```python
# Importing Data Analysis modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# For changing settings for more interaction with the shell
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = 'all'

# For suppressing warnings
import warnings
warnings.filterwarnings('ignore')

# For splitting data into train, test, to encode categorical
variables, to scale the features
from sklearn.preprocessing import MinMaxScaler, OneHotEncoder
from sklearn.model_selection import train_test_split

# For implementing linear regression
import statsmodels.api as sm

# For metric to evaluate the models
from sklearn.metrics import r2_score
from statsmodels.stats.outliers_influence import
variance_inflation_factor

# setting option for pandas to display all columns
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

# Reading the dataframe
df = pd.read_csv(r"C:\\Users\\Vivek\\Downloads\\day.csv")

df.head()
```

```
   instant      dteday  season  yr  mnth  holiday  weekday  workingday  \
0        1  01-01-2018       1   0     1        0        1           1

1        2  02-01-2018       1   0     1        0        2           1

2        3  03-01-2018       1   0     1        0        3           1

3        4  04-01-2018       1   0     1        0        4           1

4        5  05-01-2018       1   0     1        0        5           1


   weathersit       temp     atemp       hum  windspeed  casual
registered  \
```

```
0           2  14.110847  18.18125  80.5833  10.749882     331
654
1           2  14.902598  17.68695  69.6087  16.652113     131
670
2           1   8.050924   9.47025  43.7273  16.636703     120
1229
3           1   8.200000  10.60610  59.0435  10.739832     108
1454
4           1   9.305237  11.46350  43.6957  12.522300      82
1518

    cnt
0   985
1   801
2  1349
3  1562
4  1600
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 730 entries, 0 to 729
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   instant     730 non-null    int64
 1   dteday      730 non-null    object
 2   season      730 non-null    int64
 3   yr          730 non-null    int64
 4   mnth        730 non-null    int64
 5   holiday     730 non-null    int64
 6   weekday     730 non-null    int64
 7   workingday  730 non-null    int64
 8   weathersit  730 non-null    int64
 9   temp        730 non-null    float64
 10  atemp       730 non-null    float64
 11  hum         730 non-null    float64
 12  windspeed   730 non-null    float64
 13  casual      730 non-null    int64
 14  registered  730 non-null    int64
 15  cnt         730 non-null    int64
dtypes: float64(4), int64(11), object(1)
memory usage: 91.4+ KB
```

```
df.head()
```

```
   instant      dteday  season  yr  mnth  holiday  weekday  workingday
\
0        1  01-01-2018       1   0     1        0        1           1
```

```
1         2  02-01-2018       1   0     1        0        2           1

2         3  03-01-2018       1   0     1        0        3           1

3         4  04-01-2018       1   0     1        0        4           1

4         5  05-01-2018       1   0     1        0        5           1


   weathersit       temp     atemp      hum  windspeed  casual  
registered  \
0           2  14.110847  18.18125  80.5833  10.749882     331  
654   
1           2  14.902598  17.68695  69.6087  16.652113     131  
670   
2           1   8.050924   9.47025  43.7273  16.636703     120  
1229   
3           1   8.200000  10.60610  59.0435  10.739832     108  
1454   
4           1   9.305237  11.46350  43.6957  12.522300      82  
1518   

    cnt  
0   985  
1   801  
2  1349  
3  1562  
4  1600  

# Keeping a copy for safesake
df1 = df.copy()
```

Feature Engineering

```
# Dropping the unnecessary features
df1.drop(['instant', 'dteday','casual', 'registered'], axis=1, 
inplace=True)

# creating categorical and continuous variable list for later
categorical_vars = ['season', 'weekday', 'holiday', 'workingday', 
'weathersit', 'yr', 'mnth']
continuous_vars  = ['temp', 'atemp', 'hum', 'windspeed']

# Changing the type of the categorical variables to category
df1[categorical_vars] = df1[categorical_vars].astype('category')
```

Data Visualisation

```
# Creating a dataframe copy for visualisation
df2 = df1.copy()
```

```python
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 730 entries, 0 to 729
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype   
---  ------      --------------  -----   
 0   season      730 non-null    category
 1   yr          730 non-null    category
 2   mnth        730 non-null    category
 3   holiday     730 non-null    category
 4   weekday     730 non-null    category
 5   workingday  730 non-null    category
 6   weathersit  730 non-null    category
 7   temp        730 non-null    float64 
 8   atemp       730 non-null    float64 
 9   hum         730 non-null    float64 
 10  windspeed   730 non-null    float64 
 11  cnt         730 non-null    int64   
dtypes: category(7), float64(4), int64(1)
memory usage: 35.1 KB
```


```python
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 730 entries, 0 to 729
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype   
---  ------      --------------  -----   
 0   season      730 non-null    category
 1   yr          730 non-null    category
 2   mnth        730 non-null    category
 3   holiday     730 non-null    category
 4   weekday     730 non-null    category
 5   workingday  730 non-null    category
 6   weathersit  730 non-null    category
 7   temp        730 non-null    float64 
 8   atemp       730 non-null    float64 
 9   hum         730 non-null    float64 
 10  windspeed   730 non-null    float64 
 11  cnt         730 non-null    int64   
dtypes: category(7), float64(4), int64(1)
memory usage: 35.1 KB
```


Variables with positive trend with target:

atemp - with neither too high or too low variance. It might not be the best predictor, but it does have good linearity temp - same as temp Variables with negative trend with target:

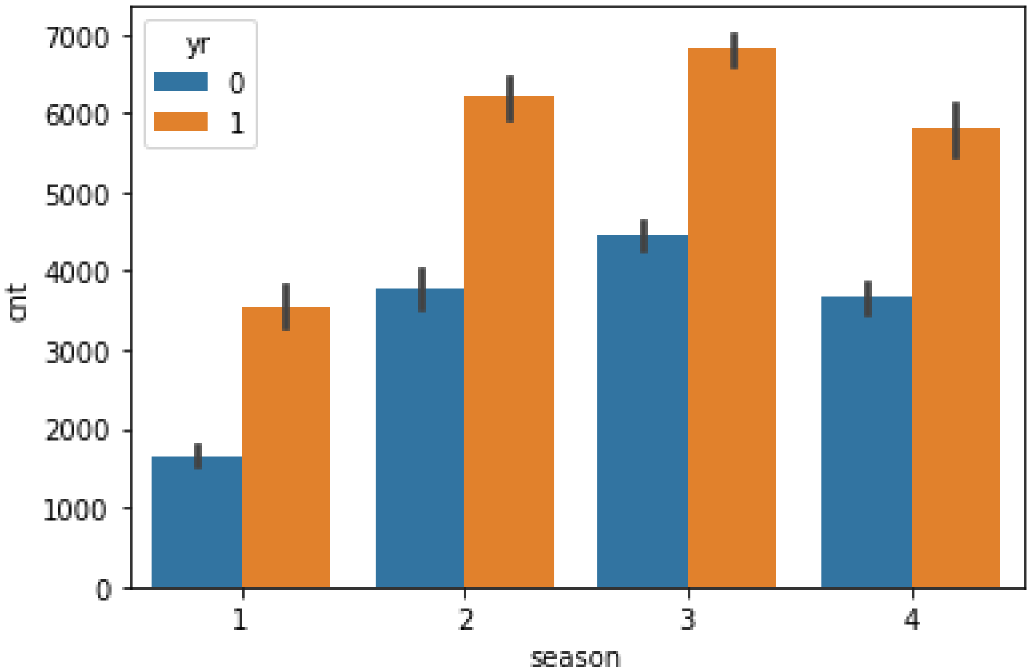
windspeed - high variance hum - high variance

```
categorical_vars, continuous_vars
```

```
(['season', 'weekday', 'holiday', 'workingday', 'weathersit', 'yr',
'mnth'],
 ['temp', 'atemp', 'hum', 'windspeed'])
```

```
sns.barplot(df2['season'], df2['cnt'], df2['yr'])
```

```
<AxesSubplot:xlabel='season', ylabel='cnt'>
```



Fall season has the highest bookings and Spring has the lowest

There are more bookings in the year 2019 than 2018

```
categorical_vars
```

```
['season', 'weekday', 'holiday', 'workingday', 'weathersit', 'yr',
'mnth']
```

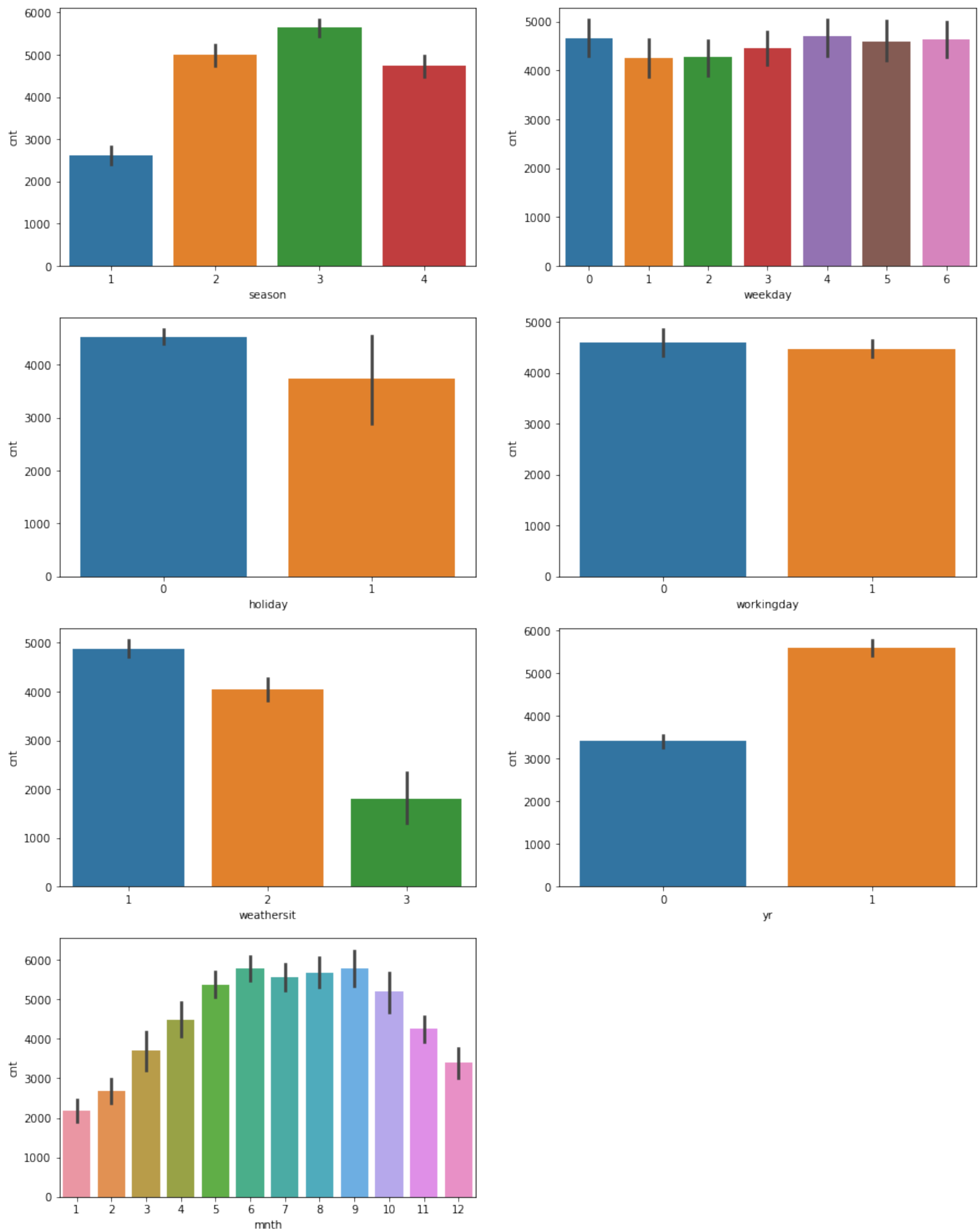
```
plt.figure(figsize=(15, 20))
plt.subplot(4,2,1)
sns.barplot(x = 'season', y = 'cnt', data = df2)
plt.subplot(4,2,2)
sns.barplot(x = 'weekday', y = 'cnt', data = df2)
plt.subplot(4,2,3)
sns.barplot(x = 'holiday', y = 'cnt', data = df2)
plt.subplot(4,2,4)
sns.barplot(x = 'workingday', y = 'cnt', data = df2)
plt.subplot(4,2,5)
```

```
sns.barplot(x = 'weathersit', y = 'cnt', data = df2)
plt.subplot(4,2,6)
sns.barplot(x = 'yr', y = 'cnt', data = df2)
plt.subplot(4,2,7)
sns.barplot(x = 'mnth', y = 'cnt', data = df2)
plt.savefig('../Images/cat_plot.png')
plt.show()
```

```
<Figure size 1080x1440 with 0 Axes>

<AxesSubplot:>

<AxesSubplot:xlabel='season', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='weekday', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='holiday', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='workingday', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='weathersit', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='yr', ylabel='cnt'>

<AxesSubplot:>

<AxesSubplot:xlabel='mnth', ylabel='cnt'>

----------------------------------------------------------------------
-----
FileNotFoundError                         Traceback (most recent call
last)
Input In [17], in <cell line: 16>()
     14 plt.subplot(4,2,7)
     15 sns.barplot(x = 'mnth', y = 'cnt', data = df2)
---> 16 plt.savefig('../Images/cat_plot.png')
     17 plt.show()

File ~\anaconda3\lib\site-packages\matplotlib\pyplot.py:958, in
savefig(*args, **kwargs)
    955 @_copy_docstring_and_deprecators(Figure.savefig)
    956 def savefig(*args, **kwargs):
    957     fig = gcf()
```

```
--> 958     res = fig.savefig(*args, **kwargs)
    959     fig.canvas.draw_idle()   # need this if 'transparent=True'
to reset colors
    960     return res

File ~\anaconda3\lib\site-packages\matplotlib\figure.py:3019, in
Figure.savefig(self, fname, transparent, **kwargs)
   3015     for ax in self.axes:
   3016         stack.enter_context(
   3017             ax.patch._cm_set(facecolor='none',
edgecolor='none'))
-> 3019 self.canvas.print_figure(fname, **kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\backend_bases.py:2319,
in FigureCanvasBase.print_figure(self, filename, dpi, facecolor,
edgecolor, orientation, format, bbox_inches, pad_inches,
bbox_extra_artists, backend, **kwargs)
   2315 try:
   2316     # _get_renderer may change the figure dpi (as vector
formats
   2317     # force the figure dpi to 72), so we need to set it again
here.
   2318     with cbook._setattr_cm(self.figure, dpi=dpi):
-> 2319         result = print_method(
   2320             filename,
   2321             facecolor=facecolor,
   2322             edgecolor=edgecolor,
   2323             orientation=orientation,
   2324             bbox_inches_restore=_bbox_inches_restore,
   2325             **kwargs)
   2326 finally:
   2327     if bbox_inches and restore_bbox:

File ~\anaconda3\lib\site-packages\matplotlib\backend_bases.py:1648,
in _check_savefig_extra_args.<locals>.wrapper(*args, **kwargs)
   1640     _api.warn_deprecated(
   1641         '3.3', name=name, removal='3.6',
   1642         message='%(name)s() got unexpected keyword argument "'
   1643                 + arg + '" which is no longer supported as of
'
   1644                 '%(since)s and will become an error '
   1645                 '%(removal)s')
   1646     kwargs.pop(arg)
-> 1648 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\_api\deprecation.py:412,
in delete_parameter.<locals>.wrapper(*inner_args, **inner_kwargs)
    402     deprecation_addendum = (
    403         f"If any parameter follows {name!r}, they should be
passed as "
```

```
    404         f"keyword, not positionally.")
    405     warn_deprecated(
    406         since,
    407         name=repr(name),
   (...)
    410                  else deprecation_addendum,
    411         **kwargs)
--> 412 return func(*inner_args, **inner_kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\backends\
backend_agg.py:541, in FigureCanvasAgg.print_png(self,
filename_or_obj, metadata, pil_kwargs, *args)
    494 """
    495 Write the figure to a PNG file.
    496
   (...)
    538     *metadata*, including the default 'Software' key.
    539 """
    540 FigureCanvasAgg.draw(self)
--> 541 mpl.image.imsave(
    542     filename_or_obj, self.buffer_rgba(), format="png",
origin="upper",
    543     dpi=self.figure.dpi, metadata=metadata,
pil_kwargs=pil_kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\image.py:1675, in
imsave(fname, arr, vmin, vmax, cmap, format, origin, dpi, metadata,
pil_kwargs)
   1673 pil_kwargs.setdefault("format", format)
   1674 pil_kwargs.setdefault("dpi", (dpi, dpi))
-> 1675 image.save(fname, **pil_kwargs)

File ~\anaconda3\lib\site-packages\PIL\Image.py:2209, in
Image.save(self, fp, format, **params)
   2207         fp = builtins.open(filename, "r+b")
   2208     else:
-> 2209         fp = builtins.open(filename, "w+b")
   2211 try:
   2212     save_handler(self, fp, filename)

FileNotFoundError: [Errno 2] No such file or directory:
'../Images/cat_plot.png'
```



cnt
season
1
2
3
4
weekday
0
1
2
3
4
5
6
holiday
workingday
weathersit
yr
mnth
7
8
9
10
11
12
1000
2000
3000
4000
5000
6000

The no of bookings increase month by month but it becomes less

as derived before yr, 2019 has highest bookings when the weather was clear has the highest bookings when the weather was snowy has the lowest bookings During Fall season, bookings were high

```
plt.figure(figsize=(10, 10))
plt.subplot(2,2,1)
sns.boxplot(x = 'atemp', data = df2)
plt.subplot(2,2,2)
sns.boxplot(x = 'temp', data = df2)
plt.subplot(2,2,3)
sns.boxplot(x = 'windspeed', data = df2)
plt.subplot(2,2,4)
sns.boxplot(x = 'hum', data = df2)
plt.show()
```

Encoding the categorical variables

```
for i in categorical_vars:
    df1[i].value_counts()
```

```
3    188
2    184
1    180
4    178
Name: season, dtype: int64
```

```
1    105
2    105
0    104
3    104
4    104
5    104
6    104
Name: weekday, dtype: int64
```

```
0    709
1     21
Name: holiday, dtype: int64
```

```
1    504
0    226
Name: workingday, dtype: int64
```

```
1    463
2    246
3     21
Name: weathersit, dtype: int64
```

```
0    365
1    365
Name: yr, dtype: int64

1     62
3     62
5     62
7     62
8     62
10    62
12    62
4     60
6     60
9     60
11    60
2     56
Name: mnth, dtype: int64

# As we saw before during visualisation, Fall is important as it has 
highest amount of bookings,
# so to lose the danger of dropping Fall, will encode it seperately

# Encoding 3 variables, weekday, weathersit, mnth
cat_encoded_df = pd.get_dummies(df1[['weekday', 'weathersit', 
'mnth']], drop_first=True)

# Now, encoding season
season_encoded_df = pd.get_dummies(df1['season'])
season_encoded_df = season_encoded_df[[2,3,4]]
season_encoded_df.columns = ['Summer', 'Fall', 'Winter']

# Concatenating the both dataframes
cat_encoded_df = pd.concat([cat_encoded_df, season_encoded_df],axis=1)

# As we have encoded the variables, we are going to drop the 4 
variables
df1.drop(['weekday', 'weathersit', 'mnth', 'season'], axis=1, 
inplace=True)

# concatenating encoded dataframe with original dataframe
df1 = pd.concat([df1, cat_encoded_df], axis=1)

df1.head()

  yr holiday workingday       temp     atemp      hum  windspeed   cnt 
\
0  0       0          1  14.110847  18.18125  80.5833  10.749882   985

1  0       0          1  14.902598  17.68695  69.6087  16.652113   801

2  0       0          1   8.050924   9.47025  43.7273  16.636703  1349
```

```
3  0       0          1   8.200000  10.60610  59.0435  10.739832  1562

4  0       0          1   9.305237  11.46350  43.6957  12.522300  1600


   weekday_1  weekday_2  weekday_3  weekday_4  weekday_5  weekday_6  \
0          1          0          0          0          0          0
1          0          1          0          0          0          0
2          0          0          1          0          0          0
3          0          0          0          1          0          0
4          0          0          0          0          1          0

   weathersit_2  weathersit_3  mnth_2  mnth_3  mnth_4  mnth_5  mnth_6
mnth_7  \
0             1             0       0       0       0       0       0
0
1             1             0       0       0       0       0       0
0
2             0             0       0       0       0       0       0
0
3             0             0       0       0       0       0       0
0
4             0             0       0       0       0       0       0
0

   mnth_8  mnth_9  mnth_10  mnth_11  mnth_12  Summer  Fall  Winter
0       0       0        0        0        0       0     0       0
1       0       0        0        0        0       0     0       0
2       0       0        0        0        0       0     0       0
3       0       0        0        0        0       0     0       0
4       0       0        0        0        0       0     0       0
```

Feature Scaling

```
# Feature Scaling should be done after splitting data in order to
avoid data breach.
# So we are going to divide the dataframe into train and test data
df_train, df_test = train_test_split(df1, test_size=0.25, random_state
= 100)

# Scaling the data in training data
scaler = MinMaxScaler()
df_train[continuous_vars] =
scaler.fit_transform(df_train[continuous_vars])

df_train[continuous_vars].head()

         temp     atemp       hum  windspeed
311  0.435124  0.437339  0.743667   0.057471
```

```
523  0.735215  0.680985  0.482181   0.286093
381  0.391151  0.374375  0.737917   0.659615
413  0.358285  0.362754  0.550880   0.319514
253  0.740406  0.695906  0.735509   0.156398
```

Building the model

```python
# We are going to use RFE for feature elimination
from sklearn.feature_selection import RFE

from sklearn.linear_model import LinearRegression

lr = LinearRegression()

y_train = df_train.cnt
X_train = df_train.drop('cnt', axis=1)
lr.fit(X_train, y_train)
rfe = RFE(lr,n_features_to_select = 20)
rfe.fit(X_train, y_train)
```

```
LinearRegression()
```

```
RFE(estimator=LinearRegression(), n_features_to_select=20)
```

```python
# Finding the dataframe with variables, support, ranking in order to
eliminate the variables
pd.DataFrame(zip(X_train.columns, rfe.support_, rfe.ranking_), columns
= ['Columns', 'Support', 'Ranking'])[rfe.support_]
```

	Columns	Support	Ranking
0	yr	True	1
1	holiday	True	1
2	workingday	True	1
3	temp	True	1
4	atemp	True	1
5	hum	True	1
6	windspeed	True	1
13	weathersit_2	True	1
14	weathersit_3	True	1
15	mnth_2	True	1
16	mnth_3	True	1
17	mnth_4	True	1
18	mnth_5	True	1
19	mnth_6	True	1
21	mnth_8	True	1
22	mnth_9	True	1
23	mnth_10	True	1
26	Summer	True	1
27	Fall	True	1
28	Winter	True	1

```
columns = X_train.columns[rfe.support_]
```

With the columns we have extracted with the help of RFE we are going to build our first model

Linear Model 1

```
X_train_sm = X_train[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr1 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr1.summary())

                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
0.851
Model:                            OLS   Adj. R-squared:
0.845
Method:                 Least Squares   F-statistic:
150.3
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
1.44e-202
Time:                        23:35:51   Log-Likelihood:
-4400.3
No. Observations:                 547   AIC:
8843.
Df Residuals:                     526   BIC:
8933.
Df Model:                          20

Covariance Type:            nonrobust

======================================================================
==========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
----------
const       2016.7512    239.814      8.410      0.000    1545.640
2487.863
yr          2001.6321     67.161     29.803      0.000    1869.695
2133.569
holiday    -1023.6950    212.233     -4.823      0.000   -1440.623
-606.767
workingday  -179.6461     76.718     -2.342      0.020    -330.358
-28.934
temp        3621.8473   1169.541      3.097      0.002    1324.303
```

```
5919.392
atemp           412.2622   1181.741      0.349      0.727   -1909.250
2733.774
hum           -1382.4254    321.471     -4.300      0.000   -2013.950
-750.900
windspeed     -1538.8523    218.697     -7.036      0.000   -1968.480
-1109.225
weathersit_2  -495.0557     87.029     -5.688      0.000    -666.024
-324.088
weathersit_3 -2182.8260    226.460     -9.639      0.000   -2627.703
-1737.949
mnth_2          202.0170    153.326      1.318      0.188     -99.189
503.223
mnth_3          478.2138    148.869      3.212      0.001     185.762
770.666
mnth_4          374.2058    221.811      1.687      0.092     -61.539
809.950
mnth_5          548.0678    224.183      2.445      0.015     107.663
988.473
mnth_6          443.4181    195.091      2.273      0.023      60.166
826.671
mnth_8          533.3705    155.028      3.440      0.001     228.820
837.921
mnth_9         1096.6479    147.226      7.449      0.000     807.425
1385.871
mnth_10         440.2961    145.848      3.019      0.003     153.779
726.813
Summer          881.5690    185.185      4.760      0.000     517.776
1245.362
Fall            524.3565    200.569      2.614      0.009     130.343
918.370
Winter         1503.1464    131.680     11.415      0.000    1244.464
1761.829
======================================================================
========
Omnibus:                       86.057   Durbin-Watson:
2.015
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
215.081
Skew:                          -0.806   Prob(JB):
1.98e-47
Kurtosis:                       5.615   Cond. No.
91.4
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
4	temp	64.19
5	atemp	57.84
0	const	53.20
19	Fall	7.02
18	Summer	5.87
13	mnth_5	3.58
12	mnth_4	3.16
20	Winter	2.97
14	mnth_6	2.55
6	hum	1.99
15	mnth_8	1.88
11	mnth_3	1.79
8	weathersit_2	1.59
17	mnth_10	1.58
16	mnth_9	1.48
10	mnth_2	1.47
9	weathersit_3	1.27
7	windspeed	1.26
2	holiday	1.11
3	workingday	1.10
1	yr	1.04

Removing the variable atemp as it has 0.817 p value

Temp has the highest vif at this time but p value precendence comes first so removing atemp variable

```
columns = list(X_train_sm.columns)
```

Linear Model 2

```
columns.remove('atemp')

X_train_sm = X_train_sm[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr2 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr2.summary())
```

```
                            OLS Regression Results                            
======================================================================
========
Dep. Variable:                    cnt   R-squared:                       
0.851
Model:                            OLS   Adj. R-squared:                  
0.846
Method:                 Least Squares   F-statistic:                     
158.5
Date:                Thu, 27 Apr 2023   Prob (F-statistic):          
1.19e-203
Time:                        23:35:57   Log-Likelihood:                
-4400.4
No. Observations:                 547   AIC:                             
8841.
Df Residuals:                     527   BIC:                             
8927.
Df Model:                          19                                         

Covariance Type:            nonrobust                                         

======================================================================
==========
                   coef    std err          t      P>|t|      [0.025      
0.975]
----------------------------------------------------------------------
----------
const         2018.6400    239.553      8.427      0.000    1548.043    
2489.237
yr            2001.0506     67.085     29.829      0.000    1869.264    
2132.837
holiday      -1025.9502    211.957     -4.840      0.000   -1442.335    
-609.565
workingday    -178.0769     76.522     -2.327      0.020    -328.403     
-27.750
temp          4015.4572    307.676     13.051      0.000    3411.036    
4619.879
hum          -1375.7185    320.628     -4.291      0.000   -2005.585    
-745.852
windspeed    -1554.4480    213.901     -7.267      0.000   -1974.651   
-1134.245
weathersit_2  -496.0979     86.906     -5.708      0.000    -666.822    
-325.374
weathersit_3 -2189.1631    225.542     -9.706      0.000   -2632.235   
-1746.092
mnth_2         202.8172    153.181      1.324      0.186     -98.103     
503.737
mnth_3         479.4271    148.705      3.224      0.001     187.300     
771.554
```

```
mnth_4          378.3924    221.302      1.710      0.088     -56.349
813.134
mnth_5          546.9336    223.973      2.442      0.015     106.944
986.923
mnth_6          439.6665    194.632      2.259      0.024      57.317
822.016
mnth_8          527.0380    153.833      3.426      0.001     224.836
829.240
mnth_9         1095.9481    147.090      7.451      0.000     806.994
1384.902
mnth_10         440.0573    145.725      3.020      0.003     153.784
726.331
Summer          881.3128    185.029      4.763      0.000     517.828
1244.798
Fall            519.2056    199.858      2.598      0.010     126.590
911.821
Winter         1505.0287    131.459     11.449      0.000    1246.780
1763.277
======================================================================
========
Omnibus:                       85.343   Durbin-Watson:
2.014
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
212.849
Skew:                          -0.800   Prob(JB):
6.03e-47
Kurtosis:                       5.603   Cond. No.
22.5
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
0	const	53.17
18	Fall	6.98
17	Summer	5.87
4	temp	4.45
12	mnth_5	3.58

```
11         mnth_4   3.15
19         Winter   2.96
13         mnth_6   2.54
5             hum   1.99
14         mnth_8   1.85
10         mnth_3   1.79
7    weathersit_2   1.59
16        mnth_10   1.58
15         mnth_9   1.48
9          mnth_2   1.47
8    weathersit_3   1.26
6       windspeed   1.21
2         holiday   1.11
3      workingday   1.10
1              yr   1.04
```

Removing mnth_4 variable because of it's p value.

Linear Model 3

```
columns.remove('mnth_4')

X_train_sm = X_train_sm[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr2 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr2.summary())

                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
0.850
Model:                            OLS   Adj. R-squared:
0.845
Method:                 Least Squares   F-statistic:
166.5
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
3.89e-204
Time:                        23:36:02   Log-Likelihood:
-4401.9
No. Observations:                 547   AIC:
8842.
Df Residuals:                     528   BIC:
8924.
Df Model:                          18

Covariance Type:            nonrobust
```

```
==============================================================================
==========
                     coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
----------
const         2060.5336    238.731      8.631      0.000    1591.555
2529.512
yr            2001.8872     67.205     29.788      0.000    1869.865
2133.909
holiday      -1035.4979    212.269     -4.878      0.000   -1452.494
-618.502
workingday    -175.9176     76.651     -2.295      0.022    -326.497
-25.339
temp          4065.7528    306.824     13.251      0.000    3463.008
4668.498
hum          -1423.8815    319.970     -4.450      0.000   -2052.452
-795.311
windspeed    -1519.9063    213.332     -7.125      0.000   -1938.991
-1100.822
weathersit_2  -494.6797     87.060     -5.682      0.000    -665.706
-323.653
weathersit_3 -2183.0760    225.924     -9.663      0.000   -2626.897
-1739.255
mnth_2         158.6194    151.259      1.049      0.295    -138.524
455.763
mnth_3         357.9644    130.878      2.735      0.006     100.859
615.069
mnth_5         282.8435    162.494      1.741      0.082     -36.370
602.057
mnth_6         238.9904    155.547      1.536      0.125     -66.576
544.557
mnth_8         482.8431    151.922      3.178      0.002     184.396
781.290
mnth_9        1065.2217    146.253      7.283      0.000     777.912
1352.532
mnth_10        431.0288    145.895      2.954      0.003     144.424
717.634
Summer        1092.2257    138.164      7.905      0.000     820.806
1363.645
Fall           501.5745    199.955      2.508      0.012     108.770
894.379
Winter        1466.7146    129.771     11.302      0.000    1211.783
1721.646
==============================================================================
========
Omnibus:                       87.748   Durbin-Watson:
2.033
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
```

```
219.438
Skew:                          -0.821   Prob(JB):
2.24e-48
Kurtosis:                       5.633   Cond. No.
22.0
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
0	const	52.62
17	Fall	6.96
4	temp	4.41
16	Summer	3.26
18	Winter	2.88
5	hum	1.97
11	mnth_5	1.88
13	mnth_8	1.80
12	mnth_6	1.62
7	weathersit_2	1.59
15	mnth_10	1.57
14	mnth_9	1.46
9	mnth_2	1.43
10	mnth_3	1.38
8	weathersit_3	1.26
6	windspeed	1.20
2	holiday	1.11
3	workingday	1.10
1	yr	1.04

Removing the variable mnth_5 because of it's p value

Linear Model 4

```
columns.remove('mnth_5')

X_train_sm = X_train_sm[columns]
```

```
X_train_sm = sm.add_constant(X_train_sm)
lr3 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr3.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
0.849
Model:                            OLS   Adj. R-squared:
0.845
Method:                 Least Squares   F-statistic:
175.4
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
1.30e-204
Time:                        23:36:06   Log-Likelihood:
-4403.4
No. Observations:                 547   AIC:
8843.
Df Residuals:                     529   BIC:
8920.
Df Model:                          17

Covariance Type:            nonrobust

======================================================================
==========
                   coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
----------
const         2022.0727    238.162      8.490      0.000    1554.214
2489.931
yr            1992.1995     67.102     29.689      0.000    1860.379
2124.020
holiday      -1049.0533    212.533     -4.936      0.000   -1466.566
-631.541
workingday    -174.7403     76.795     -2.275      0.023    -325.601
-23.879
temp          4225.2586    293.381     14.402      0.000    3648.923
4801.594
hum          -1379.0014    319.541     -4.316      0.000   -2006.726
-751.277
windspeed    -1550.2442    213.027     -7.277      0.000   -1968.726
-1131.762
weathersit_2  -496.6277     87.220     -5.694      0.000    -667.967
-325.288
weathersit_3 -2197.9848    226.194     -9.717      0.000   -2642.334
-1753.635
```

```
mnth_2           141.1642     151.216        0.934        0.351     -155.892
438.221
mnth_3           300.4955     126.888        2.368        0.018       51.230
549.761
mnth_6           126.4722     141.750        0.892        0.373     -151.990
404.934
mnth_8           453.7776     151.291        2.999        0.003      156.572
750.983
mnth_9          1042.7947     145.964        7.144        0.000      756.054
1329.535
mnth_10          408.1495     145.580        2.804        0.005      122.164
694.135
Summer          1151.4965     134.159        8.583        0.000      887.946
1415.047
Fall             424.0259     195.302        2.171        0.030       40.364
807.688
Winter          1426.3182     127.924       11.150        0.000     1175.017
1677.619
======================================================================
========
Omnibus:                        84.961   Durbin-Watson:
2.036
Prob(Omnibus):                   0.000   Jarque-Bera (JB):
199.708
Skew:                           -0.817   Prob(JB):
4.31e-44
Kurtosis:                        5.468   Cond. No.
21.9
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] = 
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i 
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

```
        Features    VIF
0          const  52.17
16          Fall   6.62
4           temp   4.02
15        Summer   3.06
17        Winter   2.78
```

```
5              hum   1.96
12          mnth_8   1.78
7     weathersit_2   1.59
14         mnth_10   1.56
13          mnth_9   1.45
9           mnth_2   1.43
11          mnth_6   1.34
10          mnth_3   1.30
8     weathersit_3   1.26
6        windspeed   1.19
2          holiday   1.11
3       workingday   1.10
1               yr   1.04
```

Have to remove mnth_6 because of it's p value

Linear Model 5

```
columns.remove('mnth_6')

X_train_sm = X_train_sm[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr4 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr4.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
0.849
Model:                            OLS   Adj. R-squared:
0.845
Method:                 Least Squares   F-statistic:
186.4
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
1.39e-205
Time:                        23:36:11   Log-Likelihood:
-4403.8
No. Observations:                 547   AIC:
8842.
Df Residuals:                     530   BIC:
8915.
Df Model:                          16

Covariance Type:            nonrobust

======================================================================
==========
```

```
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
----------
const        2032.7748    237.814      8.548      0.000    1565.602
2499.948
yr           1990.1304     67.049     29.682      0.000    1858.415
2121.846
holiday     -1052.7901    212.451     -4.955      0.000   -1470.139
-635.441
workingday   -172.9511     76.754     -2.253      0.025    -323.731
-22.171
temp         4304.4088    279.594     15.395      0.000    3755.160
4853.657
hum         -1421.3129    315.941     -4.499      0.000   -2041.964
-800.662
windspeed   -1564.0351    212.424     -7.363      0.000   -1981.332
-1146.738
weathersit_2  -492.7044     87.092     -5.657      0.000    -663.792
-321.617
weathersit_3 -2189.5837    225.955     -9.690      0.000   -2633.461
-1745.707
mnth_2        135.7609    151.065      0.899      0.369    -160.999
432.521
mnth_3        285.3676    125.726      2.270      0.024      38.386
532.349
mnth_8        425.8285    147.984      2.878      0.004     135.120
716.537
mnth_9       1023.6001    144.342      7.091      0.000     740.047
1307.153
mnth_10       397.6856    145.079      2.741      0.006     112.686
682.685
Summer       1153.3839    134.117      8.600      0.000     889.918
1416.850
Fall          408.2197    194.459      2.099      0.036      26.214
790.225
Winter       1416.1490    127.391     11.117      0.000    1165.897
1666.402
======================================================================
========
Omnibus:                       85.052   Durbin-Watson:
2.038
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
198.567
Skew:                          -0.821   Prob(JB):
7.62e-44
Kurtosis:                       5.453   Cond. No.
21.5
======================================================================
```

```
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
0	const	52.03
15	Fall	6.56
4	temp	3.65
14	Summer	3.06
16	Winter	2.76
5	hum	1.91
11	mnth_8	1.70
7	weathersit_2	1.59
13	mnth_10	1.55
9	mnth_2	1.42
12	mnth_9	1.42
10	mnth_3	1.27
8	weathersit_3	1.25
6	windspeed	1.19
2	holiday	1.11
3	workingday	1.10
1	yr	1.03

Removing mnth_3 because of it's p value

Linear Model 6

```
columns.remove('mnth_3')

X_train_sm = X_train_sm[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr5 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr5.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
```

```
0.848
Model:                            OLS   Adj. R-squared:
0.843
Method:                 Least Squares   F-statistic:
197.0
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
1.25e-205
Time:                        23:36:15   Log-Likelihood:
-4406.5
No. Observations:                 547   AIC:
8845.
Df Residuals:                     531   BIC:
8914.
Df Model:                          15

Covariance Type:            nonrobust

======================================================================
==========
                   coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
----------
const         2141.4804    233.850      9.157      0.000    1682.096
2600.865
yr            1987.1194     67.298     29.527      0.000    1854.917
2119.322
holiday      -1092.7053    212.548     -5.141      0.000   -1510.243
-675.167
workingday    -176.2524     77.040     -2.288      0.023    -327.593
-24.912
temp          4341.9169    280.194     15.496      0.000    3791.492
4892.342
hum          -1464.3087    316.603     -4.625      0.000   -2086.257
-842.360
windspeed    -1537.5688    212.932     -7.221      0.000   -1955.861
-1119.277
weathersit_2  -490.4484     87.426     -5.610      0.000    -662.192
-318.705
weathersit_3 -2155.4878    226.335     -9.523      0.000   -2600.109
-1710.866
mnth_2          35.3122    145.001      0.244      0.808    -249.534
320.158
mnth_8         428.4897    148.557      2.884      0.004     136.658
720.322
mnth_9        1028.9912    144.886      7.102      0.000     744.372
1313.611
mnth_10        391.7635    145.621      2.690      0.007     105.699
677.828
```

```
Summer         1082.2256    130.910      8.267      0.000     825.061
1339.391
Fall            291.5674    188.277      1.549      0.122     -78.291
661.426
Winter         1316.9578    120.127     10.963      0.000    1080.975
1552.941
======================================================================
========
Omnibus:                       81.868   Durbin-Watson:
2.034
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
199.713
Skew:                          -0.777   Prob(JB):
4.29e-44
Kurtosis:                       5.519   Cond. No.
21.2
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
0	const	49.92
14	Fall	6.10
4	temp	3.64
13	Summer	2.89
15	Winter	2.44
5	hum	1.91
10	mnth_8	1.70
7	weathersit_2	1.59
12	mnth_10	1.55
11	mnth_9	1.42
9	mnth_2	1.30
8	weathersit_3	1.25
6	windspeed	1.18
2	holiday	1.10
3	workingday	1.10
1	yr	1.03

Removing Fall because of it's high VIF value

Linear Model 7

```
columns.remove('Fall')

X_train_sm = X_train_sm[columns]
X_train_sm = sm.add_constant(X_train_sm)
lr6 = sm.OLS(y_train, X_train_sm.astype(float)).fit()
print(lr6.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                    cnt   R-squared:
0.847
Model:                            OLS   Adj. R-squared:
0.843
Method:                 Least Squares   F-statistic:
210.3
Date:                Thu, 27 Apr 2023   Prob (F-statistic):
2.78e-206
Time:                        23:36:18   Log-Likelihood:
-4407.7
No. Observations:                 547   AIC:
8845.
Df Residuals:                     532   BIC:
8910.
Df Model:                          14

Covariance Type:            nonrobust

======================================================================
==========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
----------
const       2141.6003    234.157      9.146      0.000    1681.614
2601.586
yr          1979.0470     67.184     29.457      0.000    1847.069
2111.025
holiday    -1101.8404    212.745     -5.179      0.000   -1519.764
-683.917
workingday  -175.7625     77.140     -2.278      0.023    -327.300
-24.225
temp        4658.7811    191.675     24.306      0.000    4282.249
5035.313
hum        -1524.2448    314.641     -4.844      0.000   -2142.337
```

```
-906.153
windspeed    -1560.0776    212.714     -7.334      0.000   -1977.940
-1142.215
weathersit_2  -483.9904     87.441     -5.535      0.000    -655.763
-312.218
weathersit_3 -2119.1543    225.411     -9.401      0.000   -2561.959
-1676.349
mnth_2         -13.3413    141.742     -0.094      0.925    -291.785
265.102
mnth_8         512.9906    138.355      3.708      0.000     241.202
784.779
mnth_9        1102.6833    137.028      8.047      0.000     833.501
1371.866
mnth_10        361.8594    144.525      2.504      0.013      77.950
645.768
Summer         940.8796     93.967     10.013      0.000     756.287
1125.472
Winter        1225.8513    104.873     11.689      0.000    1019.836
1431.867
======================================================================
========
Omnibus:                       76.323   Durbin-Watson:
2.033
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
174.586
Skew:                          -0.749   Prob(JB):
1.23e-38
Kurtosis:                       5.327   Cond. No.
20.6
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
vif = pd.DataFrame()
vif['Features'] = X_train_sm.columns
vif['VIF'] =
[variance_inflation_factor(X_train_sm.astype(float).values, i) for i
in range(X_train_sm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = 'VIF', ascending=False)
vif
```

	Features	VIF
0	const	49.92
5	hum	1.88
14	Winter	1.85
4	temp	1.70

```
7   weathersit_2   1.58
12       mnth_10   1.52
13        Summer   1.49
10        mnth_8   1.47
11        mnth_9   1.26
9         mnth_2   1.24
8   weathersit_3   1.23
6      windspeed   1.18
2        holiday   1.10
3     workingday   1.10
1             yr   1.03
```

Now we move on to Predictions, So we are going to scale the dataframe with test data and start predicting with latest linear model

```
scaler = MinMaxScaler()
df_test[continuous_vars] =
scaler.fit_transform(df_test[continuous_vars])

df_test[continuous_vars].head()
```

	temp	atemp	hum	windspeed
184	0.837241	0.778767	0.534223	0.149393
535	0.911423	0.855132	0.470417	0.231142
299	0.496221	0.492359	0.777843	0.443398
221	0.890387	0.805661	0.236659	0.449707
152	0.821739	0.749249	0.070765	0.682387

```
y_test = df_test.pop('cnt')
X_test = df_test
X_test = sm.add_constant(X_test)
X_test = X_test[columns]
```

To evaluaate the model, we are going to check the r2 score and then we will plot a distribution plot for error terms

```
y_pred = lr6.predict(X_test)
r2_score(y_test, y_pred)
```

```
0.7848149370296662
```

```
n = 183
k = 12
R2 = r2_score(y_test, y_pred)
adj_r2 = 1 - ((1-R2)*(n-1)/(n-k-1))
adj_r2
```

```
0.7696254031729368
```

```
X_test.shape
```

```
(183, 15)
```

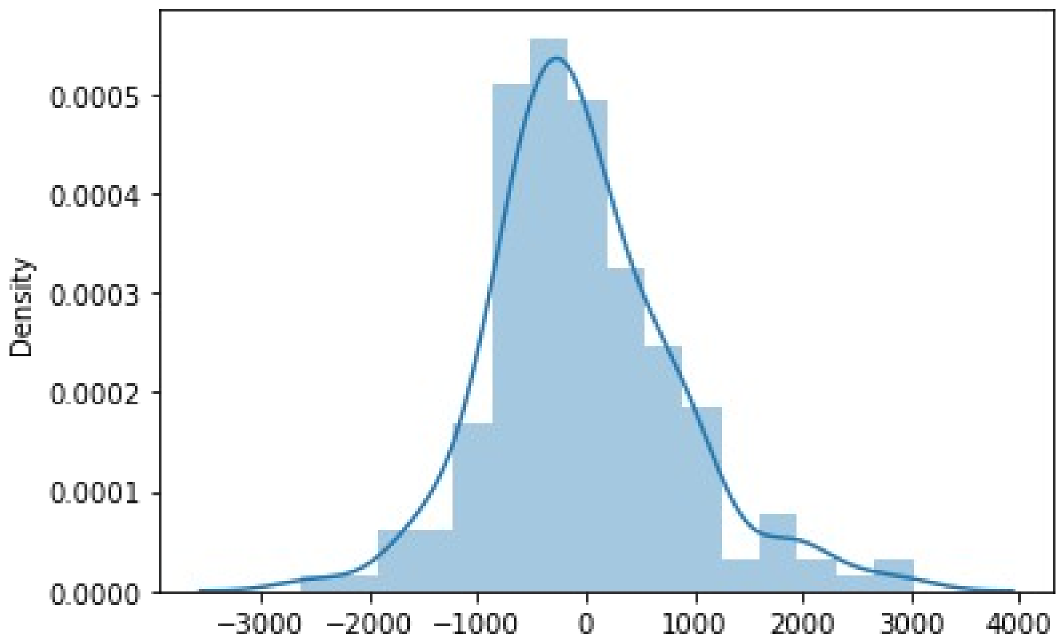
```
sns.distplot(y_pred - y_test)
plt.savefig('../Images/Residual.png')
```

```
<AxesSubplot:ylabel='Density'>
```

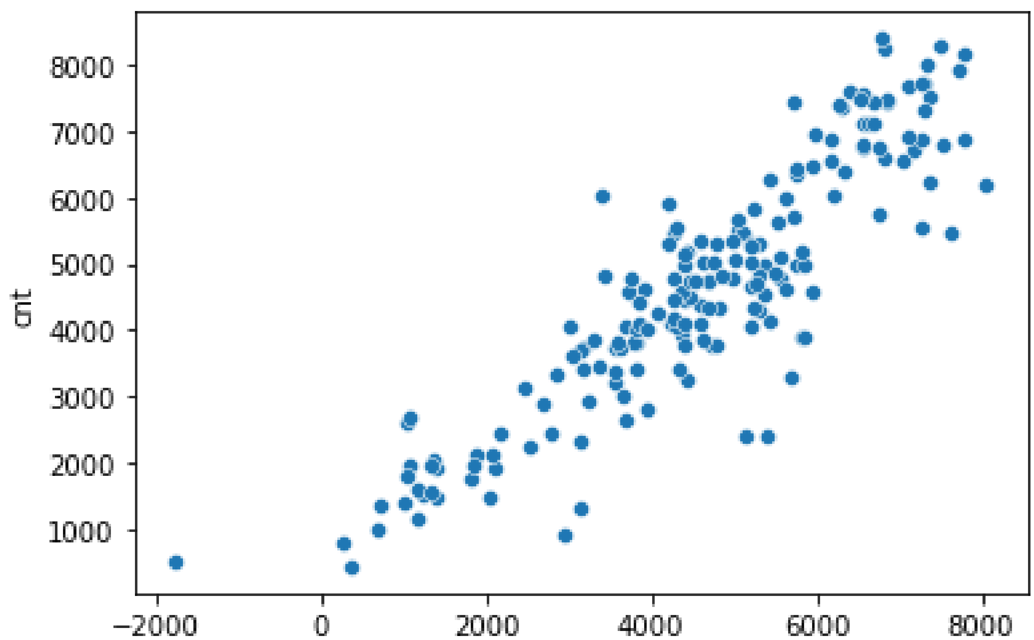
```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call
last)
Input In [51], in <cell line: 2>()
      1 sns.distplot(y_pred - y_test)
----> 2 plt.savefig('../Images/Residual.png')

File ~\anaconda3\lib\site-packages\matplotlib\pyplot.py:958, in
savefig(*args, **kwargs)
    955 @_copy_docstring_and_deprecators(Figure.savefig)
    956 def savefig(*args, **kwargs):
    957     fig = gcf()
--> 958     res = fig.savefig(*args, **kwargs)
    959     fig.canvas.draw_idle()   # need this if 'transparent=True'
to reset colors
    960     return res

File ~\anaconda3\lib\site-packages\matplotlib\figure.py:3019, in
Figure.savefig(self, fname, transparent, **kwargs)
   3015     for ax in self.axes:
   3016         stack.enter_context(
   3017             ax.patch._cm_set(facecolor='none',
edgecolor='none'))
-> 3019 self.canvas.print_figure(fname, **kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\backend_bases.py:2319,
in FigureCanvasBase.print_figure(self, filename, dpi, facecolor,
edgecolor, orientation, format, bbox_inches, pad_inches,
bbox_extra_artists, backend, **kwargs)
   2315 try:
   2316     # _get_renderer may change the figure dpi (as vector
formats
   2317     # force the figure dpi to 72), so we need to set it again
here.
   2318     with cbook._setattr_cm(self.figure, dpi=dpi):
-> 2319         result = print_method(
   2320             filename,
   2321             facecolor=facecolor,
   2322             edgecolor=edgecolor,
   2323             orientation=orientation,
   2324             bbox_inches_restore=_bbox_inches_restore,
   2325             **kwargs)
```

```
   2326 finally:
   2327     if bbox_inches and restore_bbox:

File ~\anaconda3\lib\site-packages\matplotlib\backend_bases.py:1648,
in _check_savefig_extra_args.<locals>.wrapper(*args, **kwargs)
   1640     _api.warn_deprecated(
   1641         '3.3', name=name, removal='3.6',
   1642         message='%(name)s() got unexpected keyword argument "'
   1643                 + arg + '" which is no longer supported as of
'
   1644                 '%(since)s and will become an error '
   1645                 '%(removal)s')
   1646     kwargs.pop(arg)
-> 1648 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\_api\deprecation.py:412,
in delete_parameter.<locals>.wrapper(*inner_args, **inner_kwargs)
    402     deprecation_addendum = (
    403         f"If any parameter follows {name!r}, they should be
passed as "
    404         f"keyword, not positionally.")
    405     warn_deprecated(
    406         since,
    407         name=repr(name),
   (...)
    410                  else deprecation_addendum,
    411         **kwargs)
--> 412 return func(*inner_args, **inner_kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\backends\
backend_agg.py:541, in FigureCanvasAgg.print_png(self,
filename_or_obj, metadata, pil_kwargs, *args)
    494 """
    495 Write the figure to a PNG file.
    496
   (...)
    538     *metadata*, including the default 'Software' key.
    539 """
    540 FigureCanvasAgg.draw(self)
--> 541 mpl.image.imsave(
    542     filename_or_obj, self.buffer_rgba(), format="png",
origin="upper",
    543     dpi=self.figure.dpi, metadata=metadata,
pil_kwargs=pil_kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\image.py:1675, in
imsave(fname, arr, vmin, vmax, cmap, format, origin, dpi, metadata,
pil_kwargs)
   1673 pil_kwargs.setdefault("format", format)
   1674 pil_kwargs.setdefault("dpi", (dpi, dpi))
```

```
-> 1675 image.save(fname, **pil_kwargs)

File ~\anaconda3\lib\site-packages\PIL\Image.py:2209, in
Image.save(self, fp, format, **params)
   2207         fp = builtins.open(filename, "r+b")
   2208     else:
-> 2209         fp = builtins.open(filename, "w+b")
   2211 try:
   2212     save_handler(self, fp, filename)

FileNotFoundError: [Errno 2] No such file or directory:
'../Images/Residual.png'
```




```
sns.scatterplot(y_pred, y_test)

<AxesSubplot:ylabel='cnt'>
```




```
lr6.params.shape
```

```
(15,)
```

```
plt.figure(figsize = (15, 8))
sns.barplot(lr6.params.values, lr6.params.values)
ticks = np.arange(0,15,1)
plt.xticks(ticks, labels = lr6.params.sort_values().index)
plt.show()
```

```
<Figure size 1080x576 with 0 Axes>
```

```
<AxesSubplot:>
```

```
([<matplotlib.axis.XTick at 0x1f45f73e1c0>,
  <matplotlib.axis.XTick at 0x1f45f73e190>,
  <matplotlib.axis.XTick at 0x1f45f1b8100>,
  <matplotlib.axis.XTick at 0x1f45f790610>,
  <matplotlib.axis.XTick at 0x1f45f790d60>,
  <matplotlib.axis.XTick at 0x1f45f79b4f0>,
  <matplotlib.axis.XTick at 0x1f45f79bc40>,
  <matplotlib.axis.XTick at 0x1f45f7a13d0>,
  <matplotlib.axis.XTick at 0x1f45f79b8b0>,
  <matplotlib.axis.XTick at 0x1f45f790880>,
  <matplotlib.axis.XTick at 0x1f45f7a1bb0>,
  <matplotlib.axis.XTick at 0x1f45f7a8340>,
  <matplotlib.axis.XTick at 0x1f45f7a8a90>,
  <matplotlib.axis.XTick at 0x1f45f7ae220>,
  <matplotlib.axis.XTick at 0x1f45f7ae970>],
 [Text(0, 0, 'weathersit_3'),
```

```
 Text(1, 0, 'windspeed'),
 Text(2, 0, 'hum'),
 Text(3, 0, 'holiday'),
 Text(4, 0, 'weathersit_2'),
 Text(5, 0, 'workingday'),
 Text(6, 0, 'mnth_2'),
 Text(7, 0, 'mnth_10'),
 Text(8, 0, 'mnth_8'),
 Text(9, 0, 'Summer'),
 Text(10, 0, 'mnth_9'),
 Text(11, 0, 'Winter'),
 Text(12, 0, 'yr'),
 Text(13, 0, 'const'),
 Text(14, 0, 'temp')])
```