Multiple linear regression model for Bike sharing

July 30, 2022

1 PROBLEM GOAL:

Develop a model to find the variables that are significant in the demand for shared bikes with the available independent variables and report appropriate metrics of your model evaluation.

2 Data link:

https://docs.google.com/spreadsheets/d/1WieH-IPBXkKPj46XCEsjAgShaRdZ3A9BTN2zz27jSN4/edit?usp=sha

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

3 Reading data:

[2]:
```
bike = pd.DataFrame(pd.read_csv("day.csv"))
```

[3]:
```
bike.head()
```

[3]:
```
   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
```

4 Check the information of the dataset:

[4]:
```
bike.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
```

5 To check the descriptive information of the dataset:

[5]:
```
bike.describe()
```

[5]:
```
          instant      season          yr        mnth     holiday     weekday  \
count  730.000000  730.000000  730.000000  730.000000  730.000000  730.000000
mean   365.500000    2.498630    0.500000    6.526027    0.028767    2.995890
std    210.877136    1.110184    0.500343    3.450215    0.167266    2.000339
min      1.000000    1.000000    0.000000    1.000000    0.000000    0.000000
25%    183.250000    2.000000    0.000000    4.000000    0.000000    1.000000
50%    365.500000    3.000000    0.500000    7.000000    0.000000    3.000000
75%    547.750000    3.000000    1.000000   10.000000    0.000000    5.000000
```

```
max    730.000000    4.000000    1.000000   12.000000    1.000000    6.000000

       workingday  weathersit        temp       atemp         hum   windspeed  \
count  730.000000  730.000000  730.000000  730.000000  730.000000  730.000000
mean     0.690411    1.394521   20.319259   23.726322   62.765175   12.763620
std      0.462641    0.544807    7.506729    8.150308   14.237589    5.195841
min      0.000000    1.000000    2.424346    3.953480    0.000000    1.500244
25%      0.000000    1.000000   13.811885   16.889713   52.000000    9.041650
50%      1.000000    1.000000   20.465826   24.368225   62.625000   12.125325
75%      1.000000    2.000000   26.880615   30.445775   72.989575   15.625589
max      1.000000    3.000000   35.328347   42.044800   97.250000   34.000021

            casual   registered          cnt
count   730.000000   730.000000   730.000000
mean    849.249315  3658.757534  4508.006849
std     686.479875  1559.758728  1936.011647
min       2.000000    20.000000    22.000000
25%     316.250000  2502.250000  3169.750000
50%     717.000000  3664.500000  4548.500000
75%    1096.500000  4783.250000  5966.000000
max    3410.000000  6946.000000  8714.000000
```

6 To check the shape of the data:

[6]:
```
print(bike.shape)
```

```
(730, 16)
```

[7]:
```
bike.columns
```

[7]:
```
Index(['instant', 'dteday', 'season', 'yr', 'mnth', 'holiday', 'weekday',
       'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',
       'casual', 'registered', 'cnt'],
      dtype='object')
```

7 Drop columns that are not useful for analysis:

Based on the high level look at the data and the data dictionary, the following variables can be removed from further analysis:

instant : Its only an index value

dteday : This has the date, Since we already have seperate columns for 'year' & 'month',hence, we could live without this column.

casual & registered : Both these columns contains the count of bike booked by different categories of customers. Since our objective is to find the total count of bikes and not by specific category, we will ignore these two columns. More over, we have created a new variable to have the ratio of these customer types.

We will save the new dataframe as bike_new, so that the original dataset is preserved for any future analysis/validation

[8]:
```
bike_new=bike[['season', 'yr', 'mnth', 'holiday', 'weekday',
       'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',
       'cnt']]
```

[9]:
```
bike_new.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    int64
 1   yr          730 non-null    int64
 2   mnth        730 non-null    int64
 3   holiday     730 non-null    int64
 4   weekday     730 non-null    int64
 5   workingday  730 non-null    int64
 6   weathersit  730 non-null    int64
 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: float64(4), int64(8)
memory usage: 68.6 KB
```

[10]:
```
bike_new['season']=bike_new['season'].astype('category')
bike_new['weathersit']=bike_new['weathersit'].astype('category')
bike_new['mnth']=bike_new['mnth'].astype('category')
bike_new['weekday']=bike_new['weekday'].astype('category')
```

```
C:\Users\MAHA\AppData\Local\Temp/ipykernel_3252/468937576.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  bike_new['season']=bike_new['season'].astype('category')
C:\Users\MAHA\AppData\Local\Temp/ipykernel_3252/468937576.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
```

```
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  bike_new['weathersit']=bike_new['weathersit'].astype('category')
C:\Users\MAHA\AppData\Local\Temp/ipykernel_3252/468937576.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  bike_new['mnth']=bike_new['mnth'].astype('category')
C:\Users\MAHA\AppData\Local\Temp/ipykernel_3252/468937576.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  bike_new['weekday']=bike_new['weekday'].astype('category')
```

[11]:

```
bike_new.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    int64
 2   mnth        730 non-null    category
 3   holiday     730 non-null    int64
 4   weekday     730 non-null    category
 5   workingday  730 non-null    int64
 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(4), float64(4), int64(4)
memory usage: 49.7 KB
```

8 SPLITTING THE DATA

Splitting the data to Train and Test: - We will now split the data into TRAIN and TEST (70:30 ratio)

We will use train_test_split method from sklearn package for this

[12]:
```
from sklearn.model_selection import train_test_split
np.random.seed(0)
df_train, df_test = train_test_split(bike_new, train_size = 0.70, test_size = 0.
↪30, random_state = 333)
```

[13]:
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 510 entries, 483 to 366
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   season      510 non-null    category
 1   yr          510 non-null    int64
 2   mnth        510 non-null    category
 3   holiday     510 non-null    int64
 4   weekday     510 non-null    category
 5   workingday  510 non-null    int64
 6   weathersit  510 non-null    category
 7   temp        510 non-null    float64
 8   atemp       510 non-null    float64
 9   hum         510 non-null    float64
 10  windspeed   510 non-null    float64
 11  cnt         510 non-null    int64
dtypes: category(4), float64(4), int64(4)
memory usage: 38.9 KB
```

[14]:
```
df_train.shape
```

[14]:
```
(510, 12)
```

[15]:
```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 219 entries, 22 to 313
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   season      219 non-null    category
 1   yr          219 non-null    int64
 2   mnth        219 non-null    category
 3   holiday     219 non-null    int64
```

```
 4   weekday     219 non-null    category
 5   workingday  219 non-null    int64
 6   weathersit  219 non-null    category
 7   temp        219 non-null    float64
 8   atemp       219 non-null    float64
 9   hum         219 non-null    float64
 10  windspeed   219 non-null    float64
 11  cnt         219 non-null    int64
dtypes: category(4), float64(4), int64(4)
memory usage: 17.3 KB
```

[16]:
```python
#check the shape before splitting
df_test.shape
```

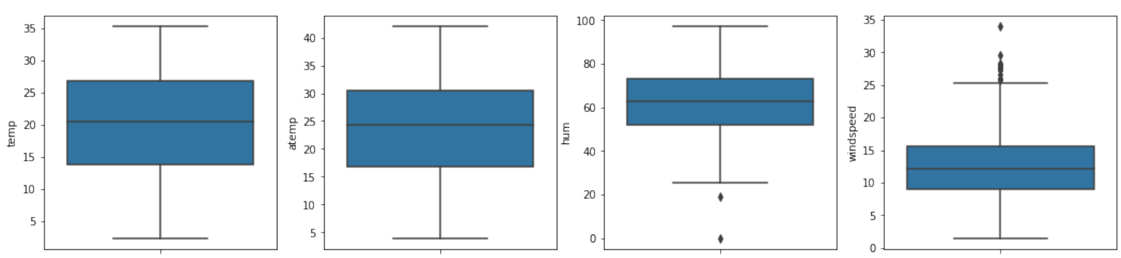
[16]:
```
(219, 12)
```

[17]:
```python
#check the train set columns
df_train.columns
```

[17]:
```
Index(['season', 'yr', 'mnth', 'holiday', 'weekday', 'workingday',
       'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'cnt'],
      dtype='object')
```

9 Outlier checking

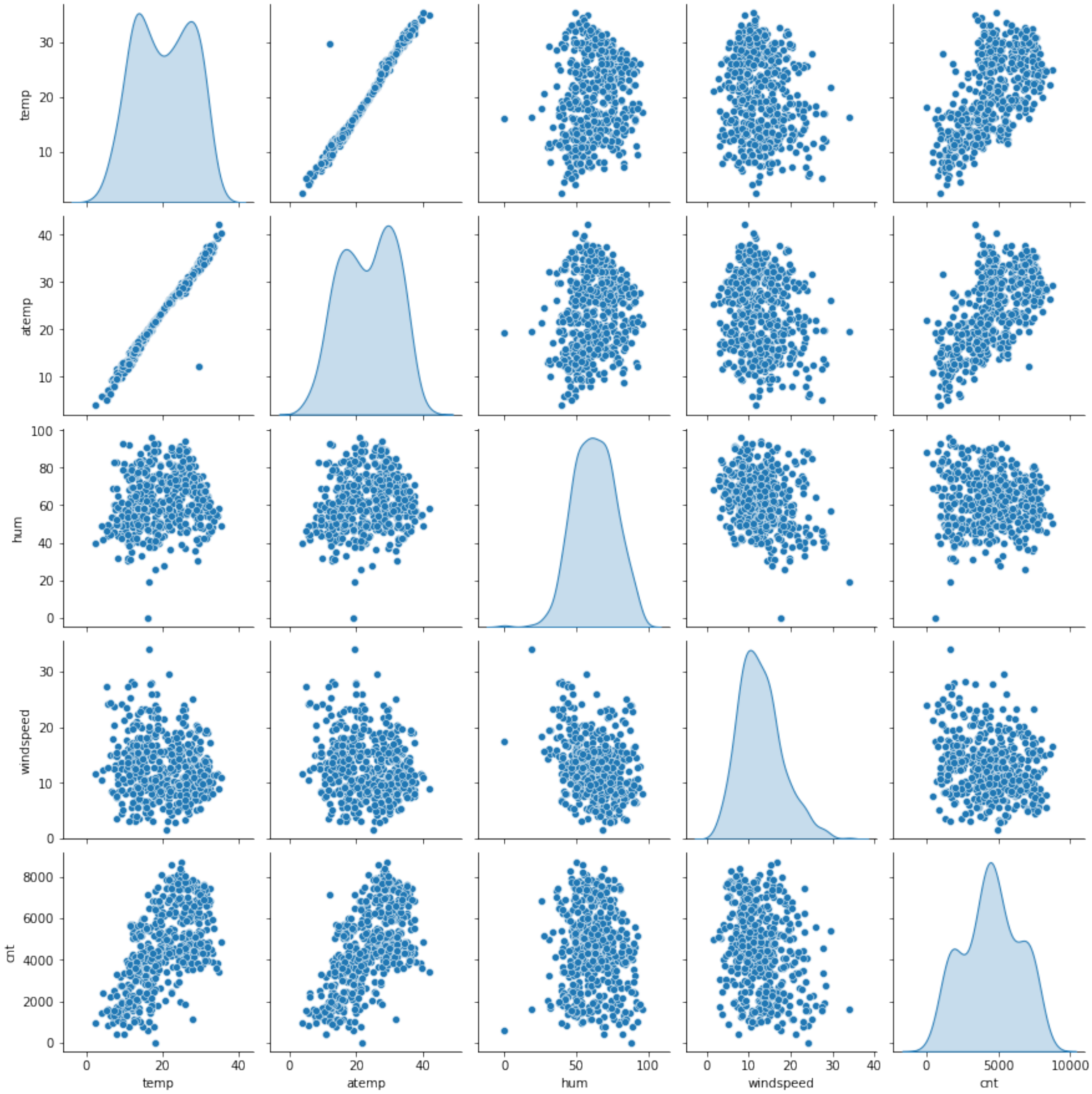
[18]:
```python
#Draw boxplot for indepent variables
cols = ['temp', 'atemp', 'hum', 'windspeed']
plt.figure(figsize=(18,4))

i = 1
for col in cols:
    plt.subplot(1,4,i)
    sns.boxplot(y=col, data=bike_new)
    i+=1
```



10 Draw pair Plots to check the linear relationship

[19]:

```
#Draw pairplots for continuous numeric variables using seaborn
bike_num=df_train[[ 'temp', 'atemp', 'hum', 'windspeed','cnt']]
sns.pairplot(bike_num, diag_kind='kde')
plt.show()
```



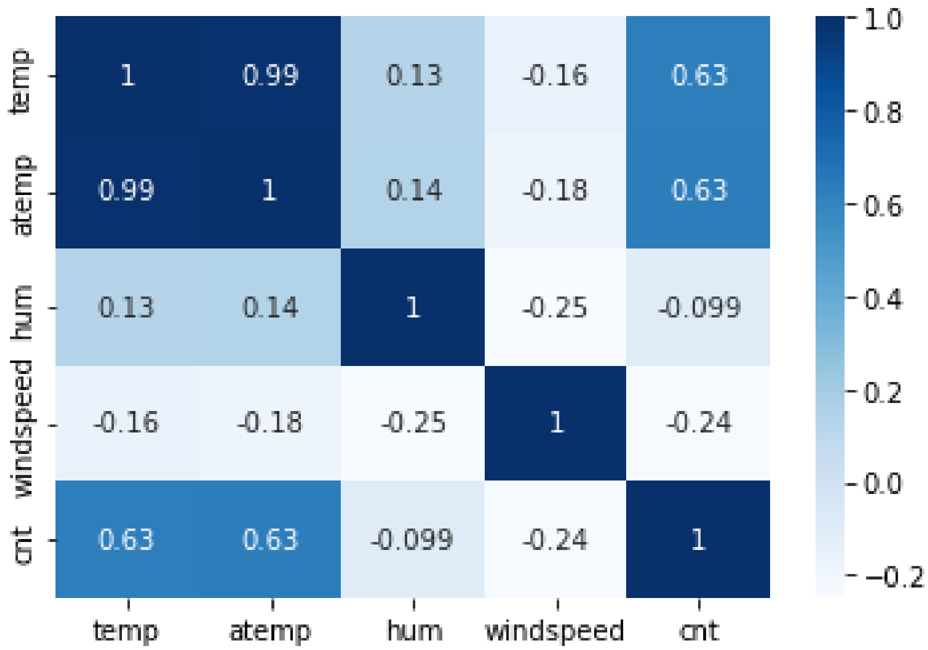
11 Find the Correlation between the Numerical Variable

[20]:

```
bike_new.corr()
```

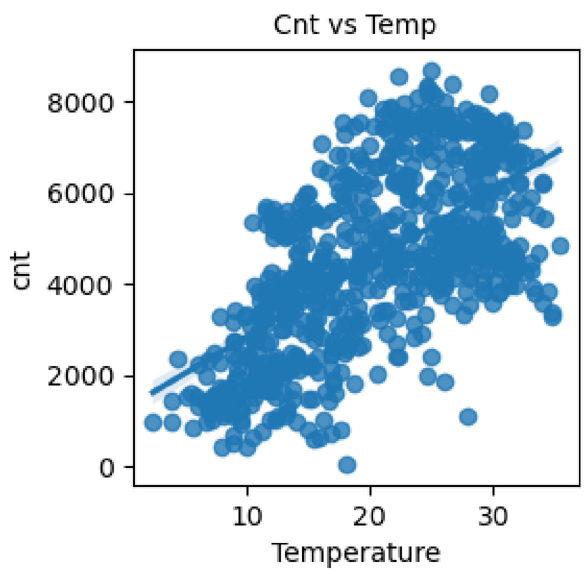
[20]:
```
                  yr   holiday  workingday      temp     atemp       hum  \
yr          1.000000  0.008195   -0.011852  0.048789  0.047215 -0.112547
holiday     0.008195  1.000000   -0.257009 -0.028764 -0.032703 -0.015662
workingday -0.011852 -0.257009    1.000000  0.002044  0.010657  0.053770
temp        0.048789 -0.028764    0.002044  1.000000  0.991696  0.128565
atemp       0.047215 -0.032703    0.010657  0.991696  1.000000  0.141512
hum        -0.112547 -0.015662    0.053770  0.128565  0.141512  1.000000
windspeed  -0.011624  0.006257   -0.002453 -0.158186 -0.183876 -0.248506
cnt         0.569728 -0.068764   -0.027640  0.627044  0.630685 -0.098543

            windspeed       cnt
yr          -0.011624  0.569728
holiday      0.006257 -0.068764
workingday  -0.002453 -0.027640
temp        -0.158186  0.627044
atemp       -0.183876  0.630685
hum         -0.248506 -0.098543
windspeed    1.000000 -0.235132
cnt         -0.235132  1.000000
```

[21]:
```
#Draw Headmap
sns.heatmap(bike_new[['temp','atemp','hum','windspeed','cnt']].corr(),␣
 ↪cmap='Blues', annot = True)
plt.show()
```

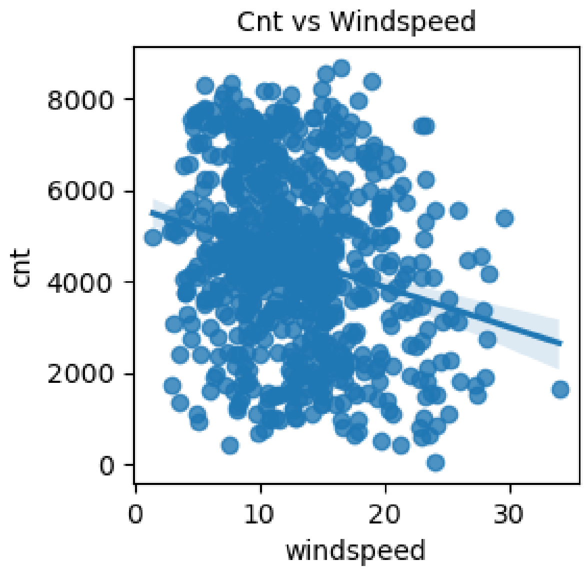


12 Analysing Categorical Variabels with target variables

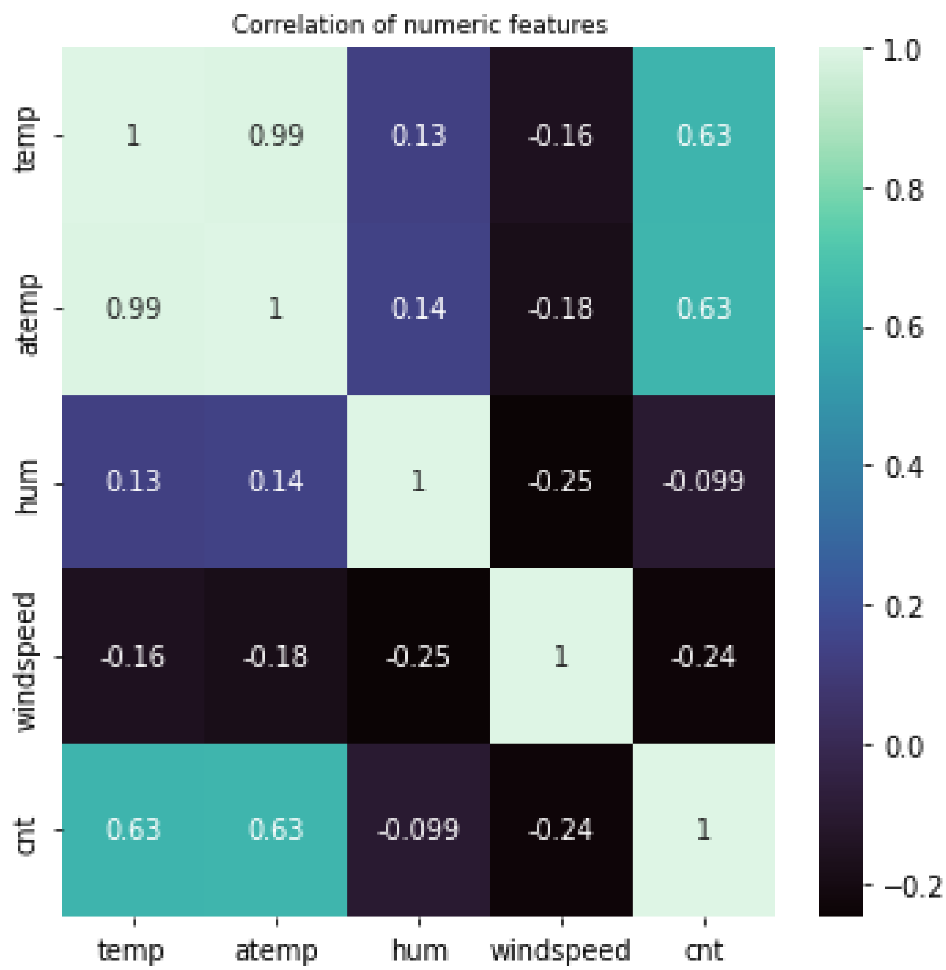
[22]:
```
plt.figure(figsize=(3,3),dpi=100)
plt.title("Cnt vs Temp",fontsize=10)
sns.regplot(data=bike_new,y="cnt",x="temp")
plt.xlabel("Temperature")
plt.show()
```



[23]:
```
plt.figure(figsize=(3,3),dpi=100)
plt.title("Cnt vs Windspeed",fontsize=10)
sns.regplot(data=bike_new,y="cnt",x="windspeed")
plt.show()
```



[24]:
```
num_features = ["temp","atemp","hum","windspeed","cnt"]
plt.figure(figsize=(6,6),dpi=70)
plt.title("Correlation of numeric features",fontsize=9)
sns.heatmap(bike_new[num_features].corr(),annot= True,cmap="mako")
plt.show()
```



13 Data Preparation for Linear Regression

[25]:
```
from sklearn.preprocessing import MinMaxScaler
```

[26]:
```
scaler = MinMaxScaler()
```

[27]:
```
df_train.head()
```

[27]:
```
     season  yr  mnth  holiday  weekday  workingday  weathersit       temp  \
483       2   1     4        0        1           1           1  18.791653
650       4   1    10        0        0           0           1  16.126653
212       3   0     8        0        3           1           1  31.638347
714       4   1    12        0        1           1           2  14.862500
8         1   0     1        0        2           1           1   5.671653
```

```
        atemp      hum  windspeed   cnt
483  22.50605  58.7083   7.832836  6304
650  19.56980  49.4583   9.791514  7109
212  35.16460  55.0833  10.500039  4266
714  18.49690  83.8750   6.749714  3786
8     5.80875  43.4167  24.250650   822
```

[28]:
```python
df_train.columns
```

[28]:
```
Index(['season', 'yr', 'mnth', 'holiday', 'weekday', 'workingday',
       'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'cnt'],
      dtype='object')
```

[29]:
```python
num_vars = ['temp', 'atemp', 'hum', 'windspeed','cnt']
df_train[num_vars] = scaler.fit_transform(df_train[num_vars])
```

[30]:
```python
df_train.head()
```

[30]:
```
     season  yr mnth  holiday weekday  workingday weathersit      temp  \
483       2   1    4        0       1           1          1  0.497426
650       4   1   10        0       0           0          1  0.416433
212       3   0    8        0       3           1          1  0.887856
714       4   1   12        0       1           1          2  0.378013
8         1   0    1        0       2           1          1  0.098690

        atemp       hum  windspeed       cnt
483  0.487055  0.609956   0.194850  0.722734
650  0.409971  0.513852   0.255118  0.815347
212  0.819376  0.572294   0.276919  0.488265
714  0.381804  0.871429   0.161523  0.433042
8    0.048706  0.451083   0.700017  0.092039
```

[31]:
```python
df_train.describe()
```

[31]:
```
               yr     holiday  workingday        temp       atemp         hum  \
count  510.000000  510.000000  510.000000  510.000000  510.000000  510.000000
mean     0.501961    0.023529    0.692157    0.540901    0.515631    0.647390
std      0.500487    0.151726    0.462054    0.227898    0.213626    0.149722
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000    0.000000    0.000000    0.343228    0.335807    0.536147
50%      1.000000    0.000000    1.000000    0.540519    0.525578    0.646367
75%      1.000000    0.000000    1.000000    0.740406    0.692378    0.757900
max      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000

        windspeed         cnt
count  510.000000  510.000000
mean     0.346318    0.515144
```

```
std      0.160266    0.224281
min      0.000000    0.000000
25%      0.230784    0.359468
50%      0.325635    0.516337
75%      0.434287    0.685861
max      1.000000    1.000000
```

[32]:
```
y_train = df_train.pop('cnt')
X_train = df_train
```

14 Use RFE to eliminate some columns

[33]:
```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression
```

[34]:
```
lm = LinearRegression()
lm.fit(X_train, y_train)
rfe=RFE(lm,n_features_to_select=15)
rfe = rfe.fit(X_train, y_train)
```

[35]:
```
list(zip(X_train.columns,rfe.support_,rfe.ranking_))
```

[35]:
```
[('season', True, 1),
 ('yr', True, 1),
 ('mnth', True, 1),
 ('holiday', True, 1),
 ('weekday', True, 1),
 ('workingday', True, 1),
 ('weathersit', True, 1),
 ('temp', True, 1),
 ('atemp', True, 1),
 ('hum', True, 1),
 ('windspeed', True, 1)]
```

[36]:
```
col = X_train.columns[rfe.support_]
col
```

[36]:
```
Index(['season', 'yr', 'mnth', 'holiday', 'weekday', 'workingday',
       'weathersit', 'temp', 'atemp', 'hum', 'windspeed'],
      dtype='object')
```

[37]:
```
X_train.columns[~rfe.support_]
```

[37]:
```
Index([], dtype='object')
```

[38]:
```
X_train_rfe = X_train[col]
```

Function to calculate VIFs and print them

[39]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif = pd.DataFrame()
vif['Features'] = X_train_rfe.columns
vif['VIF'] = [variance_inflation_factor(X_train_rfe.values, i) for i in␣
 ↪range(X_train_rfe.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

[39]:

	Features	VIF
8	atemp	347.92
7	temp	335.26
9	hum	26.45
0	season	21.47
2	mnth	15.79
6	weathersit	12.91
10	windspeed	4.54
5	workingday	3.36
4	weekday	3.13
1	yr	1.98
3	holiday	1.09

[40]:
```
import statsmodels.api as sm
X_train_lm1 = sm.add_constant(X_train_rfe)
lr1 = sm.OLS(y_train, X_train_lm1).fit()
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\statsmodels\tsa\tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[41]:
```
lr1.params
```

[41]:
```
const         0.271175
season        0.058342
yr            0.230654
mnth         -0.005410
holiday      -0.077059
weekday       0.003025
workingday   -0.029959
weathersit   -0.070101
temp          0.227614
atemp         0.279620
hum          -0.115018
windspeed    -0.184839
dtype: float64
```

15 Model 1 - Start with all variables selected by RFE

[42]:
```
print(lr1.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.796
Model:                            OLS   Adj. R-squared:                  0.791
Method:                 Least Squares   F-statistic:                     176.3
Date:                Sat, 30 Jul 2022   Prob (F-statistic):          8.16e-164
Time:                        21:59:34   Log-Likelihood:                 444.17
No. Observations:                 510   AIC:                            -864.3
Df Residuals:                     498   BIC:                            -813.5
Df Model:                          11
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2712      0.030      8.921      0.000       0.211       0.331
season         0.0583      0.008      7.509      0.000       0.043       0.074
yr             0.2307      0.009     25.102      0.000       0.213       0.249
mnth          -0.0054      0.002     -2.205      0.028      -0.010      -0.001
holiday       -0.0771      0.031     -2.491      0.013      -0.138      -0.016
weekday        0.0030      0.002      1.327      0.185      -0.001       0.008
workingday    -0.0300      0.010     -2.931      0.004      -0.050      -0.010
weathersit    -0.0701      0.011     -6.414      0.000      -0.092      -0.049
temp           0.2276      0.142      1.602      0.110      -0.051       0.507
atemp          0.2796      0.153      1.832      0.067      -0.020       0.579
hum           -0.1150      0.042     -2.743      0.006      -0.197      -0.033
windspeed     -0.1848      0.031     -5.902      0.000      -0.246      -0.123
==============================================================================
Omnibus:                       59.353   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              119.364
Skew:                          -0.671   Prob(JB):                     1.20e-26
Kurtosis:                       4.953   Cond. No.                         391.
==============================================================================

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

*Removing the variable 'atemp' based on its High p-value & High VIF

[43]:
```
X_train_new = X_train_rfe.drop(["atemp"], axis = 1)
```

VIF Check

[44]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif = pd.DataFrame()
vif['Features'] = X_train_new.columns
vif['VIF'] = [variance_inflation_factor(X_train_new.values, i) for i in␣
↪range(X_train_new.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

[44]:

	Features	VIF
8	hum	25.94
0	season	21.40
2	mnth	15.78
6	weathersit	12.89
7	temp	7.87
9	windspeed	4.44
5	workingday	3.34
4	weekday	3.13
1	yr	1.98
3	holiday	1.09

[45]:

```
X_train_lm2 = sm.add_constant(X_train_new)
lr2 = sm.OLS(y_train, X_train_lm2).fit()
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\statsmodels\tsa\tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[46]:

```
lr2.params
```

[46]:

```
const          0.277236
season         0.058907
yr             0.230812
mnth          -0.005525
holiday       -0.077668
weekday        0.002778
workingday    -0.028730
weathersit    -0.070773
temp           0.484720
hum           -0.110721
windspeed     -0.195338
dtype: float64
```

16 Model-2 bulid after removing atemp:

[47]:
```
print(lr2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.794
Model:                            OLS   Adj. R-squared:                  0.790
Method:                 Least Squares   F-statistic:                     192.7
Date:                Sat, 30 Jul 2022   Prob (F-statistic):          3.03e-164
Time:                        21:59:35   Log-Likelihood:                 442.45
No. Observations:                 510   AIC:                            -862.9
Df Residuals:                     499   BIC:                            -816.3
Df Model:                          10
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2772      0.030      9.153      0.000       0.218       0.337
season         0.0589      0.008      7.570      0.000       0.044       0.074
yr             0.2308      0.009     25.061      0.000       0.213       0.249
mnth          -0.0055      0.002     -2.248      0.025      -0.010      -0.001
holiday       -0.0777      0.031     -2.505      0.013      -0.139      -0.017
weekday        0.0028      0.002      1.218      0.224      -0.002       0.007
workingday    -0.0287      0.010     -2.810      0.005      -0.049      -0.009
weathersit    -0.0708      0.011     -6.464      0.000      -0.092      -0.049
temp           0.4847      0.022     21.899      0.000       0.441       0.528
hum           -0.1107      0.042     -2.639      0.009      -0.193      -0.028
windspeed     -0.1953      0.031     -6.330      0.000      -0.256      -0.135
==============================================================================
Omnibus:                       52.728   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              103.061
Skew:                          -0.613   Prob(JB):                     4.17e-23
Kurtosis:                       4.829   Cond. No.                         92.6
==============================================================================

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

1) Removing the variable 'hum' based on its Very High 'VIF' value. 2)Even though the VIF of hum is second highest, we decided to drop 'hum' and not 'temp' based on general knowledge that temperature can be an important factor for a business like bike rentals, and wanted to retain 'temp'.

[48]:
```
X_train_new = X_train_new.drop(["hum"], axis = 1)
```

VIF check

[49]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif = pd.DataFrame()
vif['Features'] = X_train_new.columns
vif['VIF'] = [variance_inflation_factor(X_train_new.values, i) for i in␣
 ↪range(X_train_new.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

[49]:

	Features	VIF
0	season	21.36
2	mnth	15.37
7	temp	6.75
6	weathersit	5.95
8	windspeed	4.43
5	workingday	3.09
4	weekday	3.08
1	yr	1.98
3	holiday	1.08

[50]:
```
X_train_lm3 = sm.add_constant(X_train_new)
lr3 = sm.OLS(y_train, X_train_lm3).fit()
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\statsmodels\tsa\tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[51]:
```
lr3.params
```

[51]:
```
const          0.232782
season         0.059786
yr             0.233247
mnth          -0.006415
holiday       -0.079728
weekday        0.002890
workingday    -0.031422
weathersit    -0.088825
temp           0.472745
windspeed     -0.171276
dtype: float64
```

17 Model-3 bulid after removing hum:

[52]:
```
print(lr3.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.791
Model:                            OLS   Adj. R-squared:                  0.788
Method:                 Least Squares   F-statistic:                     210.8
Date:                Sat, 30 Jul 2022   Prob (F-statistic):          6.34e-164
Time:                        21:59:36   Log-Likelihood:                 438.92
No. Observations:                 510   AIC:                            -857.8
Df Residuals:                     500   BIC:                            -815.5
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2328      0.025      9.193      0.000       0.183       0.283
season         0.0598      0.008      7.644      0.000       0.044       0.075
yr             0.2332      0.009     25.303      0.000       0.215       0.251
mnth          -0.0064      0.002     -2.619      0.009      -0.011      -0.002
holiday       -0.0797      0.031     -2.557      0.011      -0.141      -0.018
weekday        0.0029      0.002      1.260      0.208      -0.002       0.007
workingday    -0.0314      0.010     -3.070      0.002      -0.052      -0.011
weathersit    -0.0888      0.009    -10.329      0.000      -0.106      -0.072
temp           0.4727      0.022     21.692      0.000       0.430       0.516
windspeed     -0.1713      0.030     -5.775      0.000      -0.230      -0.113
==============================================================================
Omnibus:                       50.579   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               98.220
Skew:                          -0.593   Prob(JB):                     4.70e-22
Kurtosis:                       4.793   Cond. No.                         65.1
==============================================================================

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

*Removing the variable 'season' based on its Very High 'VIF' value

[53]:
```
X_train_new = X_train_new.drop(["season"], axis = 1)
```

VIF check

[54]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif = pd.DataFrame()
vif['Features'] = X_train_new.columns
vif['VIF'] = [variance_inflation_factor(X_train_new.values, i) for i in range(X_train_new.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
```

```
vif
```

[54]:

```
      Features   VIF
6         temp  5.95
5   weathersit  5.91
1         mnth  4.51
7    windspeed  4.43
3      weekday  3.08
4   workingday  3.08
0           yr  1.97
2      holiday  1.07
```

[55]:

```
X_train_lm4 = sm.add_constant(X_train_new)
lr4 = sm.OLS(y_train, X_train_lm4).fit()
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\statsmodels\tsa\tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[56]:

```
lr4.params
```

[56]:

```
const         0.259831
yr            0.235243
mnth          0.008893
holiday      -0.065440
weekday       0.002748
workingday   -0.030299
weathersit   -0.088357
temp          0.517163
windspeed    -0.182014
dtype: float64
```

18 Model-4 build after removing Season:

[57]:

```
print(lr4.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.767
Model:                            OLS   Adj. R-squared:                  0.763
Method:                 Least Squares   F-statistic:                     206.2
Date:                Sat, 30 Jul 2022   Prob (F-statistic):          3.84e-153
Time:                        21:59:36   Log-Likelihood:                 410.73
No. Observations:                 510   AIC:                            -803.5
Df Residuals:                     501   BIC:                            -765.4
```

```
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2598      0.026      9.816      0.000       0.208       0.312
yr             0.2352      0.010     24.182      0.000       0.216       0.254
mnth           0.0089      0.001      5.973      0.000       0.006       0.012
holiday       -0.0654      0.033     -1.992      0.047      -0.130      -0.001
weekday        0.0027      0.002      1.135      0.257      -0.002       0.008
workingday    -0.0303      0.011     -2.804      0.005      -0.052      -0.009
weathersit    -0.0884      0.009     -9.732      0.000      -0.106      -0.071
temp           0.5172      0.022     23.321      0.000       0.474       0.561
windspeed     -0.1820      0.031     -5.819      0.000      -0.243      -0.121
==============================================================================
Omnibus:                       44.143   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               68.248
Skew:                          -0.604   Prob(JB):                     1.51e-15
Kurtosis:                       4.324   Cond. No.                         61.8
==============================================================================

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

19 Final Model Interpretation

Hypothesis Testing:

Hypothesis testing states that:

H0:B1=B2=…=Bn=0

H1: at least one Bi!=0

lr4 model coefficient values const 0.259831 yr 0.235243 mnth 0.008893 holiday -0.065440 weekday 0.002748 workingday -0.030299 weathersit -0.088357 temp 0.517163 windspeed -0.182014 dtype: float64

20 Interpret

From the lr6 model summary, it is evident that all our coefficients are not equal to zerowhich means We REJECT the NULL HYPOTHESIS

The equation of best fitted surface based on model lr4:

$cnt = 0.2598 + (yr \times 0.2352) + (mnth \times 0.0089)$ - $(workingday \times 0.0303) - (weathersit \times 0.0884) + (temp \times 0.5172)$ - $(windspeed \times 0.1820)$

Interpretation of Coefficients: yr: A coefficient value of '0.2352' indicated that a unit increase in yr variable, increases the bike hire numbers by 0.2352 units.

mnth: A coefficient value of '0.0089' indicated that a unit increase in mnth variable, increase the bike hire numbers by 0.0089 units.

yr: A coefficient value of '0.2308' indicated that a unit increase in yr variable, increases the bike hire numbers by 0.2308 units.

workingday: A coefficient value of '-0.0303' indicated that a unit increase in workingday variable decrease the bike hire numbers by 0.0303 units.

weathersit: A coefficient value of '-0.0884' indicated that, a unit increase in windspeed variable decreases the bike hire numbers by 0.0884 units.

temp: A coefficient value of '0.5172' indicated that, a unit increase in workingday variable increases the bike hire numbers by 0.5172 units.

windspeed: A coefficient value of '-0.1820' indicated that, a unit increase in workingday variable decrease the bike hire numbers by 0.1820 units.

21 Error terms are normally distributed with mean zero (not X, Y)

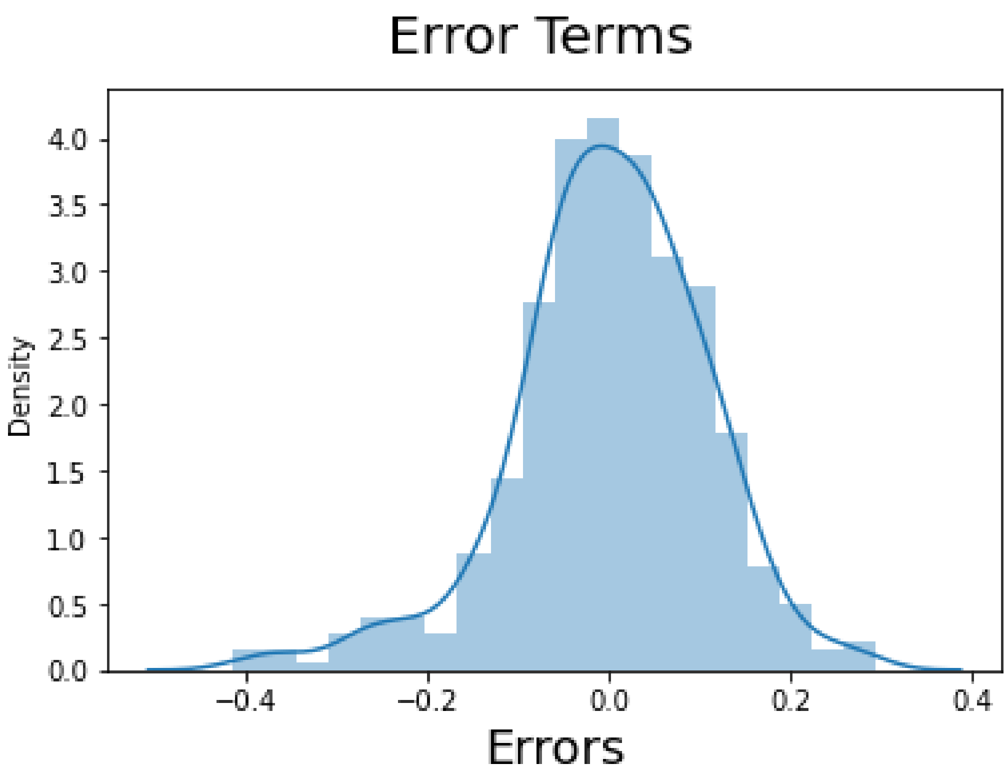
Residual Analysis Of Training Data

[58]:
```
y_train_pred = lr4.predict(X_train_lm4)
```

[59]:
```
res = y_train-y_train_pred
#plot histogram error term
fig = plt.figure()
sns.distplot((res), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)
plt.xlabel('Errors', fontsize = 18)
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[59]:
```
Text(0.5, 0, 'Errors')
```



22 Interpret:

From the above histogram, we could see that the Residuals are normally distributed. Hence our assumption for Linear Regression is valid.

23 MAKING PREDICTION USING FINAL MODEL

Now that we have fitted the model and checked the assumptions, it's time to go ahead and make predictions using the final model (lr4)

Applying the scaling on the test sets

[60]:
```
num_vars = ['temp', 'atemp', 'hum', 'windspeed','cnt']
df_test[num_vars] = scaler.transform(df_test[num_vars])
```

[61]:
```
df_test.head()
```

[61]:
```
     season  yr mnth  holiday weekday  workingday weathersit      temp  \
22        1   0    1        0       2           1          1  0.046591
468       2   1    4        0       0           0          1  0.543115
553       3   1    7        0       1           1          1  0.951196
504       2   1    5        0       1           1          1  0.699909
```

```
353       4   0   12        0       4           1          2  0.407087

        atemp       hum  windspeed       cnt
22   0.025950  0.453529   0.462217  0.110907
468  0.536771  0.522511   0.347424  0.855729
553  0.933712  0.596104   0.212829  0.534975
504  0.662746  0.551083   0.478229  0.817648
353  0.416610  0.618615   0.080770  0.428900
```

[62]:
```
df_test.describe()
```

[62]:
```
               yr     holiday  workingday        temp       atemp         hum  \
count  219.000000  219.000000  219.000000  219.000000  219.000000  219.000000
mean     0.493151    0.041096    0.689498    0.551225    0.527528    0.662567
std      0.501098    0.198967    0.463759    0.229463    0.215434    0.143562
min      0.000000    0.000000    0.000000    0.046591    0.025950    0.301299
25%      0.000000    0.000000    0.000000    0.356479    0.348019    0.553031
50%      0.000000    0.000000    1.000000    0.557653    0.549198    0.662338
75%      1.000000    0.000000    1.000000    0.751309    0.709163    0.762338
max      1.000000    1.000000    1.000000    0.984424    0.980934    1.010390

        windspeed         cnt
count  219.000000  219.000000
mean     0.346706    0.518889
std      0.159553    0.219953
min      0.073090    0.055683
25%      0.232689    0.364703
50%      0.328208    0.525771
75%      0.435708    0.676887
max      0.824380    0.963300
```

Dividing into X_test and y_test

[63]:
```
y_test = df_test.pop('cnt')
X_test = df_test
X_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 219 entries, 22 to 313
Data columns (total 11 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   season      219 non-null    category
 1   yr          219 non-null    int64
 2   mnth        219 non-null    category
 3   holiday     219 non-null    int64
 4   weekday     219 non-null    category
 5   workingday  219 non-null    int64
```

```
 6   weathersit  219 non-null    category
 7   temp        219 non-null    float64
 8   atemp       219 non-null    float64
 9   hum         219 non-null    float64
 10  windspeed   219 non-null    float64
dtypes: category(4), float64(4), int64(3)
memory usage: 15.6 KB
```

[64]:
```
col1=X_train_new.columns
X_test=X_test[col1]
X_test_lm4 = sm.add_constant(X_test)
X_test_lm4.info()
```

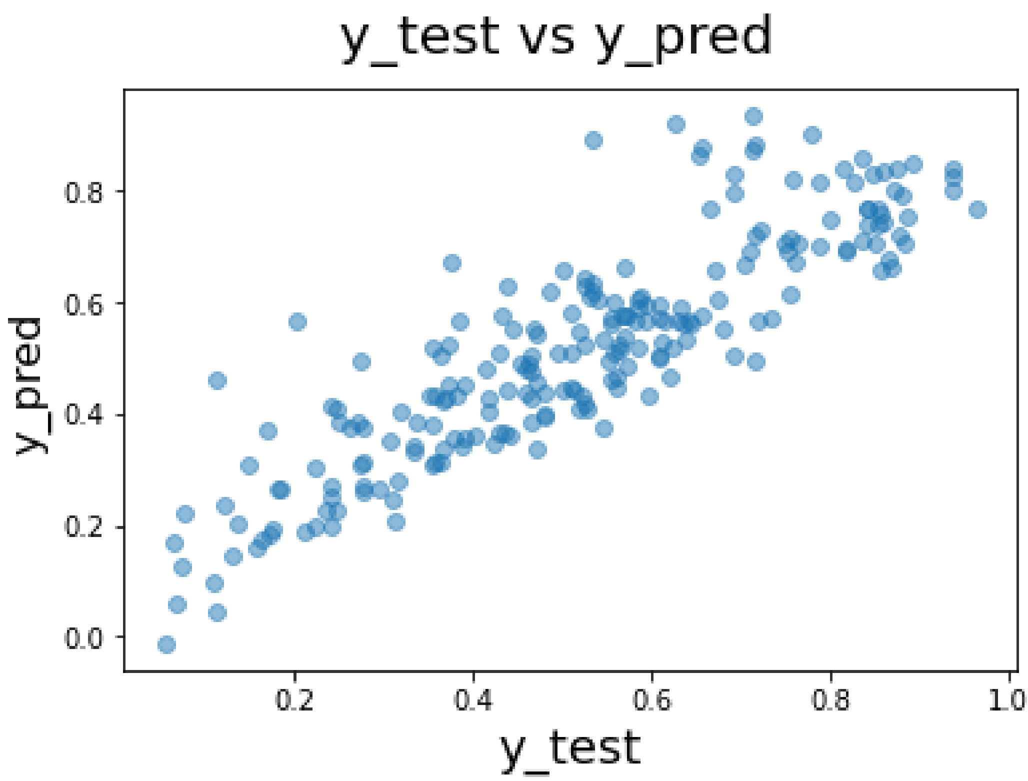
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 219 entries, 22 to 313
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   const       219 non-null    float64
 1   yr          219 non-null    int64
 2   mnth        219 non-null    category
 3   holiday     219 non-null    int64
 4   weekday     219 non-null    category
 5   workingday  219 non-null    int64
 6   weathersit  219 non-null    category
 7   temp        219 non-null    float64
 8   windspeed   219 non-null    float64
dtypes: category(3), float64(3), int64(3)
memory usage: 13.5 KB
```

```
C:\Users\MAHA\anaconda3\lib\site-packages\statsmodels\tsa\tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[65]:
```
y_pred = lr4.predict(X_test_lm4)
```

24 MODEL EVALUATION

[66]:
```
fig = plt.figure()
plt.scatter(y_test, y_pred, alpha=.5)
fig.suptitle('y_test vs y_pred', fontsize = 20)              # Plot heading
plt.xlabel('y_test', fontsize = 18)                          # X-label
plt.ylabel('y_pred', fontsize = 16)
plt.show()
```



25 R^2 Value for TEST

[67]:
```
from sklearn.metrics import r2_score
r2_score(y_test, y_pred)
```

[67]: 0.7658194327343915

26 Adjusted R^2 Value for TEST

[68]:
```
r2=0.7658194327343915
```

[69]:
```
X_test.shape
```

[69]: (219, 8)

[70]:
```
n = X_test.shape[0]
p = X_test.shape[1]
adjusted_r2 = 1-(1-r2)*(n-1)/(n-p-1)
adjusted_r2
```

[70]: 0.7568982682671301

27 Final Result Comparison

Train R^2 :0.767

Train Adjusted R^2 :0.763

Test R^2 :0.766

Test Adjusted R^2 :0.757

This seems to be a really good model that can very well 'Generalize' various datasets.

28 FINAL REPORT

As per our final Model, the top 3 predictor variables that influences the bike booking are:

temp: A coefficient value of '0.5172' indicated that, a unit increase in workingday variable increases the bike hire numbers by 0.5172 units.

yr: A coefficient value of '0.2308' indicated that a unit increase in yr variable, increases the bike hire numbers by 0.2308 units.

windspeed: A coefficient value of '-0.1820' indicated that, a unit increase in workingday variable decrease the bike hire numbers by 0.1820 units.

[ ]:

```

```