Bike Renting

Ataf Ahmed

04.01.2019

Chapter 1

Introduction

1.1 Problem Statement

The objective of this case is the prediction of bike rental count daily based on the environmental and seasonal settings.

1.2 Data

We need to develop a machine learning model which predicts the count of bikes which is going to be hired by the travellers on the situation of environmental and seasonal weather conditions.

An overview of the dataset:-

(i) Our dataset contains 731 observations and 16 feautures.
(ii) Variable Description:-

S.no	Variable Name	Description
1.	instant	Record index
2.	dteday	Date
3.	season	Season
4.	yr	Year
5.	mnth	Month of the year
6.	holiday	If a particular day is a holiday or not
7.	workingday	If a particular day is a workingday or not
8.	weekday	Day of the week
9.	weathersit	Weather condition
10.	temp	Normalized temperature in Celsius
11.	atemp	Normalized feeling temperature in Celsius
12.	hum	Humidity
13.	windspeed	Windspeed
14.	casual	Number of casual users of the record
15.	registered	Number of registered users of the record
16.	cnt	Number of total rentals of the record

The variables ***casual*** and ***registered*** are our target variables and their sum constitutes ***cnt*** which is primary target variable.

The rest of the 13 variables are our predictors or independent variables.

Chapter 2

Methodology

2.1 Preprocessing

The first step towards building any predictive model is Exploratory Data Analysis. We need to look at our data , its size,shape,the type of variables in it, the distributions of various variables and their respective relationship with one another.

Thus after loading the data we look at its shape and the data types of the variables.

- data = pd.read_csv(“day.csv)
- data.shape
- data.dtypes.

We found that our data has a shape of 731 x 16 and all the variables in our dataset are of the numeric type .

Plotting the histogram of all the features to get an overview of the distribution of the variables.

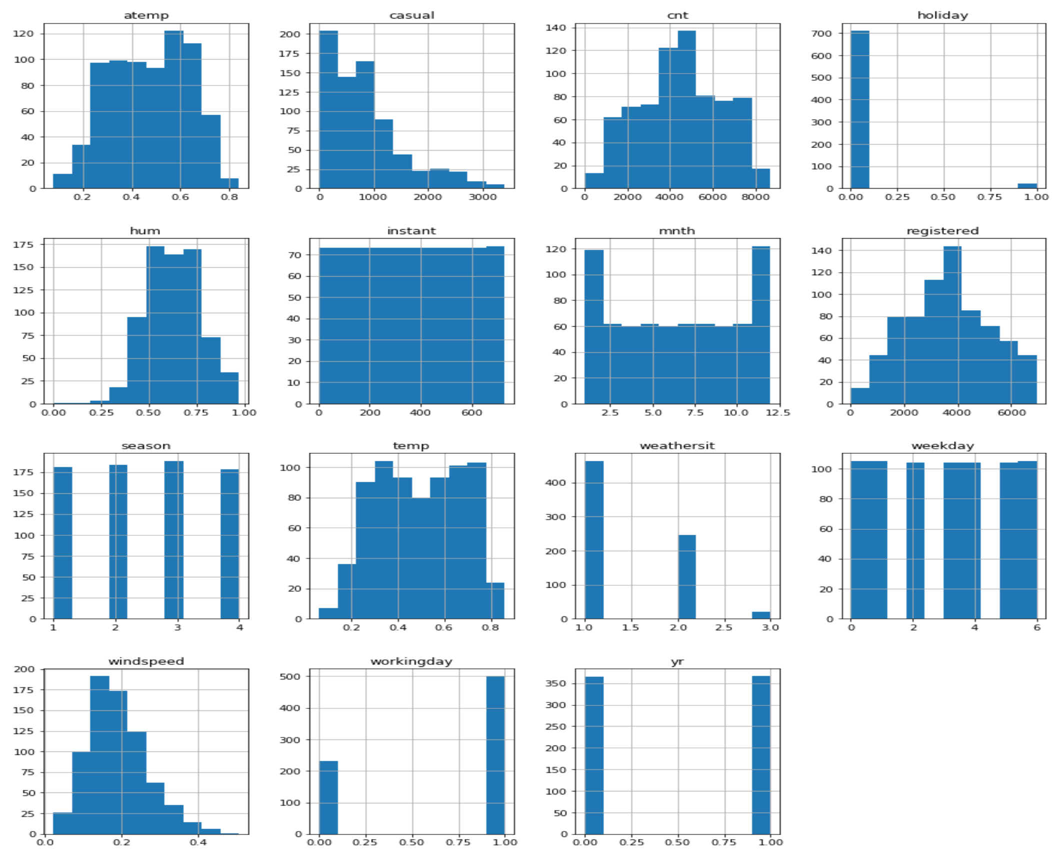


Fig2.1 Histogram of all the features.

Looking at fig2.1 we can conclude that some features in our dataset are discrete while others are continuous. The discrete variables should be converted to categorical variables(or factors) for better analysis. And since the continuous predictor variables in our data were already normalized their distributions do not show any significant skewness.

2.1.1 Converting to Categorical

Some of the variables need to be converted to factors.

- to_cat = ['yr', 'workingday' , 'weekday' , 'holiday' , 'mnth' , 'season' , 'weathersit']
- for i in to_cat:
    - data[i] = data[i].astype("category")

2.1.2 Outlier Analysis

Drawing the boxplot of all the numerical variables to look for outliers.

Fig 2.2 shows that some of our variables have outliers.

After taking a closer look at the outliers we concluded that the outliers are present in three of the variables- 'windspeed', 'hum' and 'casual'.

The total number of outliers in the data is 59, and it is better to remove this small percentage of outliers from our data as they can add bias to our training model.

Although outliers can be treated in different ways as well and one way is to impute them with new values using mean,median or K-nn imputation method. But since the number of outliers is not too much in our data deleting them would be a better approach than imputation.

For doing the same the approach used is as follows:-

- cnames = ["temp","atemp","hum","windspeed","casual","registered","cnt"]

```
for i in cnames:

        q75 , q25 = np.percentile(data.loc[:,i],[75,25])
        iqr = q75  -  q25
        min = q25 - (iqr*1.5)
        max = q75 + (iqr*1.5)
        print(min)
        print(max)
        data = data.drop(data[data.loc[:,i] < min].index)
        data = data.drop(data[data.loc[:,i] > max].index)
```

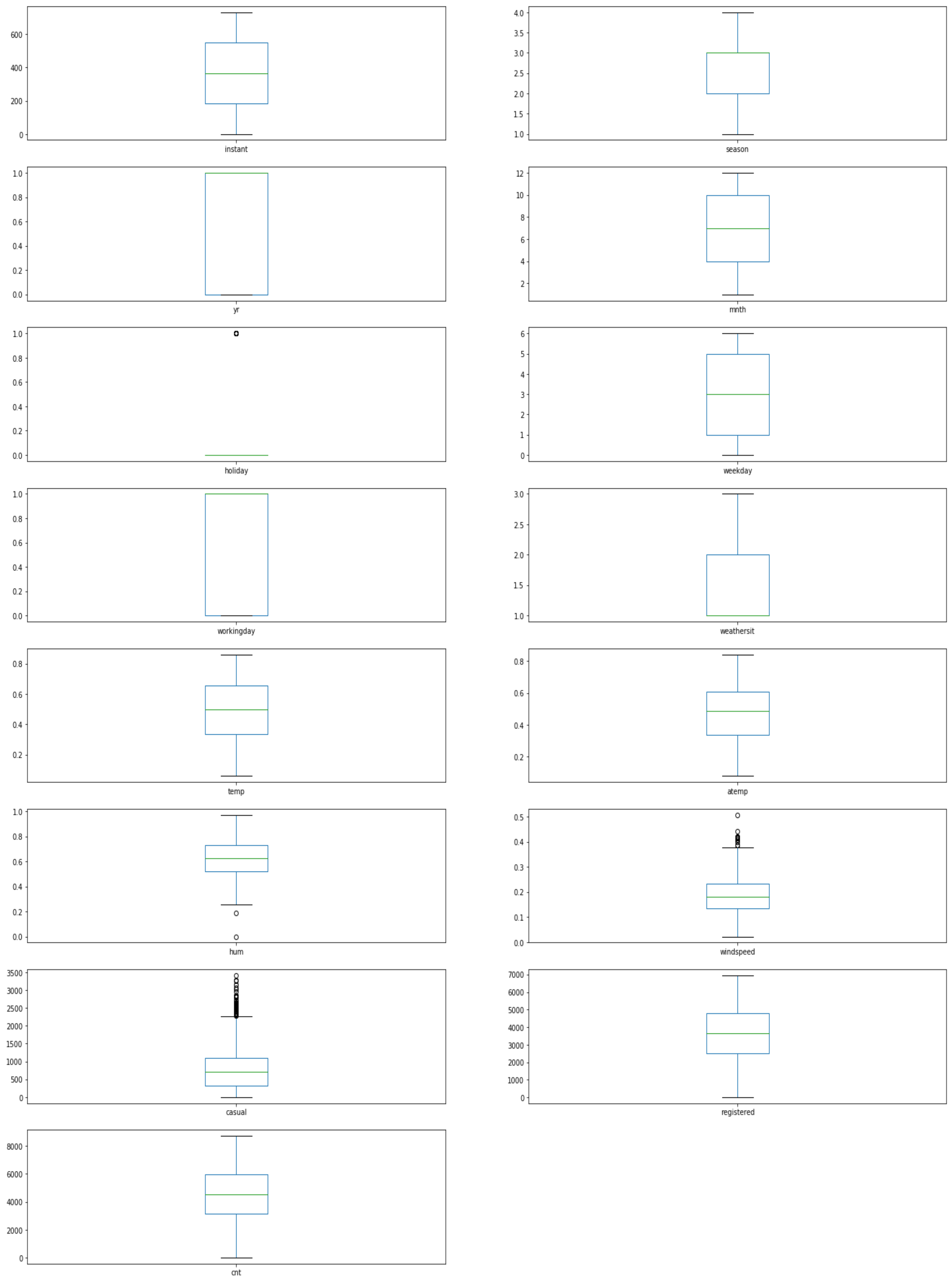


Fig 2.2 Boxplot for outliers.

2.1.3 Missing Values Analysis

Next we look if our dataset has any missing values present in it as they can heavily influence our training model. Missing values if present need to be handled in an appropriate way for building a good model.

- ➢ data.isna().sum()

Our data returned zero missing values so we can proceed further with our analysis.

2.2 Feature Selection

Selecting which features to feed in our model is an important step in model building. We want to train our model with the features that influence our target variable and help in the prediction. Features carrying no or little importance would only add to the complexity of our model. Moreover features which are highly dependent on each other also add to the complexity of the model and make our models weak. Hence we want to select the independent variables such that they have an influence on the target variable and have little influence on one another.

The preliminary step in our feature selection would be creating visuals of the numerical and categorical variables in our dataset with the target variables to better understand the patterns and how they are influencing our dependent variables.

2.2.1 Visualisations

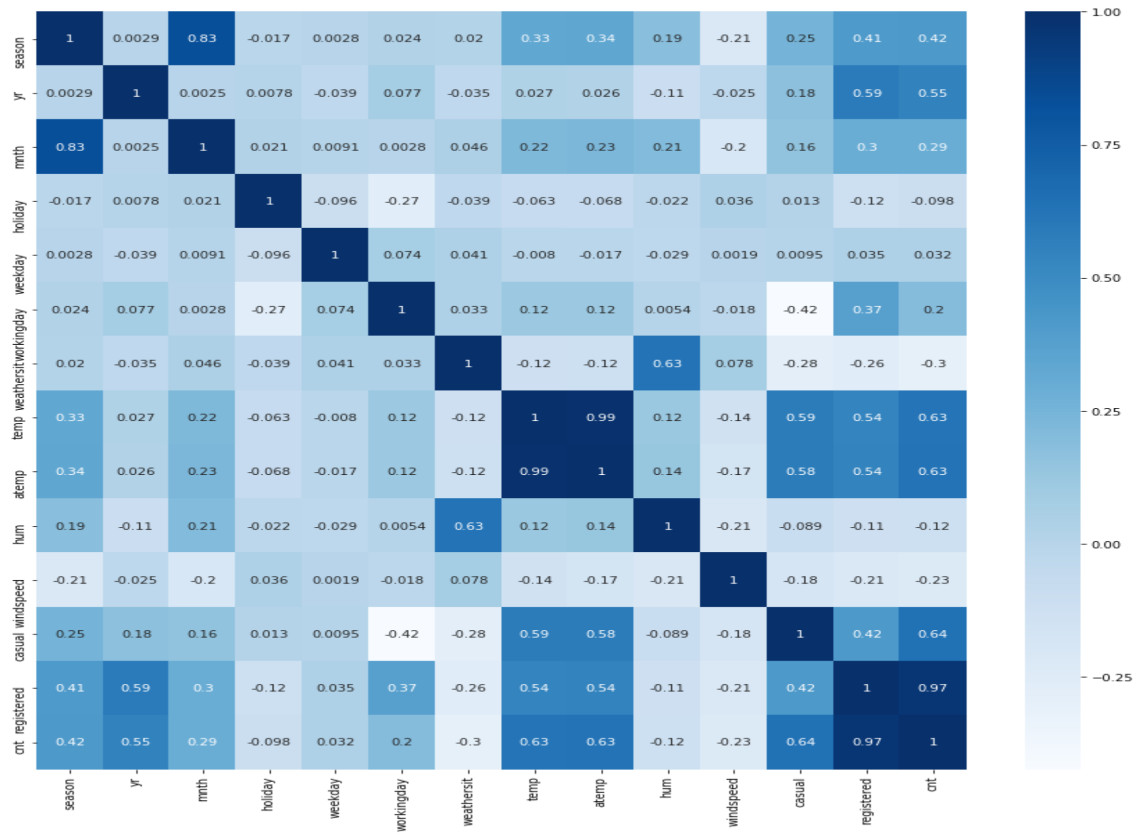


Fig 2.3 Correlation Heat map

Fig 2.3 shows a correlation heatmap of all the variables in our dataset.

The heatmap was drawn using python's seaborn library.

From the heatmap a few inferences can be drawn:-

1. Temperature has the highest positive correlation on count.
2. Season and weather are moderately positively correlated on count.
3. Humidity and weather condition show a strong positive correlation.
4. Holiday and weekday show very little effect on count.
5. Temperature and Feeling temperature are very highly correlated.

Our next analysis takes us to comprehending how the continuous variables (windspeed, humidity, temp) affect the casual and registered users and the overall count as well.

- Windspeed v/s casual/registered/count

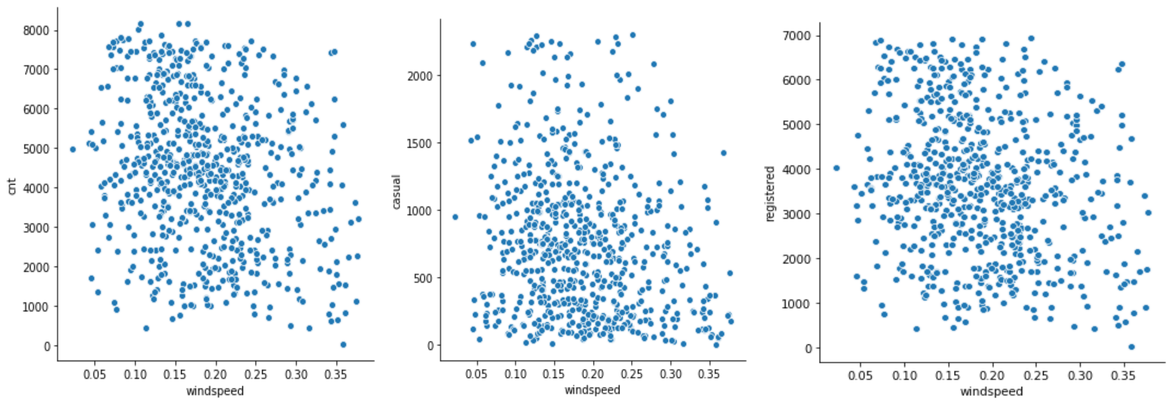


Fig2.4 Scatterplot of Windspeed with casual,registered and count.

It can be noticed that the speed of the wind is not influencing too much as to how many users rent a bike.

- Humidity v/s casual/registered/count

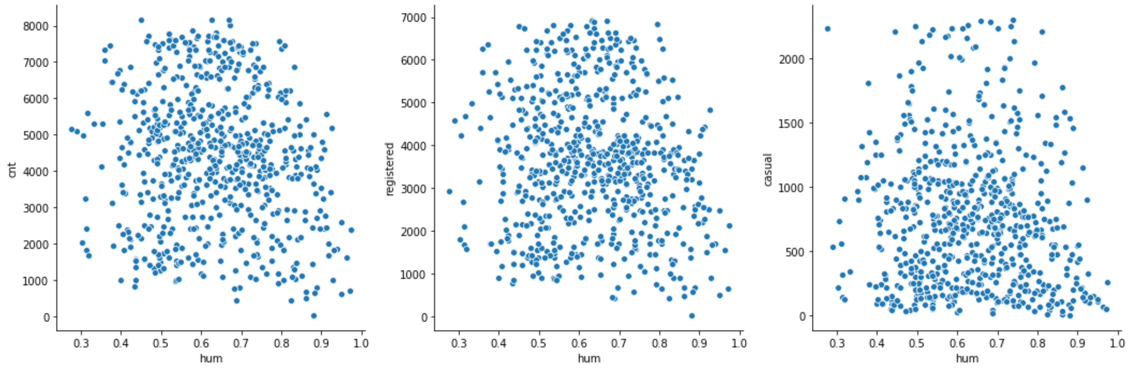


Fig2.5 Scatterplot of humidity with casual, registered and count.

Moderate values of humidity show more demand among users. Very high values of humidity generally have lesser demand especially by casual users.

➢ Temperature v/s casual/registered/count

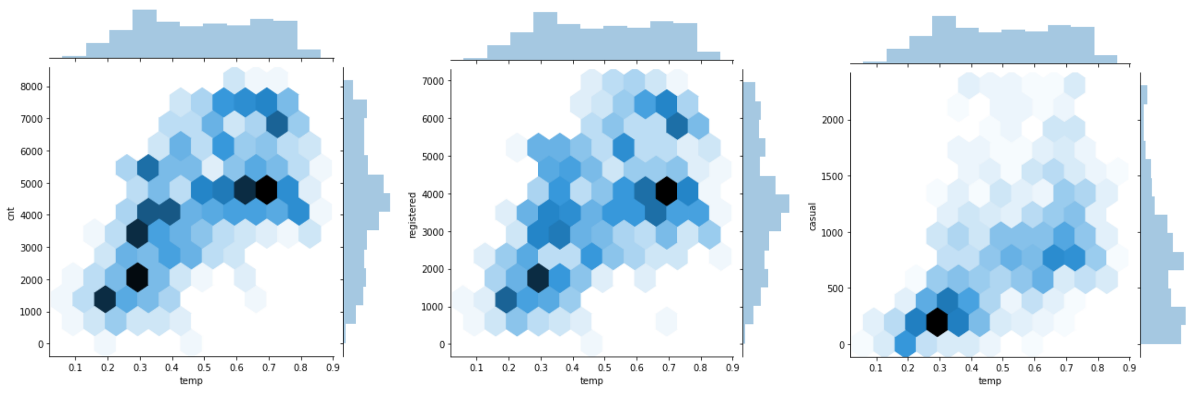


Fig 2.6 Hexagonal Jointplot of temperature with casual, registered and count.

Temperature shows some interesting interactions with the demand of rentals. It is seen that casual users do not generally prefer low temperatures ,while registered users and the total count seem to have a close to linear relationship with temperature.

Since the casual and registered users is affected by working day we build another scatter plot with workingday on it as well.

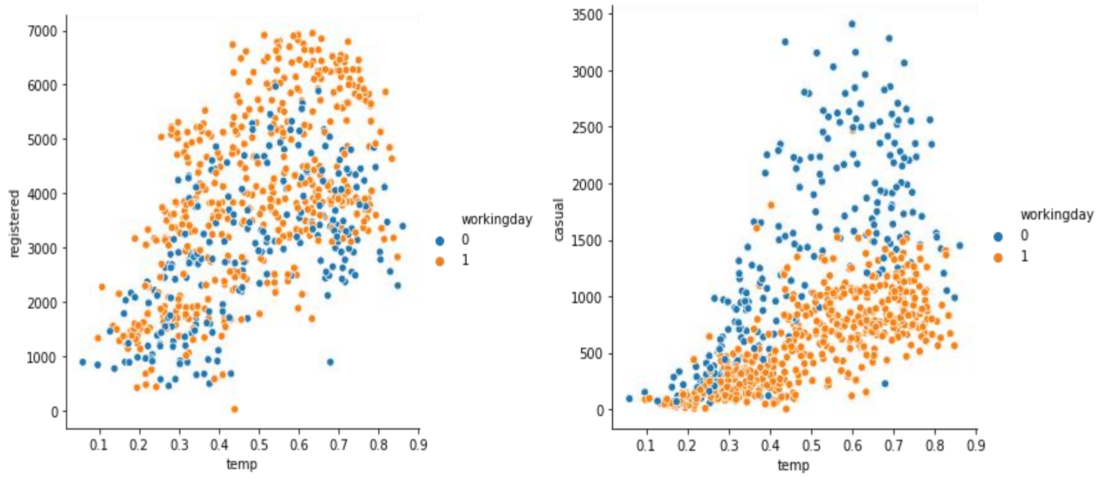


Fig 2.7 Scatterplot of registered and casual users against temperature.

It can be clearly seen that that while temperature shows a positive linear increase in the demand of both casual and registered users, casual users tend to be much higher on non-working days for the same value of temperature.

Registered users understandably are much higher on working days for the same temperature values.

2.2.2 Plotting With Categorical Variables

The next step in our analysis takes us to analysing the categorical variables in our data with plots to better understand the underlying trend in our data.

We have used python's seaborn library for our visuals.

- Year

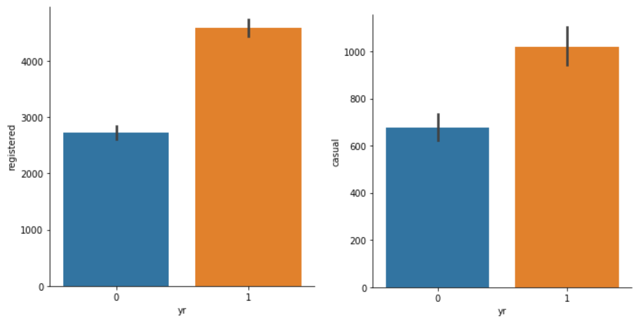


Fig 2.8 Barplot of year with registered and casual users

0 corresponds to the year 2011 and 1 to 2012.
2012 has seen a rise of almost two times in the demand of rentals in both the casual and registered users, and the rise may continue in the years to come.

- Season
  Seasons - 1:spring, 2:summer, 3:fall, 4:winter

Weather condition- 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
4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

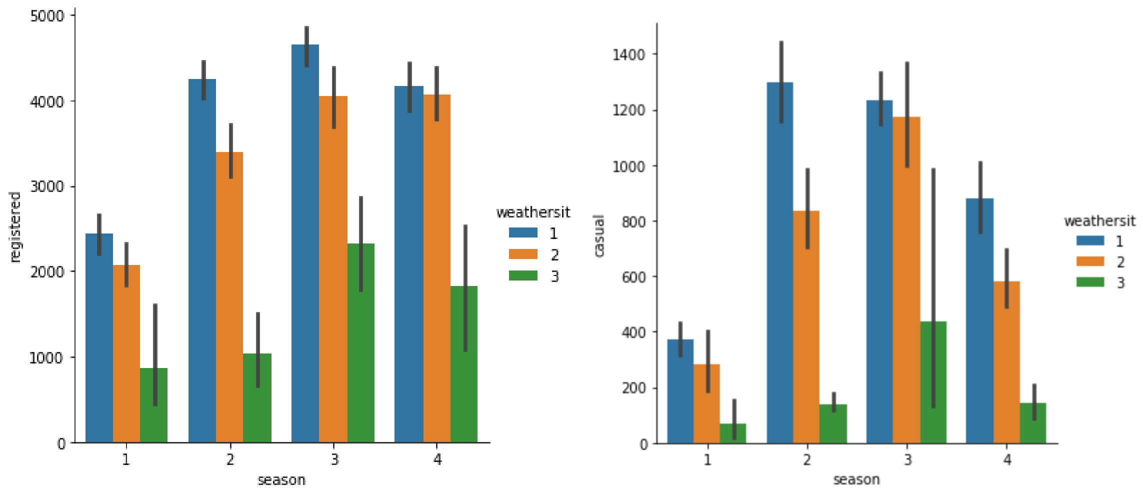


Fig 2.9 Barplot of registered and casual users with season and weather condition.

Summer and fall have the highest demand in both the casual and registered users while spring shows the minimum count of rentals. Registered users show a high demand in winters as well unlike casual users. The weather conditions 1 & 2 show the most demand across all the seasons while condition 3 has very less demand and weather condition 4 is does not have any instance.

- Workingday

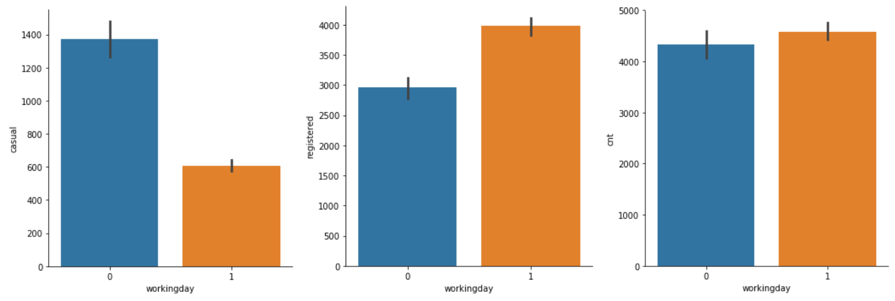


Fig 2.10 Barplot of casual and registered users with workingday

Understandably working days have a surge of registered users and vice-versa for casual users. However it is interesting to observe that the overall count on working and non working days is almost similar.

- Month

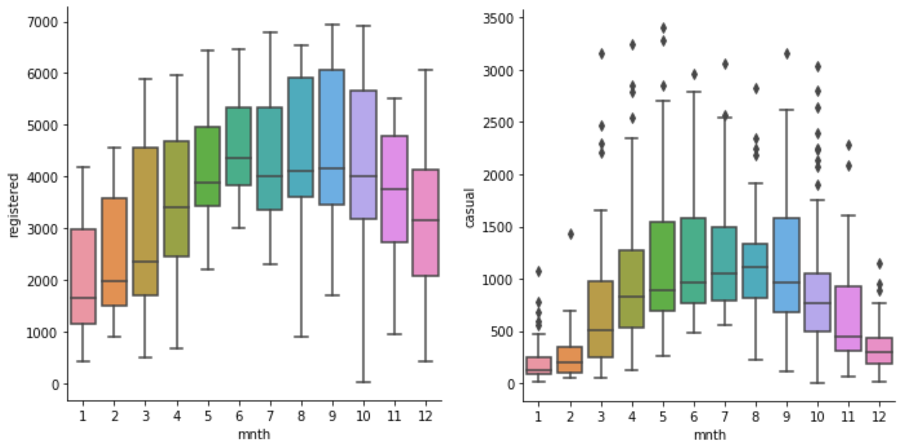
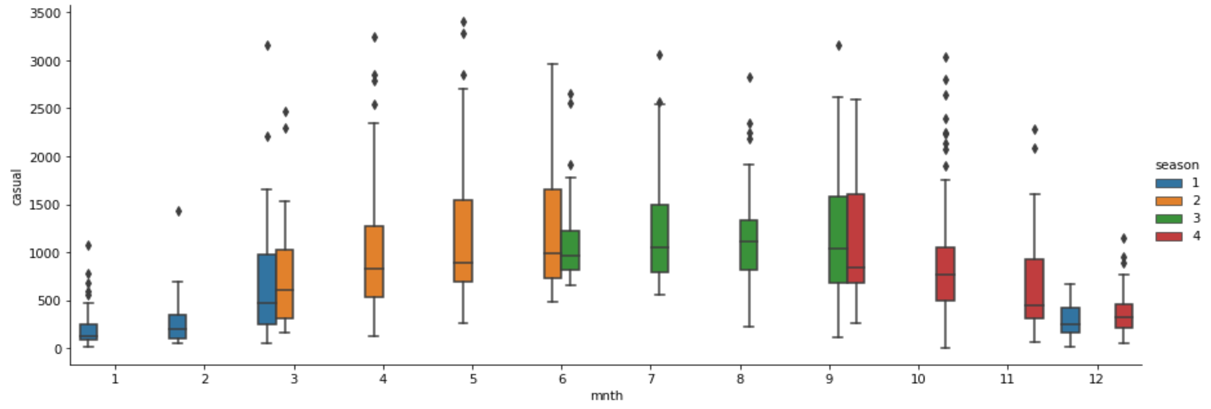


Fig 2.11 Boxplot of casual and registered users against month

May to September sees the maximum number of both casual and registered users while January, February and December see the minimum rentals. This could be down to the fact that the months in more demand correspond to the seasons summer and fall.
This can be verified by passing argument “hue=season” in the seaborn’s plotting function.



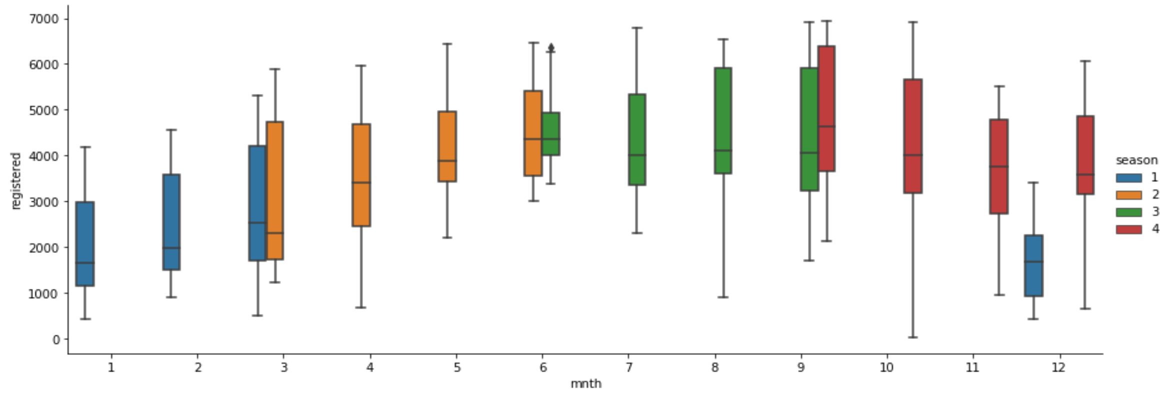


Fig 2.12 Monthly rentals with respect to season

Our hypothesis turned out to be true and it can be concluded that monthly demand is governed by the season and month may not be a very useful predictor for our model.

- Weekday

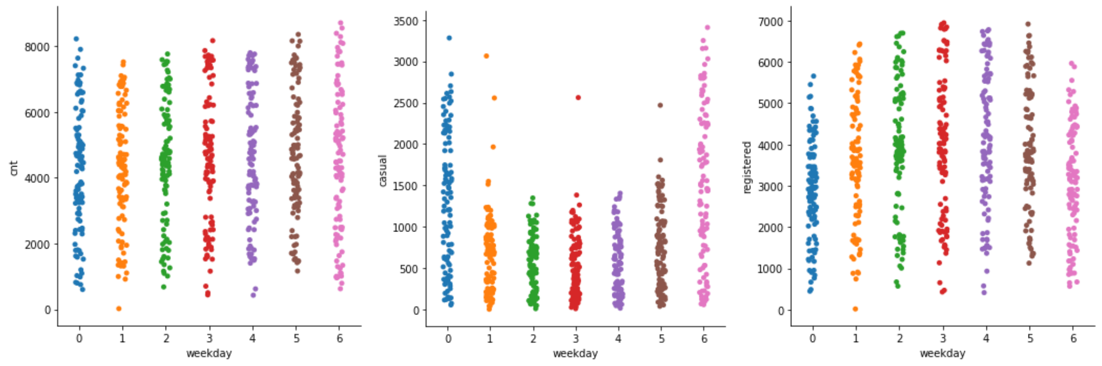


Fig 2.13 Cat plot of rentals with day of the week

Although casual users tend to be low on weekdays and high on weekends the overall count of rentals is not too much affected by this.

2.2.3 Dimension Reduction

There are a few statistical tests for feature selection- like ANOVA for categorical variables and the correlation test for numerical variables.

There are a few advance methods like Principal Component Analysis (PCA),Linear Discriminant Analysis (LDA) and Generalized Discriminant Analysis (GDA) for performing feature selection but since our dataset is of only 16 features we don't need to get into complex algorithms to reduce dimensions. Our task can be completed by the various intutions and hypothesis that we drew from the visualisations created by us.

Features to drop for training the data:-

1. instant – since it is only an index.
2. dteday – Dropping dteday because it is not a predictor.
3. mnth- Because we saw that season is the real predictor unlike month.
4. temp – temp and atemp have very similar values and I am choosing a atemp as my desired temperature learner for training.
5. casual - It is one of the target features.
6. registered – It is also a target feature.
7. cnt- Another target feature.

2.3 Feature Scaling

We don't need to provide any feature scaling technique on our data because all the continuous features in our data were already normalized when we loaded the dataset.

2.4 Feature Engineering

Feature Engineering is defined as the process of using domain knowledge of the data to create features that make machine learning algorithms work better.

While there is not much scope in our project to perform feature engineering but we will use 'One-Hot Encoding' on our predictor categorical variables.

A **one hot encoding** is a representation of categorical variables as binary vectors. This first requires that the categorical values be mapped to integer values. Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1.

We take this approach rather than integer encoding because:-

1. One hot encoding removes the chances of unnecessary weighing observations of a feature even when the categories of a feature do not have linear weighted relationship.
2. Our model has less features and it is not a computational expense to improve our model by increasing a few features.

Pandas 'get_dummies' function does our required task easily.

Features selected for one hot encoding are the ones having more than 2 levels.

Year, Season, weekday and weather are chosen for this task. We have chosen year despite it having only 2 levels is because we saw a huge yearly rise in the count of rentals and we want our model to capture this trend better.

After performing the above mentioned data cleaning processes we will first divide our data into two objects- Predictors set and Target set.

Our predictor set is represented by x and target set y.

- X = data.drop(['dteday','cnt','temp','registered','casual','mnth'],axis=1)
- Y= data[['cnt']]

2.5 Modelling

2.5.1 Model Selection

The first thing to consider when deciding the choice of the model is the type of target variable.

In our problem our target variable is of numeric type and it falls under regression type of problem.

There are a lot of algorithms to perform regression but we will use a regularized linear regression and random forests regression to build our model and will use corresponding evaluation metrics to measure the performance of our model.

Before building a model we will split our Predictor(X) set and target(Y) set into training and test data for validation of our model. We will train our model on the training data and predict for the test data and would evaluate the model accordingly.

The size of the training set chosen here is 75% of all the data, while we will test our model on the remaining 25%.

For building a model Python's scikit-learn (aka sklearn) library has been used, which provides numerous machine learning algorithms and evaluation metrics as well.

Splitting the data using train_test_split in sklearn:

- x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.25, random_state=124)

Now we have 2 pairs of object for building our model, each containing predictors and target features.

2.5.2 Linear Regression

We begin our modelling with the simplest of models, i.e. multiple linear regression.

- lm = LinearRegression().fit(x_train,y_train)
- p = lm.predict(x_test)
- r2_score(y_test,p)

2.5.3 Ridge Regression

Ridge Regression is a remedial measure taken to alleviate multicollinearity amongst regression predictor variables in a model. Often predictor variables used in regression are highly correlated. When they are, the regression coefficient of any one variable depend on which other predictor variables are included in the model, and which ones are left out. Ridge regression adds a small bias factor to the variables in order to alleviate this problem.

Sklearn has a built-in module RidgeCV ,which can be used to perform ridge regression and does automatic cross-validation as well.

The RidgeCV function demands that a parameter 'alpha' be passed to it having a array of positive float value. This alpha determines the regularization strength. Larger values of alpha specify stronger regularization. For the purpose of selecting the best alpha value we will pass an array of alpha values and call another function 'alpha_' with our model object which will tell us the best value of alpha.

- ridge = RidgeCV(alphas=(0.01,0.2,0.5,0.8),cv=5).fit(x_train,y_train)
- ridge.alpha_
- pred = ridge.predict(x_test)
- r2_score(y_test,pred)

2.5.4 Random Forest Regressor

It is an ensemble technique that builds multiple decision trees and merges them together to get a more accurate and stable prediction. Random forests are a form of bagging, and the averaging over trees can substantially reduce instability that might otherwise result.

Applying Random Forest:

- rf=RandomForestRegressor(n_estimators=200,bootstrap=True,max_depth=35,oob_score=True, random_state = 42)
- rf.fit(x_train, y_train)
- pre = rf.predict(x_test)
- rf.score(x_train,y_train)

Parameters used:
1. n_estimators = we choose 200 trees in our model after experimenting with a few values
2. bootstrap = enabling bootstrapping
3. max_depth = the maximum depth of the tree was chosen to be 35.

Chapter 3

Conclusion

3.1 Model Evaluation

There are multiple performance evaluation metrics present, but for our case we have used the following

1. R-squared – It tells us how much variability of the target feature is explained by our predictors.
2. RMSE - Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.
3. Feature Importance – The random forest regressor has a special function called feature importance which tells us the importance of predictor variables for the outcome.

3.2 Model Performance

1. Linear Regression

R-squared score - **0.83** which tells us 83% variability of our target feature is explained by our predictors.

RMSE – 801.37

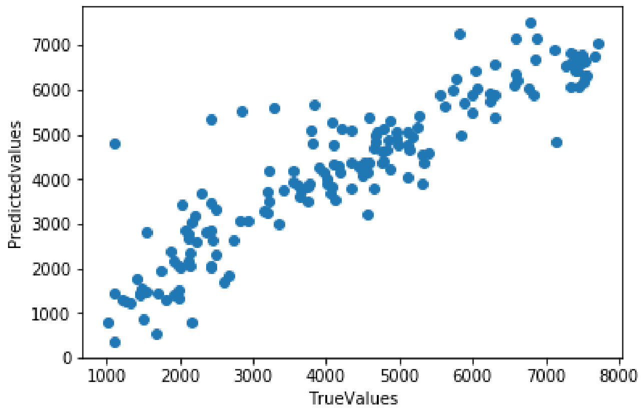


Fig 3.1 Scatterplot of predictions and actual values

As we can see from the scatterplot and our RMSE that our linear regression model is performing fairly well on our data. It is predicting correctly or close for most of the observations.

2. Ridge Regression

R-sqaured – 0.83 , which is same as that of the linear model.

RMSE – 801.6, again similar to linear model.

Our Ridge model even after cross-validation is showing similar results to that of our linear regression, so we can discard this model in favour of the linear model as it is easier to interpret

.

3. Random Forest

R-squared – 0.98, a very high value.

RMSE – 700

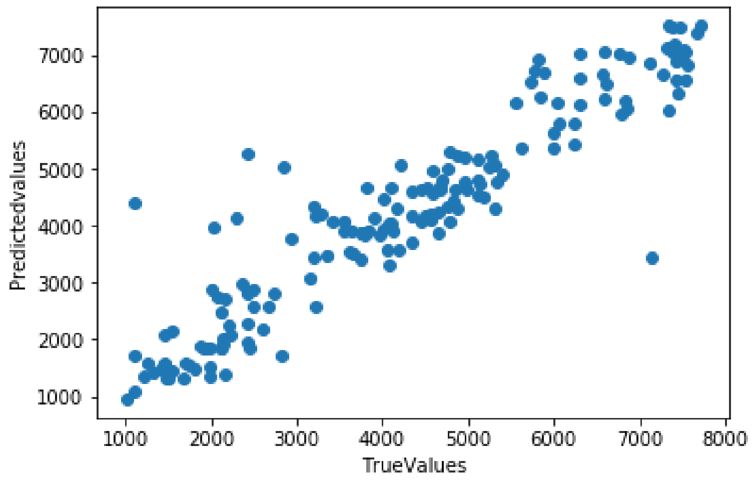


Fig 3.2 Predicted and True Values

As is evident from the plot the random forest model is showing excellent prediction results.

Feature importance:

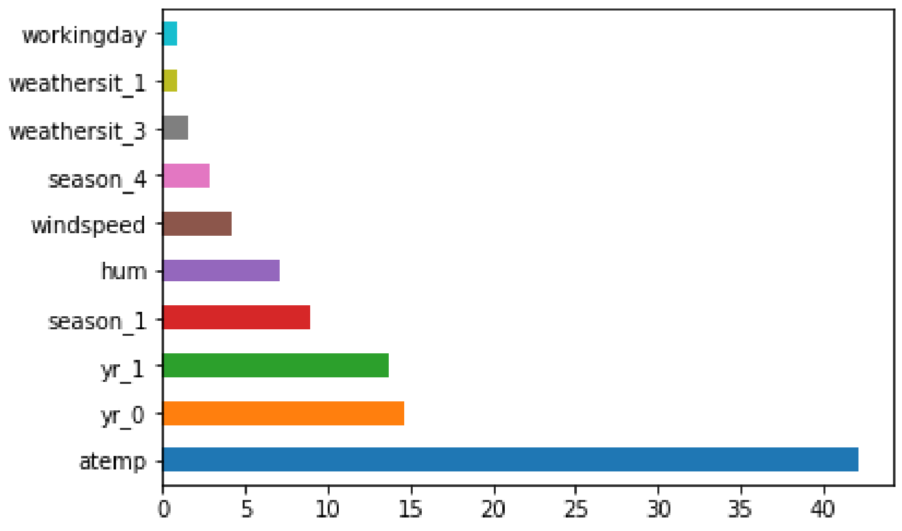


Fig 3.2 Feature importance by percentage in our model

During our exploratory data analysis we found out that temperature, year and season were the biggest factors governing the count of rentals. This hypothesis is proved true by the feature importance graph.

Weekday was considered as a weak predictor by us and it does not feature in the top 10 predictors further verifying our assumptions.

3.3 Model Selection

Our Random forest model is having near perfect R-squared value and this can mean that it is overfitting the data. While it may not be true as well , but the one downside of a tree based model is that it cannot predict values greater than what it was trained for. And as we saw that the count of rentals had been constantly increasing on a yearly basis our forest might not give true predictions on such new data.

Thus, Linear Regression model is the one finally chosen for this case, as it can predict close values on future unseen data as well and is performing fairly well as well.