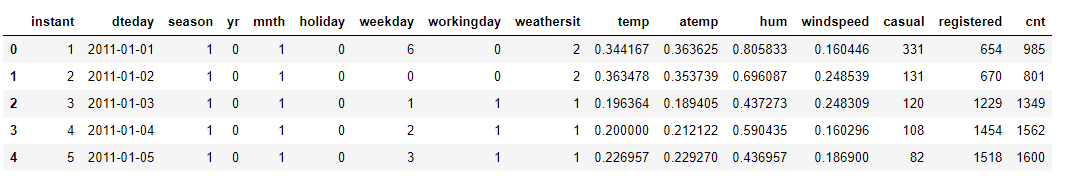
**Scope Of Project:**

In this project, we need to predict the count of bikes rented depending of some features like :

season,yr,mnth,hr,holiday ,weekday ,etc..

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. import seaborn as sns
5. from sklearn.preprocessing import MinMaxScaler
6. from sklearn.model\_selection import train\_test\_split
7. from sklearn.linear\_model import LinearRegression
8. from sklearn.metrics import mean\_absolute\_error

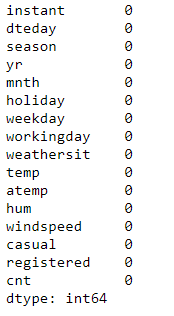
1. data1=pd.read\_csv('day.csv')
2. data1.head()



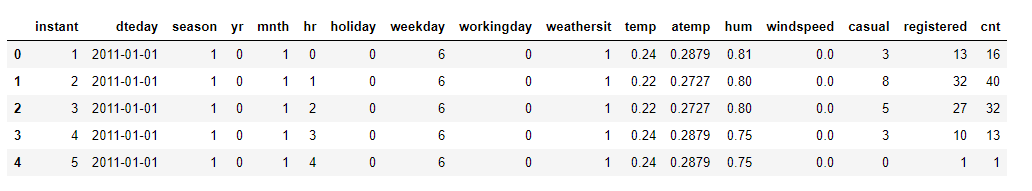
1. data1.shape



1. data1.isnull().sum()



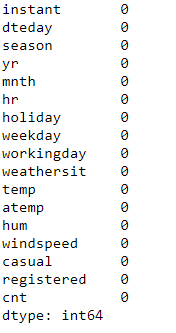
1. data2=pd.read\_csv('hour.csv')
2. data2.head()



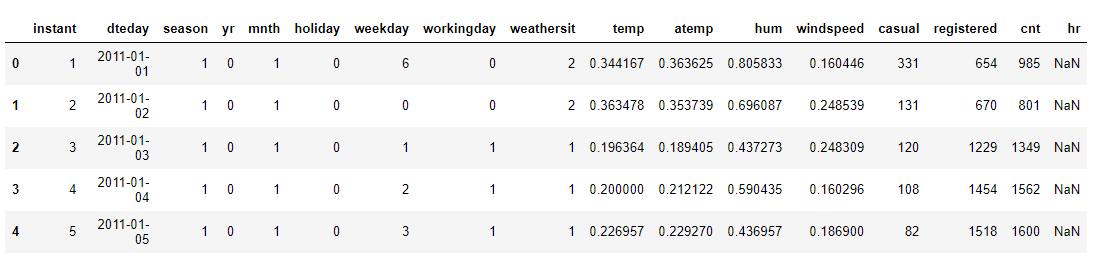
1. data2.shape



1. data2.isnull().sum()



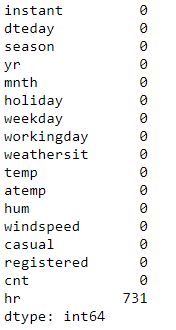
1. dataset=pd.concat([data1,data2])
2. dataset.head()



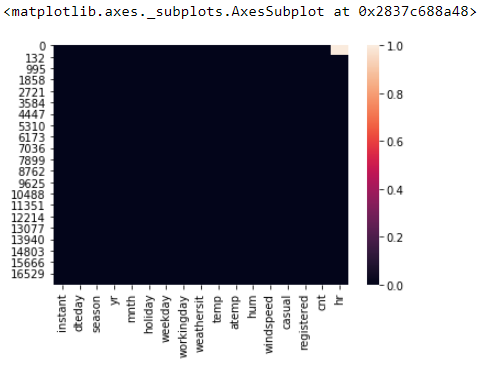
1. dataset.shape



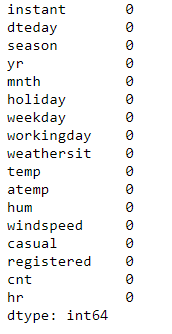
1. dataset.isnull().sum()



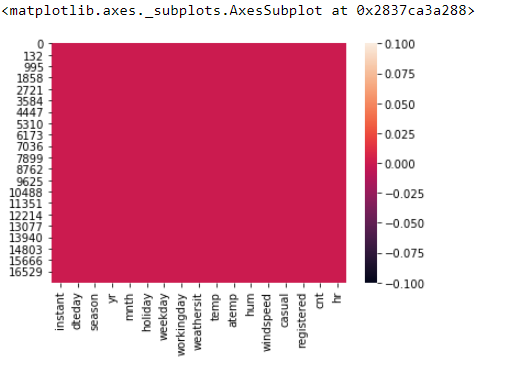
1. sns.heatmap(dataset.isnull())



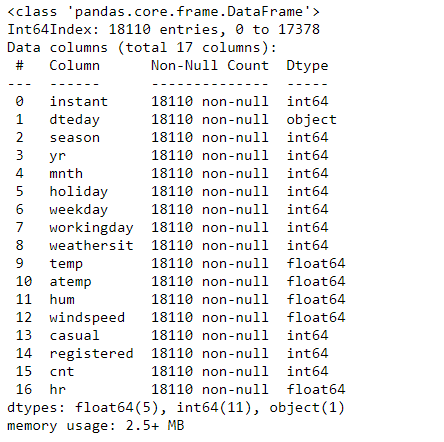
1. dataset['hr']=dataset['hr'].fillna(dataset['hr'].value\_counts().index[0])
2. dataset.isnull().sum()



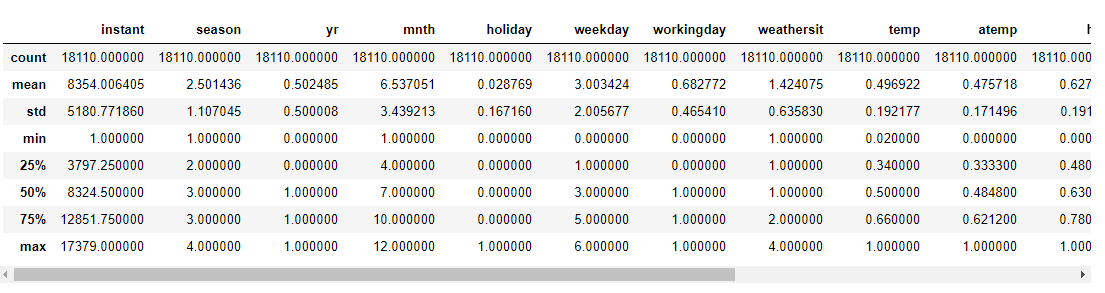
1. sns.heatmap(dataset.isnull())



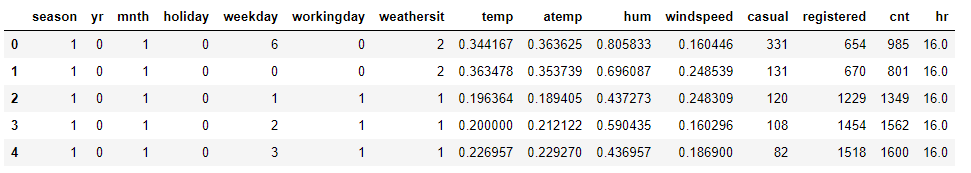
1. dataset.info()



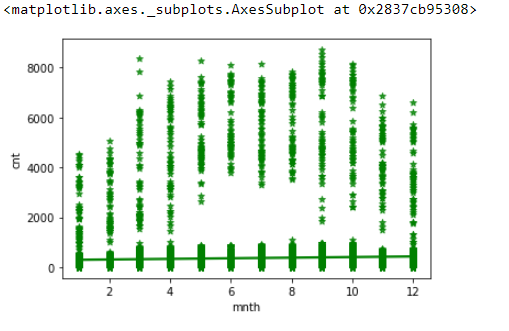
1. dataset.describe()



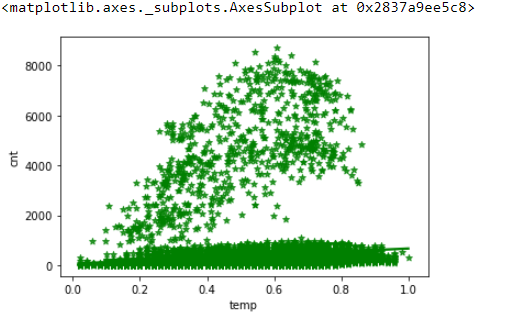
1. dataset=dataset.drop(['instant','dteday'],axis=1)
2. dataset.head()



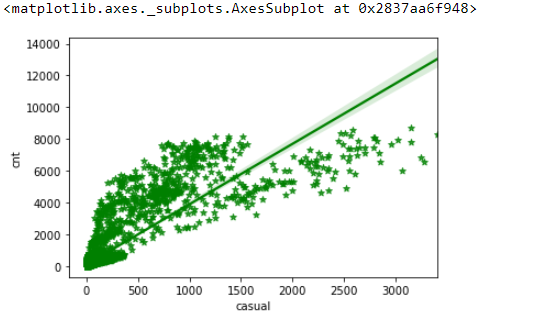
1. sns.regplot(x='mnth',y='cnt',data=dataset,color='green',marker='\*')



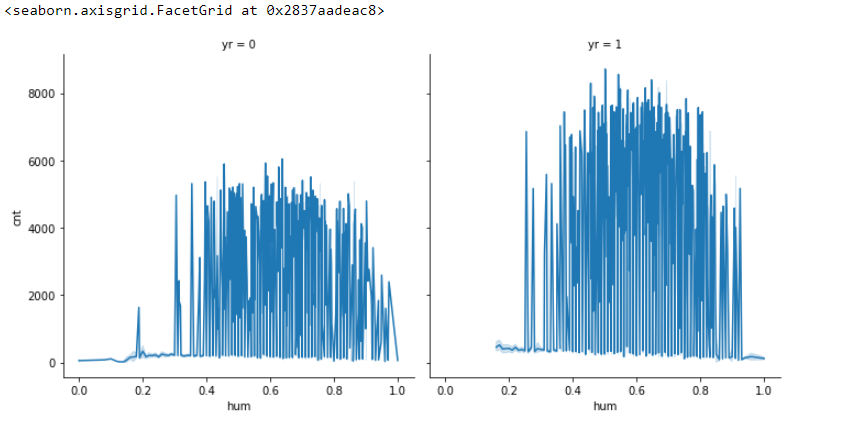
1. sns.regplot(x='temp',y='cnt',data=dataset,color='green',marker='\*')



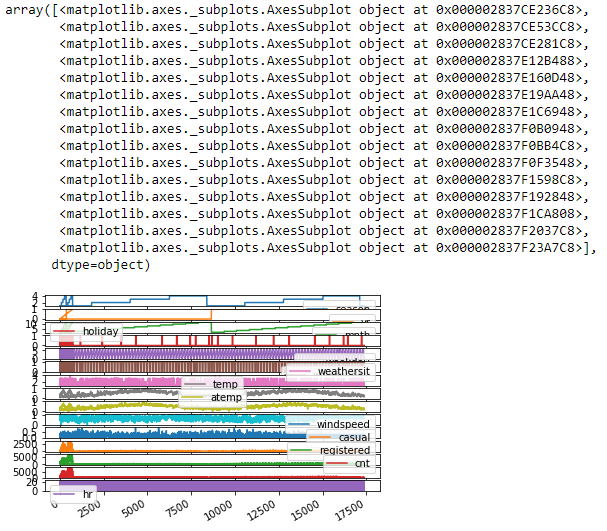
1. sns.regplot(x='casual',y='cnt',data=dataset,color='green',marker='\*')



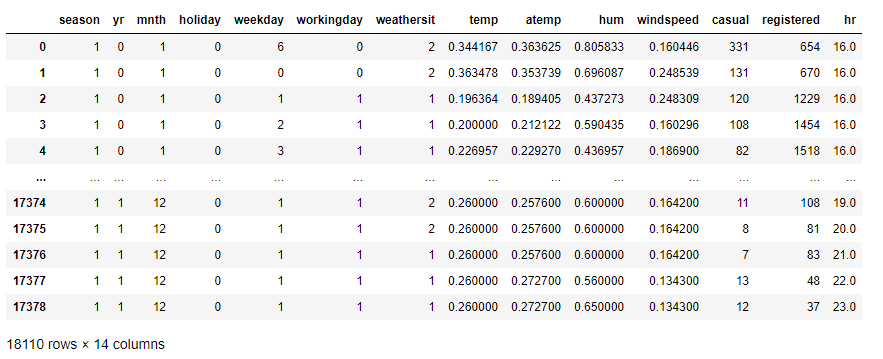
1. sns.relplot(x='hum',y='cnt',col='yr',data=dataset,kind='line')



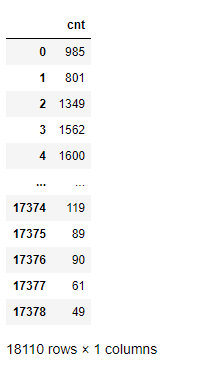
1. dataset.plot(subplots=True)



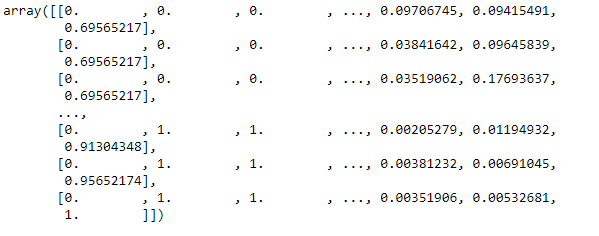
1. x=dataset[['season','yr','mnth','holiday','weekday','workingday','weathersit','temp','atemp','hum','windspeed','casual','registered','hr']]
2. x



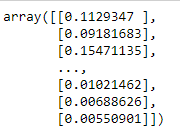
1. y=dataset[['cnt']]
2. y



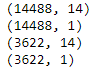
1. x=np.array(x)
2. y=np.array(y)
3. sc1=MinMaxScaler(feature\_range=(0,1))
4. x=sc1.fit\_transform(x)
5. x



1. y=sc1.fit\_transform(y)
2. y



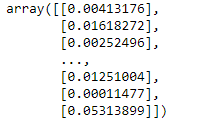
1. x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)
2. print(x\_train.shape)
3. print(y\_train.shape)
4. print(x\_test.shape)
5. print(y\_test.shape)



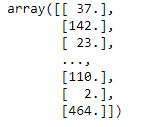
1. model=LinearRegression()
2. model.fit(x\_train,y\_train)



1. predictions=model.predict(x\_test)
2. predictions



1. org\_pred=sc1.inverse\_transform(predictions)
2. org\_pred



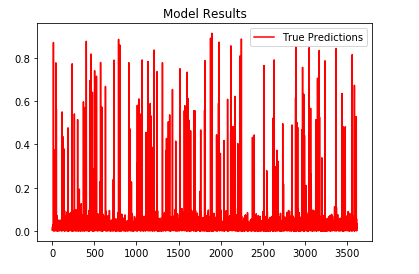
1. error=mean\_absolute\_error(y\_test,predictions)
2. error



1. score=model.score(x\_test,y\_test)
2. print(score)



1. plt.plot(y\_test,label='True Predictions',color='r')
2. #plt.plot(predictions,label='Predicted',color='b')
3. plt.title('Model Results')
4. plt.legend()
5. plt.show()



1. plt.plot(y\_test,label='True Predictions',color='r')
2. plt.plot(predictions,label='Predicted',color='b')
3. plt.title('Model Results')
4. plt.legend()
5. plt.show()

