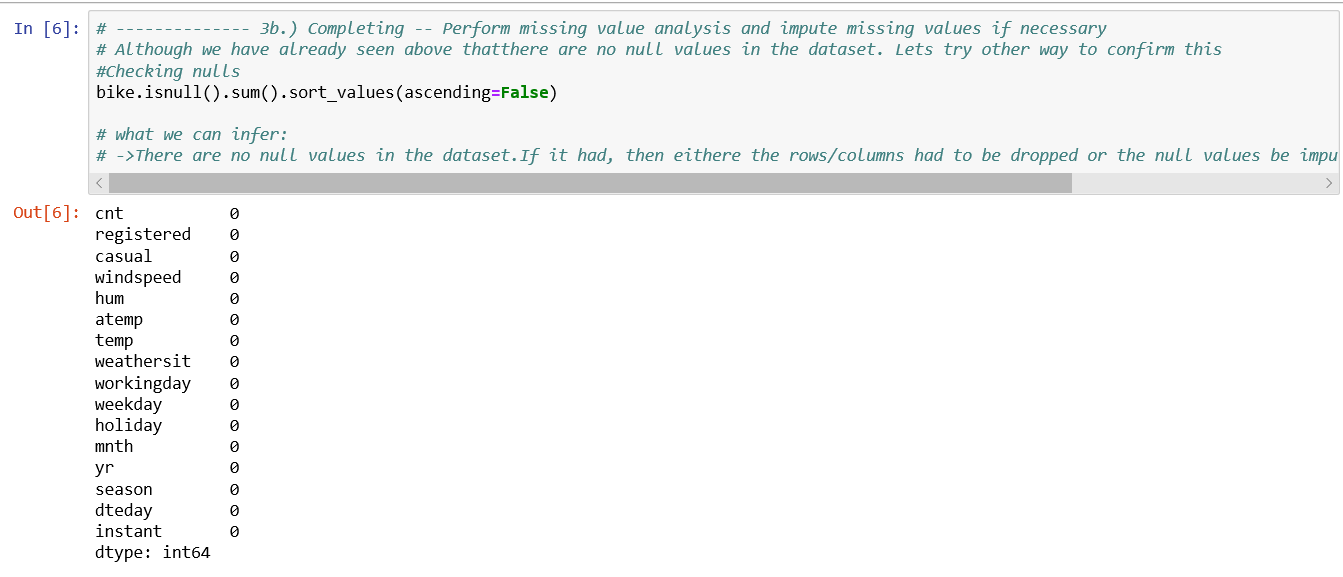
**Achievements of this step:**

* + We have checked the size and basis information of the data

**Learnings of this step:**

* + There are 16 features and 731 observations in the dataset
  + Int64(11) ,float64(4) and object(1) datatypes are used in this dataset.
  + None of the columns in dataset has nulls
    - 1. **Completing**

Perform missing value analysis and impute missing values if necessary



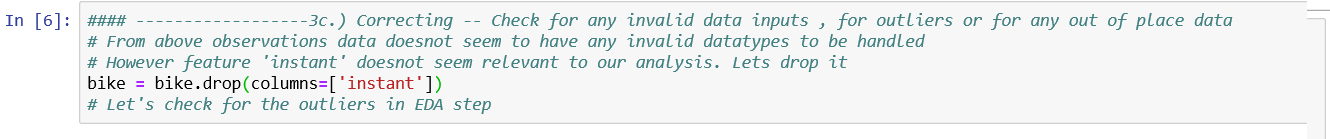
**Achievements of this step:**

* + To confirm if the dataset has any null values

**Learnings of this step:**

* + None of the columns have any null values.
  + No need for missing value handling/imputation.
    - 1. **Correcting**

Check for any invalid data inputs , for outliers or for any out of place data



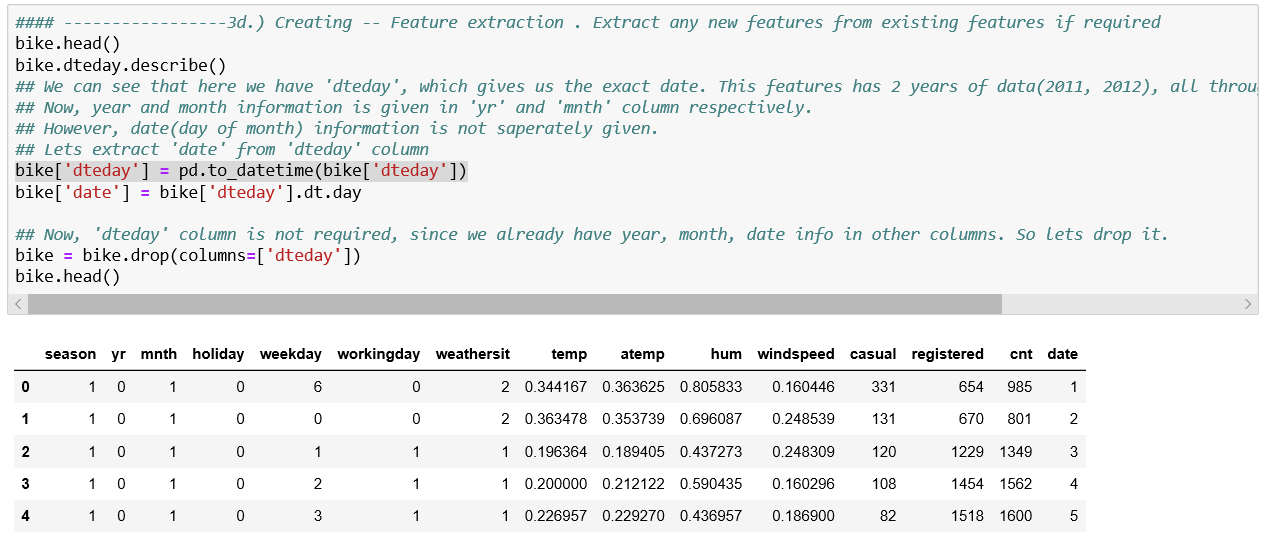
**Achievements of this step:**

* + Since ‘instant’ column does not help in our analysis, dropped it from the dataset.

**Learnings of this step:**

* + Data seems to be correct and relevant, further analysis will be done under EDA.
  + Outliers analysis and correction(if required) will be done under EDA
    - 1. **Creating**

**Feature extraction**. Extract any new features from existing features if required.



**Achievements of this step:**

* + Extracted feature ‘date’ from ‘dteday’.
  + Dropped feature ‘dteday’ since all the info we can get from this feature like(year, month and date) are already in the dataset as ‘yr’, ‘mnth’ and ‘date’ respectively. So this feature is not important for our analysis.

**Learnings of this step:**

* + ‘dteday’ is not important for our analysis, since same info is available in other features.
    - 1. **Converting**

Converting data to proper formats.

Columns like ‘yr’, ‘season’ are imported as numeric columns. However, these are categorical in nature. So, these needs to be converted to ‘categorical’ datatypes.

It is important to convert the categorical values to category as ‘numeric’ and ‘categories’ have different features. E.g.

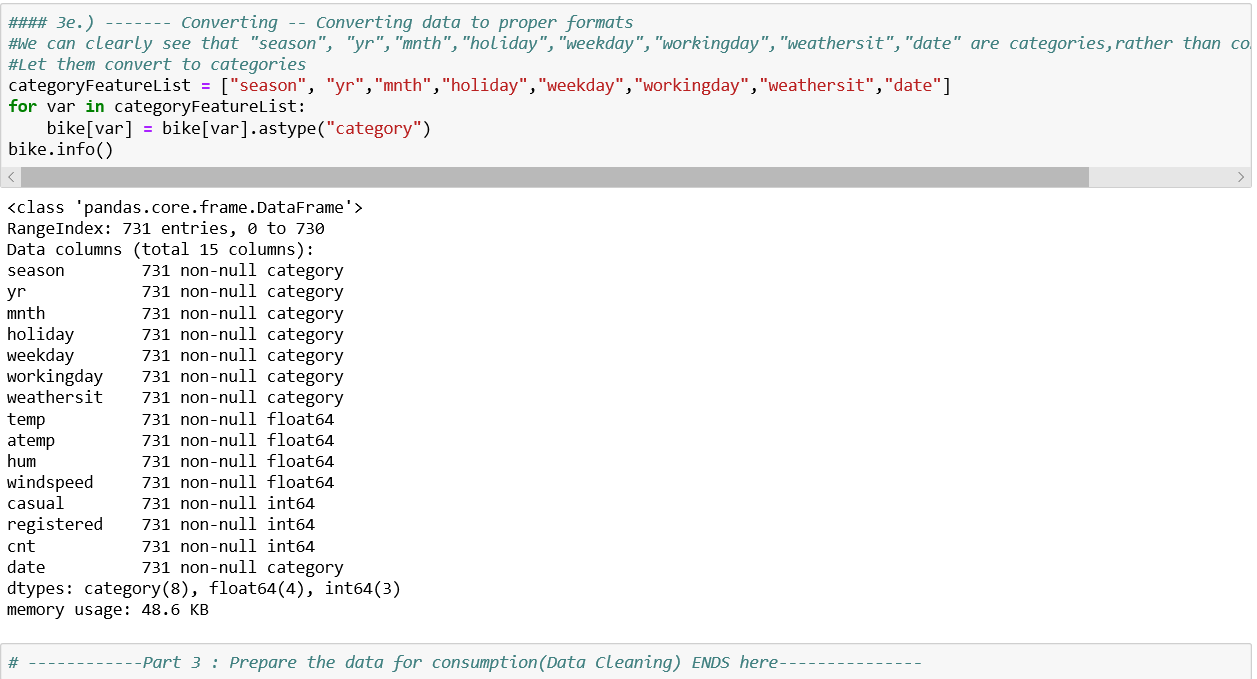
X\_category = [1,2,3,4]

X\_numeric = [1,2,3,4]

Here X\_category and X\_numeric may seem same but they are not, X\_numeric has order attached to values like 1<2 etc.

But X\_category does not, 1, 2,34 does not have any order, they are simply categories.

Feeding wrong datatypes to model may affect the results.



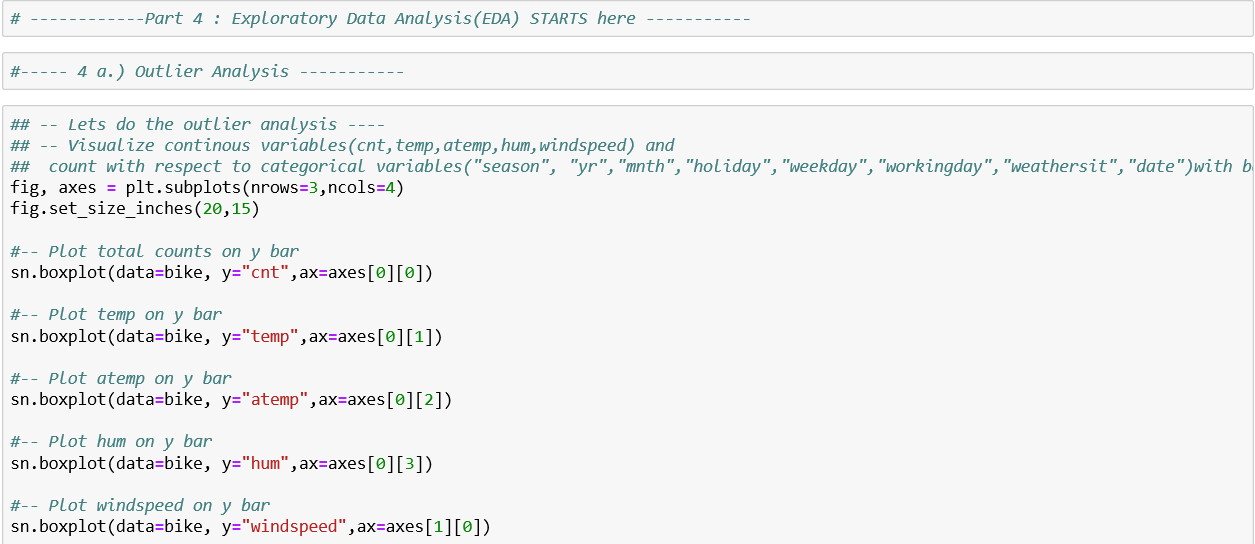
**Achievements of this step:**

* + Converted "season", "yr","mnth","holiday","weekday","workingday","weathersit","date" to ‘**category’**.

**Learnings of this step:**

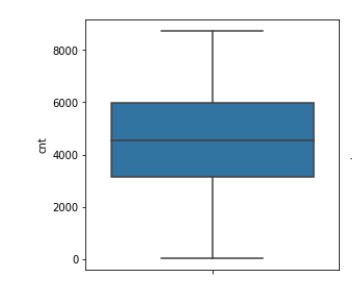
* + NA

* + 1. **Perform EDA**
       1. **Outlier Analysis using Boxplots**

****

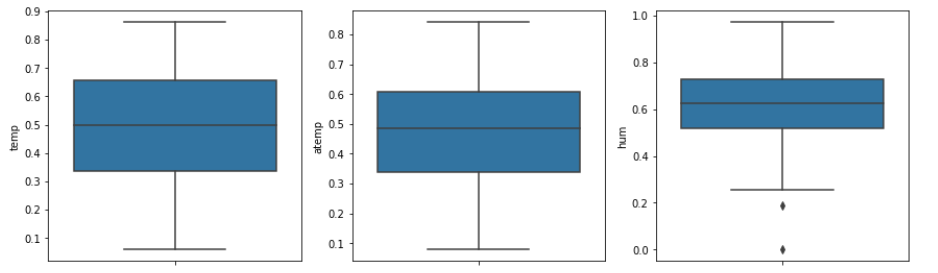
****

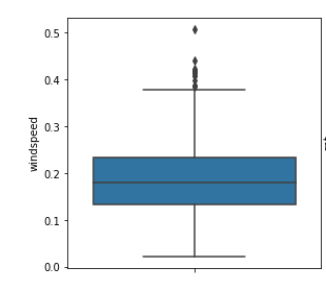
**Boxplot for rented bikes count(‘cnt’)**

****

* There are no outliers for target variable ‘cnt’. All values lies with 3IQR range of mean.

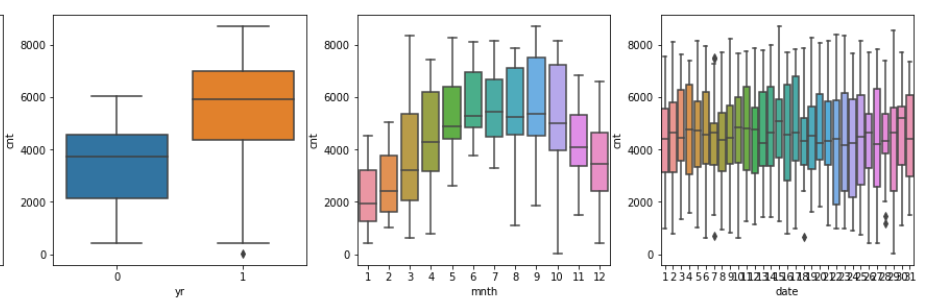
**Boxplots for numeric predictors():**

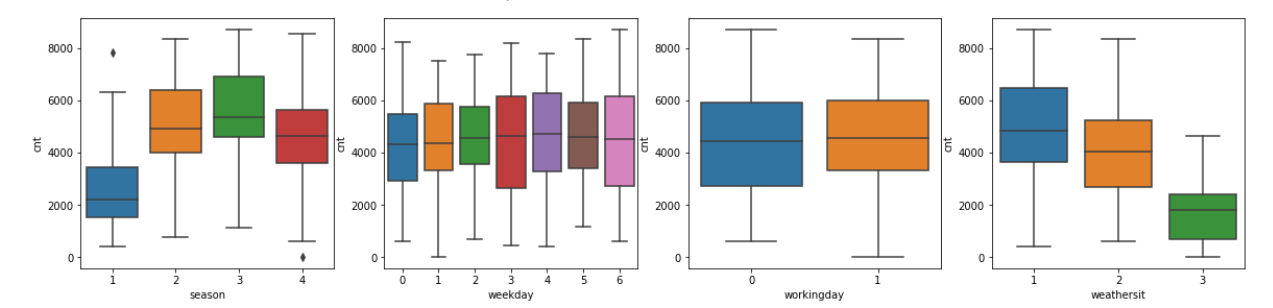
****

****

* There are couple of outliers for humidity
* There are few outliers for windspeed

**Boxplots for categorical predictors against count():**

****

****

* There is one outlier for year 2011, where the rental count is extremely low. This could be the result of some extreme weather condition
* From month boxplot, we can see there are no outliers and also can be seen that average rented bikes count is higher between 5th -10th month, which makes sense due to favourable weather conditions
* Boxplot for date has few outliers, which seems to be common noise.
* Season data is pretty much in range, only 2 outliers.
* No other categorical features have any outliers.
* Let’s wait till EDA to remove the outliers

**Achievements of this step:**

* + Identified outliers of various features

**Learnings of this step:**

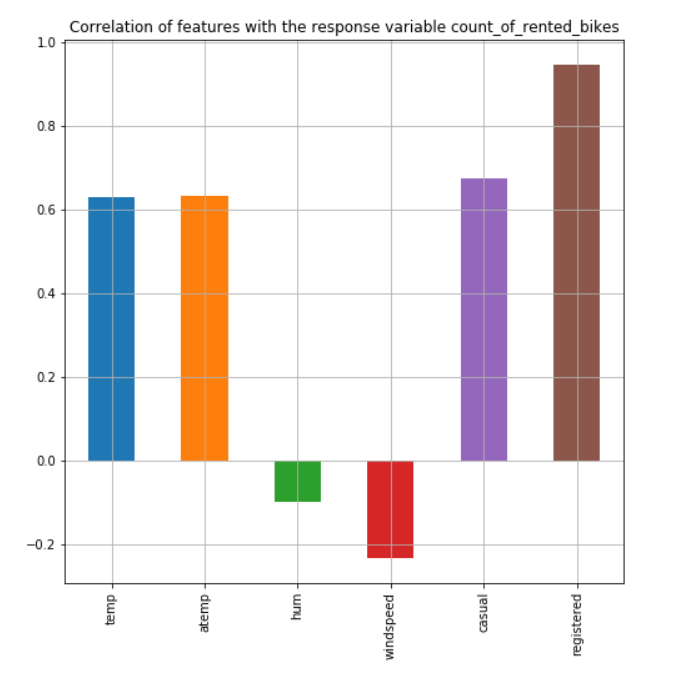
* + Most of the data set looks clean. Not many outliers found.
  + Only little noise is there in database
  + Not removing outliers as of now, as these seems to be valid entries
  + Removing little outliers will generalize the model.
    - 1. **Analysis of Numerical Features**

1. **Correlation Analysis**

We’ll analyse the

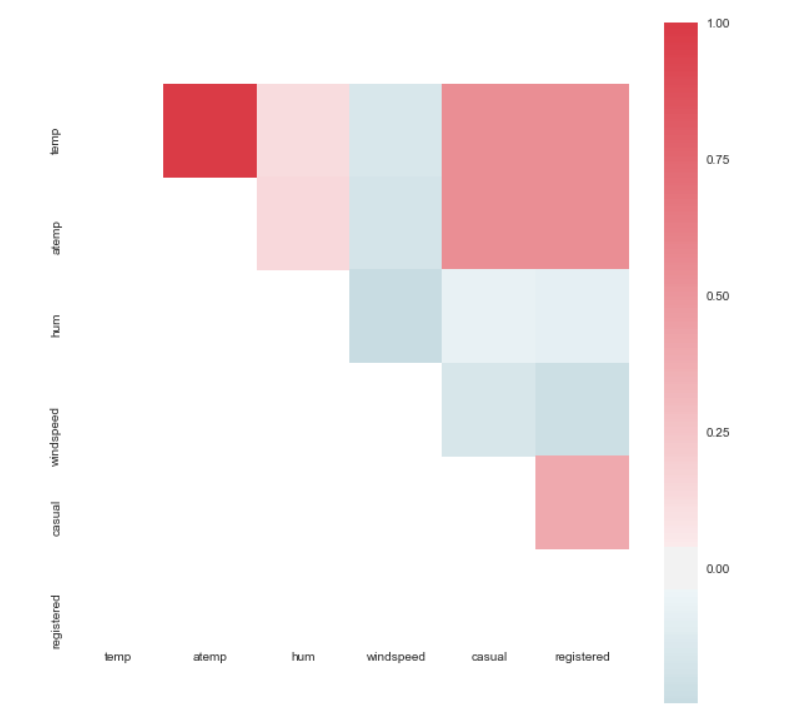
* + - relationship of all numeric independent variables with target variable(cnt)
    - relationship of all numeric variable among themselves (to detect multi collinearity)

1. **Barplot for Correlation of features with the response variable**



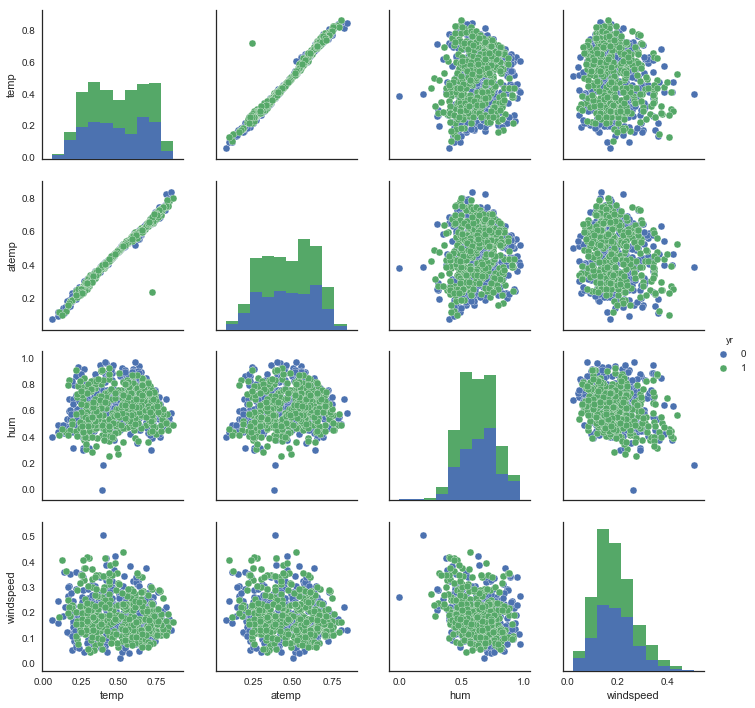
* Though this plot, it is evident that ‘temp’ and ‘atemp’ has good correlation with ‘cnt’ which is good.
* ‘registered’ and ‘casual’ has very good relation with ‘cnt’. However, we’ll not consider them for our analysis as they are the leaky variables.
* Correlation of ‘humidity’ with ‘cnt’ is less. However, lets keep this variable for now.

1. **Heatmap for Correlation of features with each other**

****

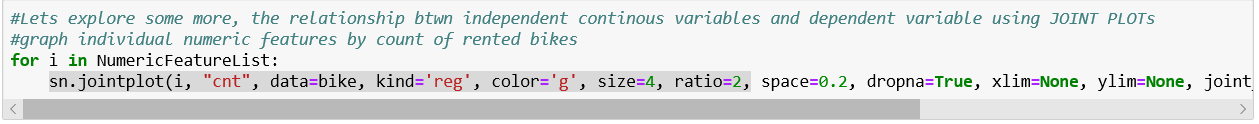
* It is evident that ‘temp’ and ‘atemp’ are highly correlated and that **multicollinearity** exists.
* To remove multicollinearity, drop one either one of ‘temp’ or ‘atemp’ needs to be dropped.

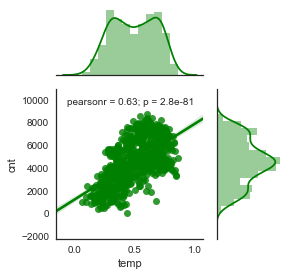
1. **Explore using pairtplots**

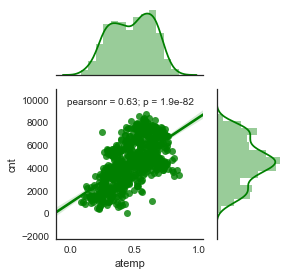


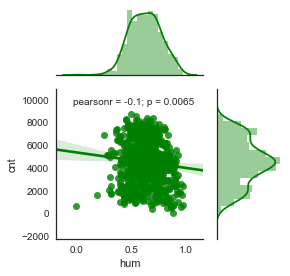
* Clearly seen above, ‘atemp’ and ‘temp’ are highly related. Hence, we need to drop one of them

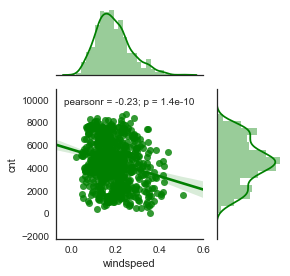
1. **Explore through Jointplots**











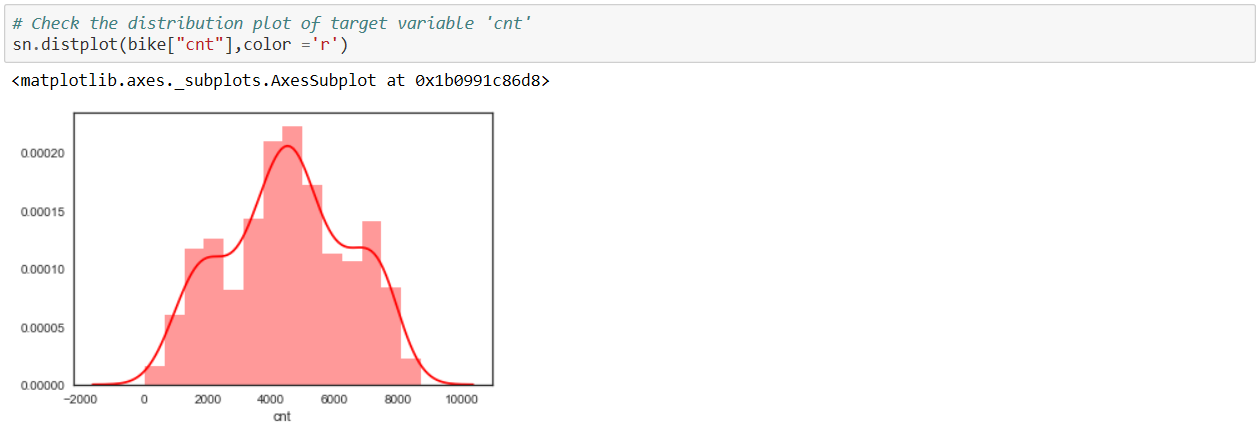
* We can see that ‘temp’ and ‘atemp’ has same pearson coefficient(0.63). However, p(temp) > p(atemp).
* So, we can safely drop ‘temp’ from the dataset.

1. **Distribution of target variable through Distplot**

For many models, it is important that the target variable follows normal distribution.

If that is not the case, many a times, we apply some techniques (like taking log) to convert the distribution to normal.

Let explore the distribution of our target variable:



* Clearly, target variable has almost normal distribution, no need to apply any technique here.

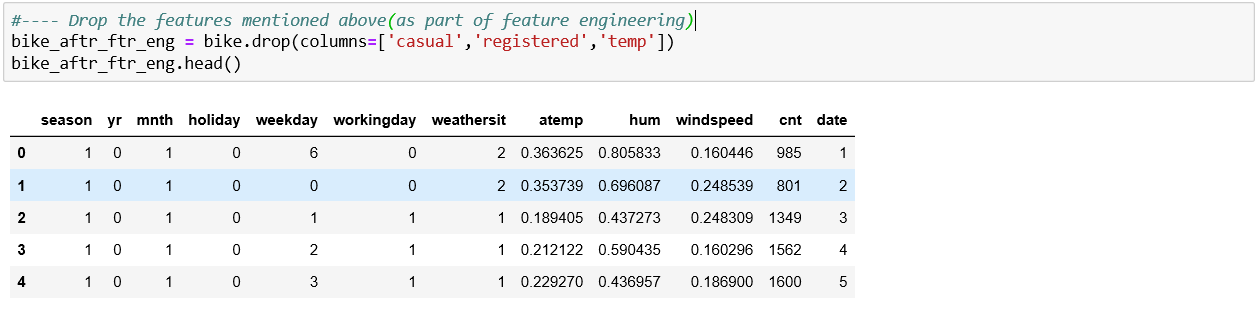
**Achievements of this step:**

* + Feature Engineering on continuous features implemented

**Learnings of this step:**

* + Target variable 'cnt' is almost normally distributed, which is a good thing.
  + From correlation with dependent variable cnt, we can see that 'casual','registered' are very highly correlated to cnt. These are actually 'leak variables'. **'casual','registered' needs to be dropped from the dataset.**
  + 'hum' has low correlation with 'cnt'. For ow, let’s keep it.
  + atemp and temp has good correlation with 'cnt'
  + From heatmap, we can see that atemp and temp are highly correlated. So, we need to drop 1 to remove multicollinearity.
  + Since, as seen from jointplot,p(atemp) < p(temp), **we can drop 'temp' and retain 'atemp' in the dataset**

**Drop the features as mentioned above, as part of feature engineering/selection**

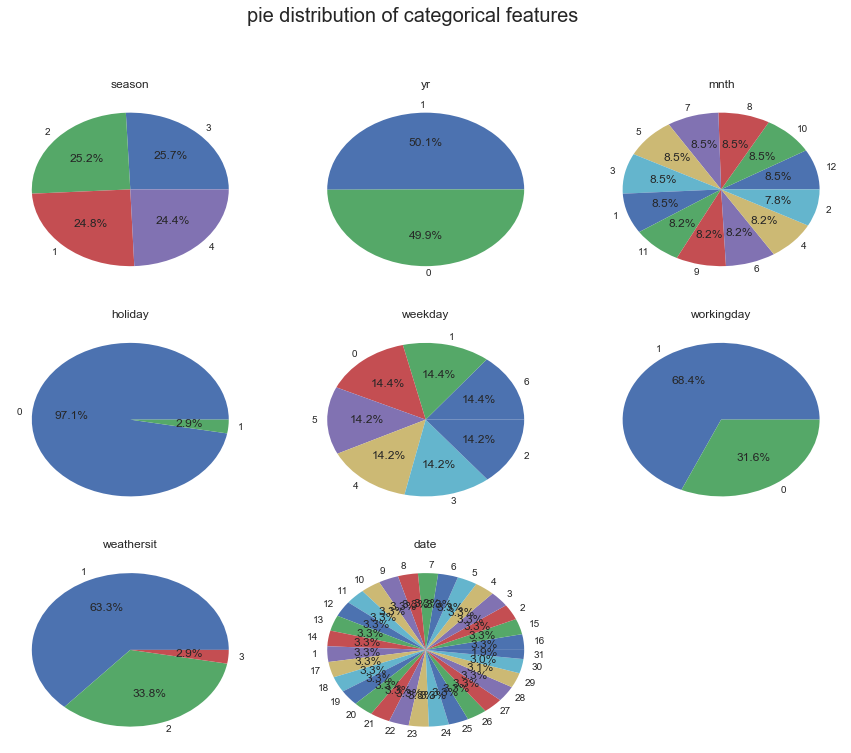


* + - 1. **Exploratory Analysis of Categorical Features**

Let’s explore categorical features now through various graphs.

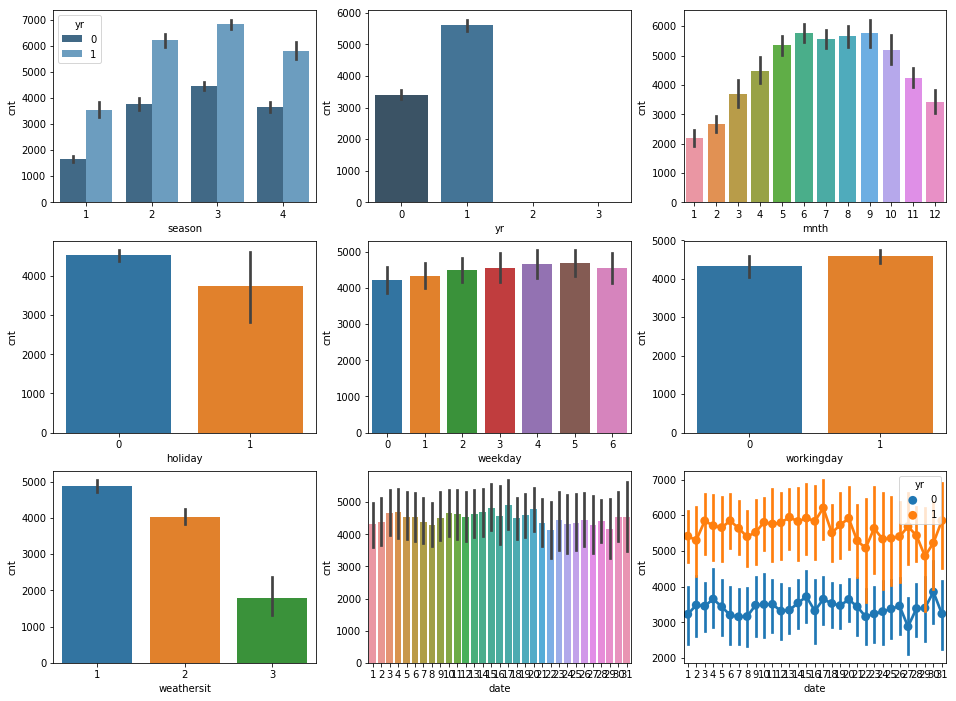
1. **Pie Distribution of Categorical Features**

****



* Most of the categorical variables are uniformally distributed, except 'holiday','weathersit','workingday'
* This makes sense for 'weathersit', as extreme weather is rare and hence %percentage of extreme weather in whole dataset is low
* This makes sense for 'holiday', as number of holidays are less in comparison to working days
* This makes sense for 'workingday' for the same reason as above
* So, categorical data seems to be pretty much uniformly distributed

1. **Barplot for Count Vs. Categorical features**



Let’s see how these categorical variables individually effects the count of rented bikes

* Does 'yr' affect count of rented bikes
* YES. the count has an upward trend wrt year
* Does 'season' affect count of rented bikes
  + YES, it seems ppl rent more bikes during season 3 and 2, i.e. highest in fall and summer and less in winter and springs. This makes sense as weather is good to ride during summer and fall.
* Does 'month' affect count of rented bikes
  + YES.ppl are likely to rent bikes more btwn the months May- October and lowest in month of Jan,Feb and Dec(in that order). This again makes sense, as this trend is in sync with favourable weather conditions
* Does 'holiday' affect count of rented bikes
  + YES. ppl rent more bikes on non-holiday than holiday. It makes sense as bikers who commute to work/school will be less on holiday.
* Does 'weekday' affect count of rented bikes
  + To some extent Yes. ppl seems to rent lesser bikes on Sat/ Sun. ie. over the weekend. Again makes sense as school and offices are closed on weekend.Monday also has lesser count of rented bikes. It may be possible the ppl visit to other places/cities over weekend and travel back in car on Monday, istead of renting bikes.
* Does 'season' affect count of rented bikes
  + Most definately YES. noone rented bike on extreme weather(season=4). ppl rent maximum bikes during a clear day (weathersit=1)
* Does 'date' affect count of rented bikes
  + Well there is no set trends. It seems to be random. Let explore bit more of it over the 12 months using pairplot
  + From the pointplot over 2 years, it looks like count distribution by ‘date’ has little similarity on trends for both years. So, let’s leave the ‘date’.

**Achievements of this step:**

* + Feature Engineering on categorical features implemented

**Learnings of this step:**

* + Categorical features are mostly uniformly distributed.
  + Most of the features effects the count of the rented bikes
  + Date does not affect the count very strongly. However, we’ll keep it for now.
    1. **Modelling**
       1. **Choosing the ML Models**

Now, the exploratory analysis is done, we need to decide on the machine learning algos we’ll use to build the predictive models.

I am going to use 3 ML algo to build 3 different models and later compare them to decide on the best model. Below are the 3 ML algo I have chosen to build model on and compare:

* + - Linear Regression Model
    - Random Forest Model
    - Gradient Boosting Model

Before going to the predictive models we are going to use, let’s understand few concepts:

* + - * **Ensemble methods:**

Ensemble methods combine several decision trees classifiers to produce better predictive performance than a single decision tree classifier. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner, thus increasing the accuracy of the model.

* + - * **Bagging**

Bagging is a technique that is used when the goal is to reduce the variance of a decision tree classifier. **Here the objective is to create several subsets of data from training sample chosen randomly with replacement.** Each collection of subset data is used to train their decision trees. As a result, we get an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree classifier.

* + - * **Boosting**

Boosting is a technique that is used to create a collection of predictors. In this technique, **learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors**. Consecutive trees (random sample) are fit and at every step, the goal is to improve the accuracy from the prior tree. When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly. This process converts weak learners into better performing model.

Now, lets get on to the models we are going to build.

1. **Linear Regression Model:**

Linear regression is one of the most commonly used predictive modelling techniques. The aim of linear regression is to find a mathematical equation for a continuous response variable Y as a function of one or more X variable(s). So that you can use this regression model to predict the Y when only the X is known.

**Y= a1 X1 + a2X2 + ……………………………. + anXn**

1. **Random Forest Model:**

**Random Forest is an ensemble machine learning algorithm that uses ‘bagging’ technique**.

Random forests are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees.[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1)[[2]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1998-2) Random decision forests correct for decision trees' habit of [overfitting](https://en.wikipedia.org/wiki/Overfitting) to their [training set](https://en.wikipedia.org/wiki/Test_set).

1. **Gradient Boosting Model:**

**Gradient Boost is an ensemble machine learning algorithm that uses ‘boosting’ technique**.

Whereas [random forests](http://uc-r.github.io/random_forests) build an ensemble of deep independent trees, GBMs build an ensemble of shallow and weak successive trees with each tree learning and improving on the previous. When combined, these many weak successive trees produce a powerful “committee” that are often hard to beat with other algorithms.

All the above model, I am building with feature engineered dataset.

* + - 1. **Choosing the Performance Measures for the Models**

We are working on regression problem, so I think the best performance matrix could be

* + - R-SQAURED
    - RMSE
    - MSE
    - MAE

We’ll measure above measures for all the 3 models to compare them.

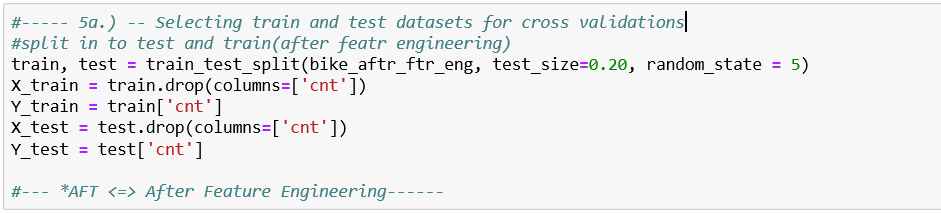
The performance dataframe will look something like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **R-Squared** | **MSE** | **RMSE** | **MAE** |
| **Linear Regression** |  |  |  |  |
| **Random Forest** |  |  |  |  |
| **Gradient Boosting** |  |  |  |  |

* + - 1. **Building the ML predictive Models**

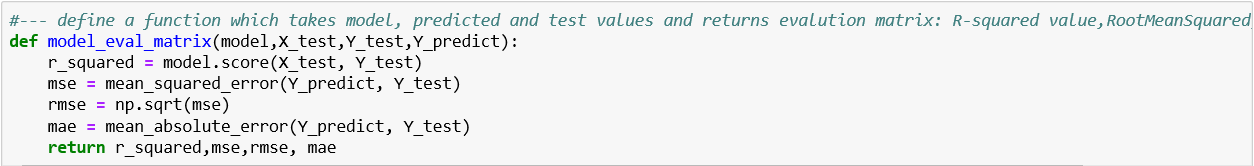
Let’s start building models. Follow the following steps:

1. **Divide Datasets for Cross Validation**



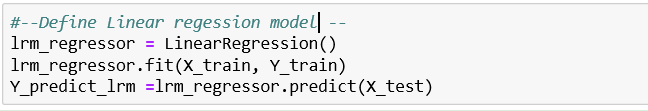
80% of data is used for training while 20% is reserved for testing(cross validating)

1. **Create function to Measure Performance**

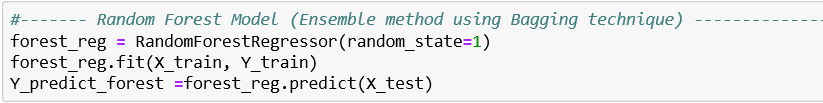


I have created a function, which takes the model, X\_test, Y\_test,Y\_predict values of any model and returns r\_squared,mse,rmse and mae. This is a generic function which would return the performance parameters of any model.

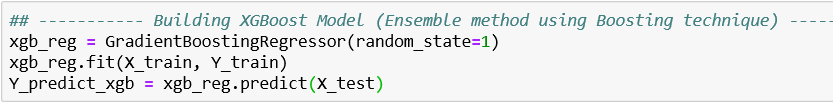
1. **Build, Train & Predict with Linear Regression Model**



1. **Build, Train & Predict with Random Forest Model**



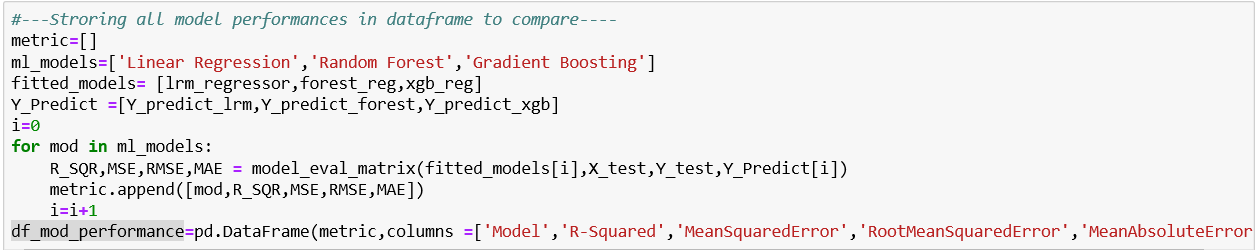
1. **Build, Train & Predict with XGBoost Model**



Now, all the 3 models are built , lets get the performance scores for the models and store in a dataframe

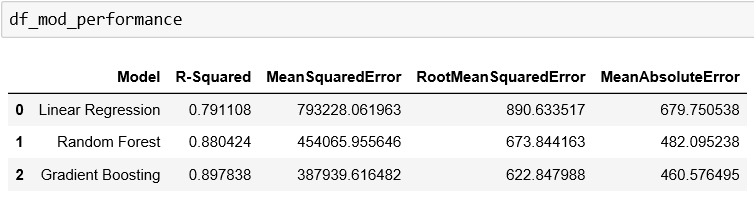
* + - 1. **Building the Performance Matrix dataframe**

Now when all the models are built, let’s call the function to measure performance for each model and store in a dataframe.



* + 1. **Model Evaluation and Comparision**

Results are as below:

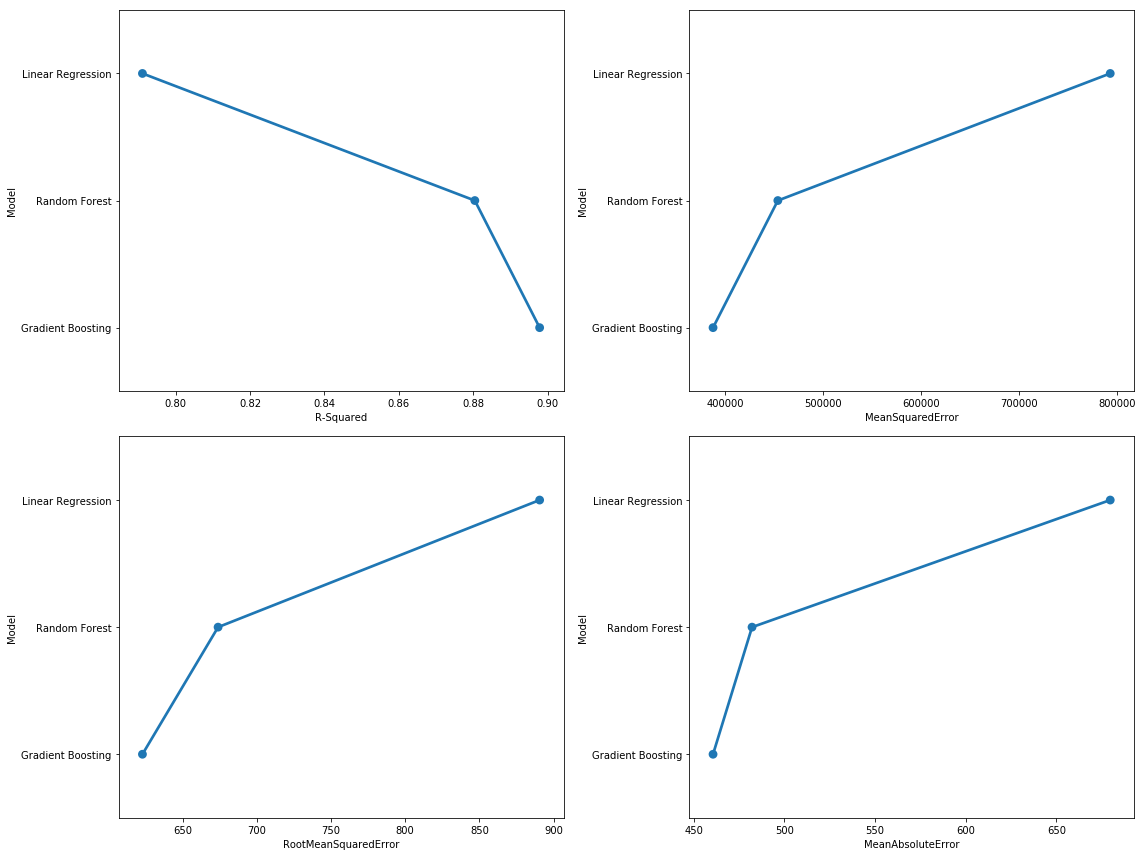
****

Linear Regression model has lowest R-squared(goodness of fit) and highest MeanAbsoluteErorr.

While **XGBoost has highest R-squared(goodness of fit) and lowest MeanAbsoluteErorr.**

Let’s visualize performance of various models:

1. **Visualize Performance Results with Pointplots**



1. Visualize Performance Results with Barplots

It is evident from the results above that XGBoost Model gives the best performance of all. Hence, **XGBOOST model is the model our final model.** Let’s create sample output file using this model.

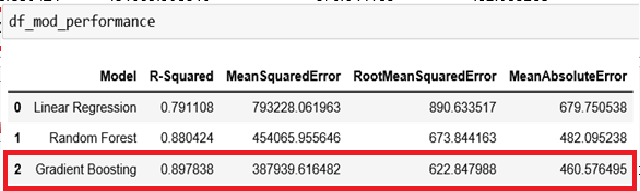
* + 1. **Hypertuning the Selected Model**

Now, Gradient Boosting is the final model, parameter hypertuning can be performed on the model to find the best parameters which will give the maximum performance.

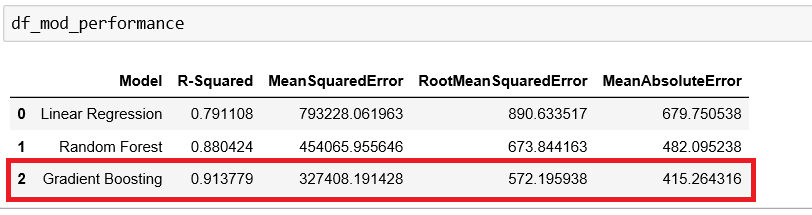
Functions like **GRIDSearchCV** from **GridSearch** library of python can be used for this.

However, I tried here simple approach of ‘hit and trial’, where I changed parameter few times and found a set which gave me maximum performance.

Performance of GradientBoosting model **before** parameter tuning:



Performance of GradientBoosting model **after** parameter tuning:



As evident from above data, after tuning the parameters, we gained R-Squared and MSE, RMSE , MAE got reduced.

**Final verdict: we are going to use Gradient Boosting Algorithm as our final model with the following parameters**:

loss='ls',

learning\_rate=0.1,

n\_estimators=300,

subsample=1.0,

criterion='friedman\_mse',

min\_samples\_split=2,

min\_samples\_leaf=1,

min\_weight\_fraction\_leaf=0.0,

max\_depth=3,

min\_impurity\_decrease=0.0,

min\_impurity\_split=None,

init=None,

random\_state=1,

max\_features=None,

alpha=0.9,

verbose=0,

max\_leaf\_nodes=100,

warm\_start=False,

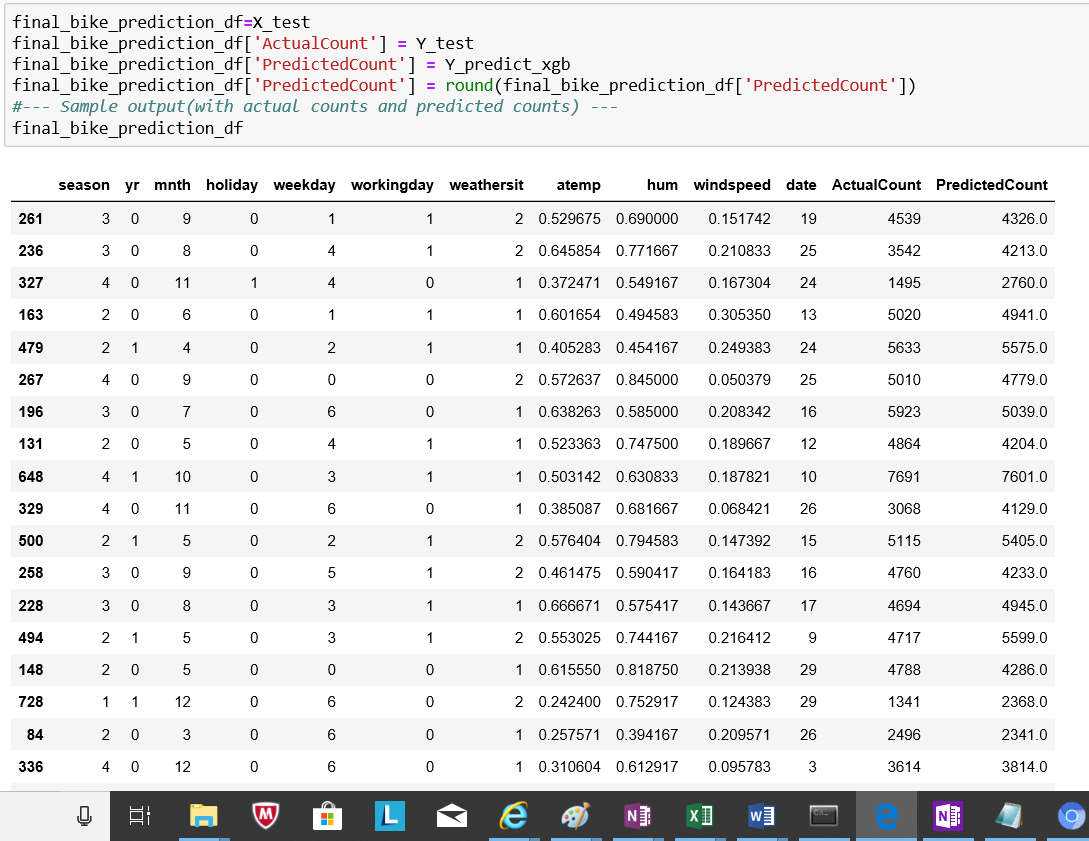
presort='auto'

* + 1. **Generating Sample Output**

Generate a sample output file with actual count and model predicted count.

**Note: The actual file is submitted as a separate file(bike\_rental\_sample\_output.csv)**

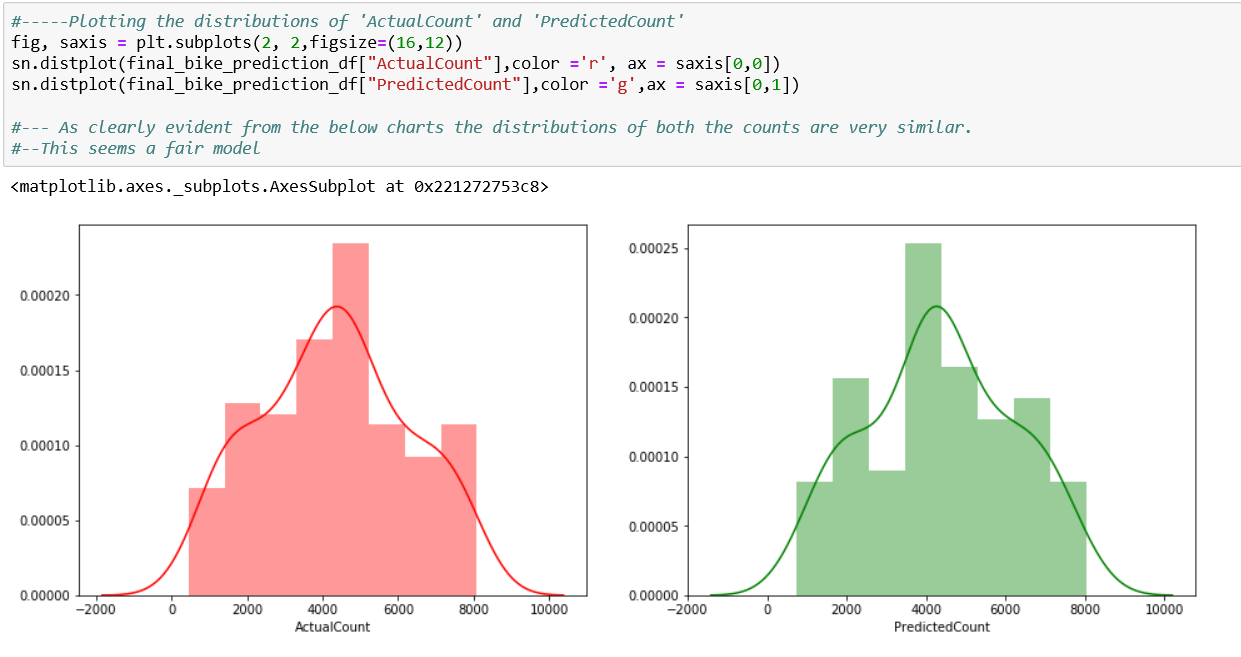
**Sample output file : **

****

1. CONCLUSION

So, we have build a good model with around 91.4% of R-Squared value(Goodness of fit) using X Gradient Boosting ML algorithm(which uses boosting technique)

Let’s see the distribution of actual values and predicted values.



Both predicted values and actual counts have almost similar distribution. Well, that’s a good end note for our model.

This model is ready for deployment now.

1. PYTHON CODE

Python code is attached here: 

Python code is also submitted as separate file bike\_rental.ipynb

1. RCODE

R code is attached here: 

R code is also submitted as separate file bike\_rental.R