# Bike Sharing Assignment

## Problem Statement

A bike-sharing system is a service in which bikes are made available for shared use to individuals on a short term basis for a price or free. Many bike share systems allow people to borrow a bike from a "dock" which is usually computer-controlled wherein the user enters the payment information, and the system unlocks it. This bike can then be returned to another dock belonging to the same system.

A US bike-sharing provider BoomBikes has recently suffered considerable dips in their revenues due to the ongoing Corona pandemic. The company is finding it very difficult to sustain in the current market scenario. So, it has decided to come up with a mindful business plan to be able to accelerate its revenue as soon as the ongoing lockdown comes to an end, and the economy restores to a healthy state.

In such an attempt, BoomBikes aspires to understand the demand for shared bikes among the people after this ongoing quarantine situation ends across the nation due to Covid-19. They have planned this to prepare themselves to cater to the people's needs once the situation gets better all around and stand out from other service providers and make huge profits.

They have contracted a consulting company to understand the factors on which the demand for these shared bikes depends. Specifically, they want to understand the factors affecting the demand for these shared bikes in the American market. The company wants to know:

- Which variables are significant in predicting the demand for shared bikes.
- How well those variables describe the bike demands Based on various meteorological surveys and people's styles, the service provider firm has gathered a large dataset on daily -bike demands across the American market based on some factors.

In [54]:

```
import warnings
warnings.filterwarnings('ignore')

import pandas as pd

# Data visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Data splitting
from sklearn.model_selection import train_test_split

# Feature scaling
from sklearn.preprocessing import MinMaxScaler

# Building a model
import statsmodels.api as sm

# VIF calculation
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Evaluation  (finding R2)
from sklearn.metrics import r2_score
```

In [55]:

```
# Importing the data set
data = pd.read_csv("day.csv")
data.head()
```

Out[55]:

| | instant | dteday | season | yr | mnth | holiday | weekday | workingday | weathersit | temp |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 01-01-2018 | 1 | 0 | 1 | 0 | 6 | 0 | 2 | 14.110847 |
| **1** | 2 | 02-01-2018 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 14.902598 |
| **2** | 3 | 03-01-2018 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 8.050924 |
| **3** | 4 | 04-01-2018 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 8.200000 |
| **4** | 5 | 05-01-2018 | 1 | 0 | 1 | 0 | 3 | 1 | 1 | 9.305237 |

In [ ]:

In [56]:

```
# Summarizing all the details of all variables.
data.describe()
```

Out[56]:

| | instant | season | yr | mnth | holiday | weekday | workingday |
|---|---|---|---|---|---|---|---|
| **count** | 730.000000 | 730.000000 | 730.000000 | 730.000000 | 730.000000 | 730.000000 | 730.000000 |
| **mean** | 365.500000 | 2.498630 | 0.500000 | 6.526027 | 0.028767 | 2.997260 | 0.683562 |
| **std** | 210.877136 | 1.110184 | 0.500343 | 3.450215 | 0.167266 | 2.006161 | 0.465405 |
| **min** | 1.000000 | 1.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 0.000000 |
| **25%** | 183.250000 | 2.000000 | 0.000000 | 4.000000 | 0.000000 | 1.000000 | 0.000000 |
| **50%** | 365.500000 | 3.000000 | 0.500000 | 7.000000 | 0.000000 | 3.000000 | 1.000000 |
| **75%** | 547.750000 | 3.000000 | 1.000000 | 10.000000 | 0.000000 | 5.000000 | 1.000000 |
| **max** | 730.000000 | 4.000000 | 1.000000 | 12.000000 | 1.000000 | 6.000000 | 1.000000 |

In [57]:

```
# Verifying whether there are null values.
data.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: 88.5+ KB
```

# Data visualization

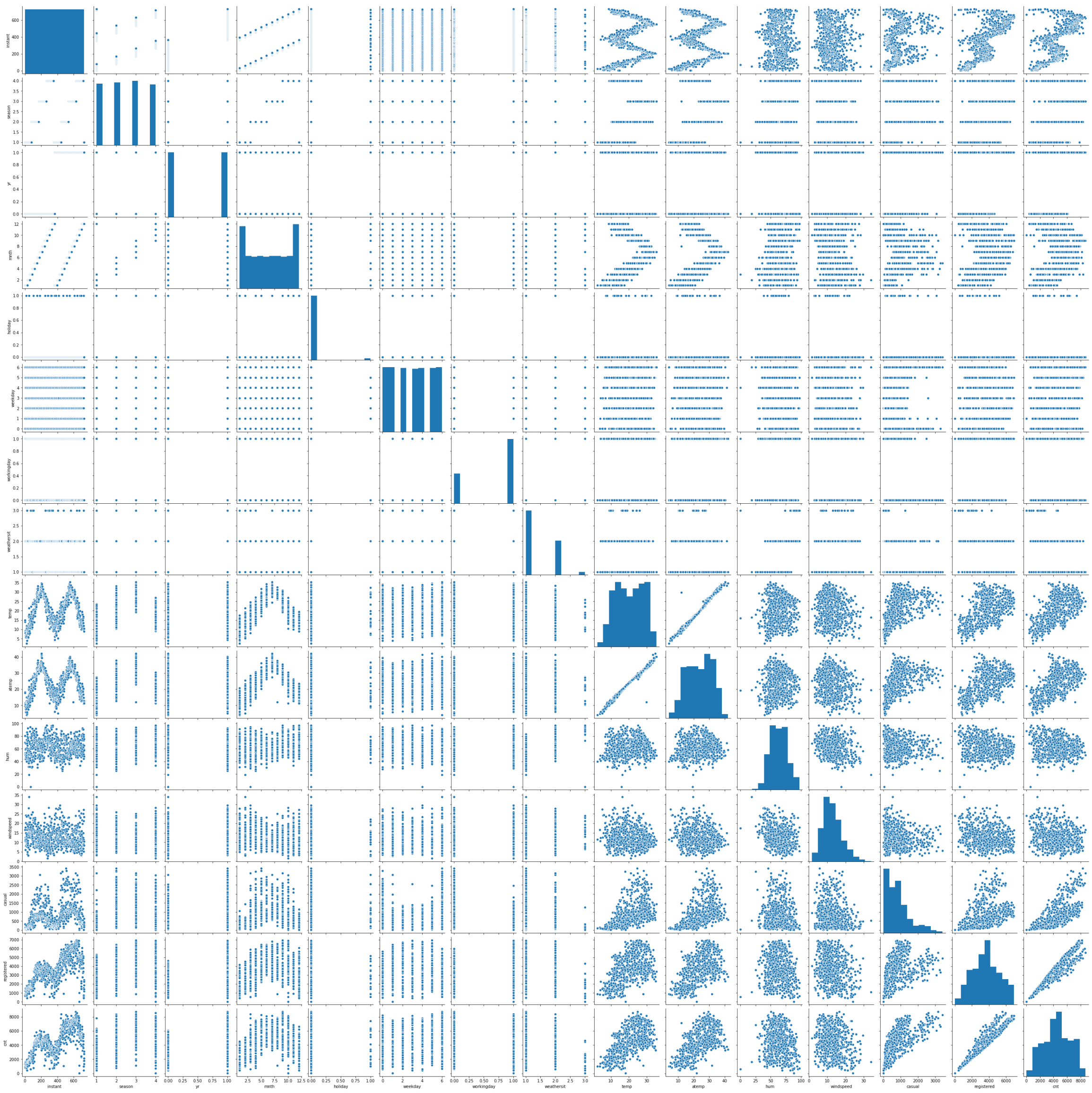
In [58]:

```
# Verifying the relation between all the variables using scatter plots.
plt.figure(figsize=[20,20])
sns.pairplot(data)
```

Out[58]:

```
<seaborn.axisgrid.PairGrid at 0x12f6ca30>
```

```
<Figure size 1440x1440 with 0 Axes>
```



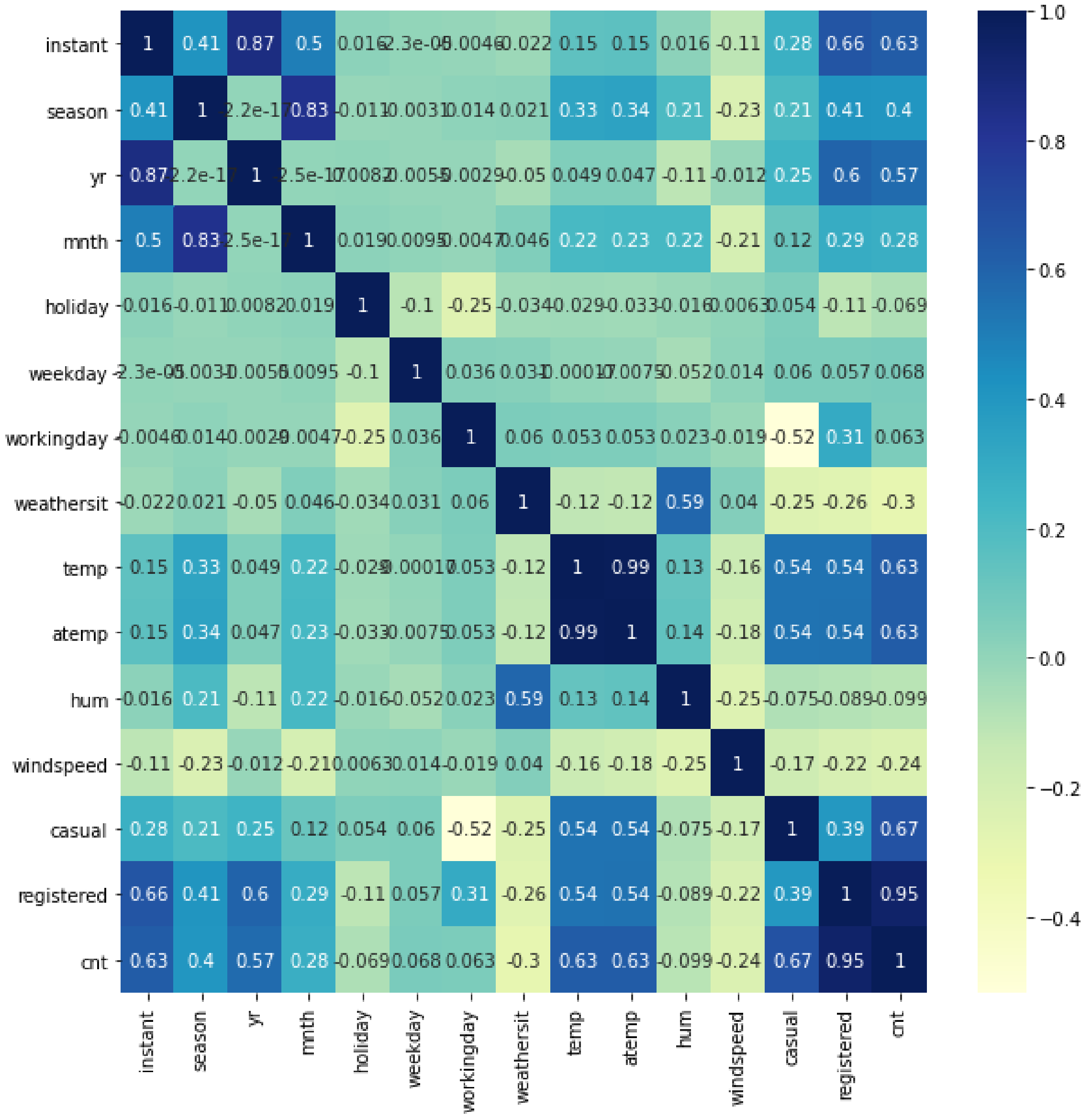
***There are several variables that are categorical in nature when we observe in the plot. All the categorical variables plots are in straight lines. These variables need creating dummy variables.***

In [59]:

```
# Finding the corrilation value of all the variables.
plt.figure(figsize=[10,10])
sns.heatmap(data.corr(), annot=True , cmap='YlGnBu')
```

Out[59]:

```
<AxesSubplot:>
```



=============================================

# Data Preparation

- Dropping unnecessory data
- Dropping highly corrilated data that are derived from the data visualization
- Creating Dummy Variables for categorical variables

## Dropping unnecessory data

There are some variables that does not add importance to the data for future predictions. They are instant, casual, registered, dteday. So dropping them wont effect the predictions.

```
Instant - Does not add any importence.
dteday - This feature is already explained by other features. So no need of it.
casual, registered - These are the type of customers using per day. They also do
nt effect the future predictions
```

In [60]:

```
# Droping the unnecessary variables

data = data.drop(['instant', 'dteday', 'casual', 'registered'], axis=1)
```

## Dropping highly corrilated data that are derived from the data visualization

In the given dataset we have two variables (atemp, temp) that are higly corrilated, i.e: corrilation value is nearly equal to zero. In this kind of situation considering only one variable will be enough. So drop one of them.

In [61]:

```
# Dropping atemp variable

data = data.drop(['atemp'], axis=1)
```

In [62]:

```
data.head()
```

Out[62]:

| | season | yr | mnth | holiday | weekday | workingday | weathersit | temp | hum | windsp |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 6 | 0 | 2 | 14.110847 | 80.5833 | 10.749 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 14.902598 | 69.6087 | 16.652 |
| 2 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 8.050924 | 43.7273 | 16.636 |
| 3 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 8.200000 | 59.0435 | 10.739 |
| 4 | 1 | 0 | 1 | 0 | 3 | 1 | 1 | 9.305237 | 43.6957 | 12.522 |

## Creating Dummy Variables for categorical variables

As we mentioned before in the pairplot section, we need to create dummy variables for some of the categorical variables. They are Season, month, weekday, weathersit.

In [63]:

```
# We first create categorical variables , replacing the numerics with actual data.

data.season = data.season.map({1:'spring', 2:'summer', 3:'fall', 4:'winter'})
data.weekday = data.weekday.map({0:'tuesdata', 1:'wednesdata', 2:'thursdata', 3:'fridat
a',4:'saturdata',5:'sundata',6:'mondata'})
data.mnth = data.mnth.map({1:'Jan', 2:'Feb', 3:'Mar', 4:'Apr', 5:'May', 6:'Jun', 7:'Ju
l', 8:'Aug', 9:'Sep', 10:'Oct', 11:'Nov', 12:'Dec'})
data.weathersit = data.weathersit.map({1:'Clear',2:'Mist+cloudy',3:'Light snow',4:'Heav
y snow'})
```

In [64]:

```
data.head()
```

Out[64]:

| | season | yr | mnth | holiday | weekday | workingday | weathersit | temp | hum | winc |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | spring | 0 | Jan | 0 | mondata | 0 | Mist+cloudy | 14.110847 | 80.5833 | 10. |
| 1 | spring | 0 | Jan | 0 | tuesdata | 0 | Mist+cloudy | 14.902598 | 69.6087 | 16. |
| 2 | spring | 0 | Jan | 0 | wednesdata | 1 | Clear | 8.050924 | 43.7273 | 16. |
| 3 | spring | 0 | Jan | 0 | thursdata | 1 | Clear | 8.200000 | 59.0435 | 10. |
| 4 | spring | 0 | Jan | 0 | fridata | 1 | Clear | 9.305237 | 43.6957 | 12. |

In [65]:

```
# Creating Dummy variables for Season, month, weekday, weathersit features.

dummies = pd.get_dummies(data[['season', 'mnth', 'weekday', 'weathersit']] , drop_first
= True)
dummies.head()
```

Out[65]:

| | season_spring | season_summer | season_winter | mnth_Aug | mnth_Dec | mnth_Feb | mnth_J |
|---|---|---|---|---|---|---|---|
| **0** | 1 | 0 | 0 | 0 | 0 | 0 | |
| **1** | 1 | 0 | 0 | 0 | 0 | 0 | |
| **2** | 1 | 0 | 0 | 0 | 0 | 0 | |
| **3** | 1 | 0 | 0 | 0 | 0 | 0 | |
| **4** | 1 | 0 | 0 | 0 | 0 | 0 | |

5 rows × 22 columns

In [66]:

```
# Now concat these dummies with the actual dataset. And drop the categorical variables.

# Concating dummies to data (axis=1, i.e: adding columns) and dropng the categorical va
lues at the same time.
data = pd.concat([data, dummies], axis=1 ).drop(['season', 'mnth', 'weekday', 'weathers
it'], axis=1)
```

## Splitting the data

In [67]:

```
data_train, data_test = train_test_split(data, train_size = 0.7, test_size = 0.3, rando
m_state = 100)
```

## scaling the features

In [68]:

```
data_train.describe()
```

Out[68]:

| | yr | holiday | workingday | temp | hum | windspeed | cnt |
|---|---|---|---|---|---|---|---|
| **count** | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 |
| **mean** | 0.507843 | 0.025490 | 0.676471 | 20.102429 | 63.112926 | 12.831318 | 4486.382353 |
| **std** | 0.500429 | 0.157763 | 0.468282 | 7.431169 | 14.156632 | 5.291832 | 1952.158739 |
| **min** | 0.000000 | 0.000000 | 0.000000 | 2.424346 | 0.000000 | 2.834381 | 22.000000 |
| **25%** | 0.000000 | 0.000000 | 0.000000 | 13.606865 | 52.270825 | 9.041918 | 3120.000000 |
| **50%** | 1.000000 | 0.000000 | 1.000000 | 20.209597 | 63.437500 | 12.083182 | 4530.000000 |
| **75%** | 1.000000 | 0.000000 | 1.000000 | 26.615847 | 73.250025 | 15.750879 | 5973.500000 |
| **max** | 1.000000 | 1.000000 | 1.000000 | 35.328347 | 97.041700 | 34.000021 | 8714.000000 |

8 rows × 29 columns

In [69]:

```
data_train.head()
```

Out[69]:

| | yr | holiday | workingday | temp | hum | windspeed | cnt | season_spring | season_ |
|---|---|---|---|---|---|---|---|---|---|
| **653** | 1 | 0 | 1 | 19.201653 | 55.8333 | 12.208807 | 7534 | 0 | |
| **576** | 1 | 0 | 1 | 29.246653 | 70.4167 | 11.083475 | 7216 | 0 | |
| **426** | 1 | 0 | 0 | 16.980847 | 62.1250 | 10.792293 | 4066 | 1 | |
| **728** | 1 | 0 | 0 | 10.489153 | 48.3333 | 23.500518 | 1796 | 1 | |
| **482** | 1 | 0 | 0 | 15.443347 | 48.9583 | 8.708325 | 4220 | 0 | |

5 rows × 29 columns

In [70]:

```
# Creating the object for MinMaxScaler()
scaler = MinMaxScaler()

# Using the obj to access the method from MinMax and perform scaling
data_train[data_train.columns] = scaler.fit_transform(data_train[data_train.columns])
```

In [71]:

```
data_train.describe()
```

Out[71]:

| | yr | holiday | workingday | temp | hum | windspeed | cnt |
|---|---|---|---|---|---|---|---|
| **count** | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 | 510.000000 |
| **mean** | 0.507843 | 0.025490 | 0.676471 | 0.537262 | 0.650369 | 0.320768 | 0.513620 |
| **std** | 0.500429 | 0.157763 | 0.468282 | 0.225844 | 0.145882 | 0.169797 | 0.224593 |
| **min** | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **25%** | 0.000000 | 0.000000 | 0.000000 | 0.339853 | 0.538643 | 0.199179 | 0.356420 |
| **50%** | 1.000000 | 0.000000 | 1.000000 | 0.540519 | 0.653714 | 0.296763 | 0.518638 |
| **75%** | 1.000000 | 0.000000 | 1.000000 | 0.735215 | 0.754830 | 0.414447 | 0.684710 |
| **max** | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 |

8 rows × 29 columns

# ============================================

# Building the model

In [72]:

```
# Splitting X and Y
y_train= data_train.pop('cnt')
X_train = data_train
```

In [73]:

```
# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())


# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.853
Model:                            OLS   Adj. R-squared:
0.845
Method:                 Least Squares   F-statistic:                     1
03.8
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          8.74e
-182
Time:                        18:38:24   Log-Likelihood:                 52
7.95
No. Observations:                 510   AIC:                            -9
99.9
Df Residuals:                     482   BIC:                            -8
81.3
Df Model:                          27
Covariance Type:            nonrobust
==========================================================================
================
                             coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                      0.2429      0.035      6.854      0.000
0.173       0.313
yr                         0.2321      0.008     28.820      0.000
0.216       0.248
holiday                    0.0067      0.024      0.278      0.781      -
0.041       0.054
workingday                 0.0937      0.012      7.783      0.000
0.070       0.117
temp                       0.4506      0.046      9.734      0.000
0.360       0.542
hum                       -0.1513      0.038     -3.933      0.000      -
0.227      -0.076
windspeed                 -0.1865      0.026     -7.257      0.000      -
0.237      -0.136
season_spring             -0.0482      0.030     -1.607      0.109      -
0.107       0.011
season_summer              0.0387      0.026      1.478      0.140      -
0.013       0.090
season_winter              0.1058      0.028      3.794      0.000
0.051       0.161
mnth_Aug                   0.0144      0.034      0.428      0.669      -
0.052       0.081
mnth_Dec                  -0.0456      0.034     -1.358      0.175      -
0.112       0.020
mnth_Feb                  -0.0323      0.033     -0.982      0.327      -
0.097       0.032
mnth_Jan                  -0.0628      0.034     -1.873      0.062      -
0.129       0.003
mnth_Jul                  -0.0404      0.035     -1.151      0.250      -
0.109       0.029
mnth_Jun                  -0.0030      0.025     -0.119      0.906      -
0.052       0.046
mnth_Mar                   0.0010      0.025      0.043      0.966      -
0.047       0.049
mnth_May                   0.0239      0.021      1.140      0.255      -
0.017       0.065
```

```
mnth_Nov                  -0.0419      0.036     -1.152      0.250      -
0.113       0.030
mnth_Oct                   0.0075      0.036      0.211      0.833      -
0.063       0.078
mnth_Sep                   0.0811      0.032      2.533      0.012
0.018       0.144
weekday_mondata            0.0985      0.013      7.300      0.000
0.072       0.125
weekday_saturdata         -0.0038      0.015     -0.263      0.793      -
0.033       0.025
weekday_sundata            0.0054      0.015      0.362      0.718      -
0.024       0.035
weekday_thursdata         -0.0135      0.015     -0.917      0.359      -
0.042       0.015
weekday_tuesdata           0.0440      0.014      3.213      0.001
0.017       0.071
weekday_wednesdata        -0.0155      0.015     -1.064      0.288      -
0.044       0.013
weathersit_Light snow     -0.2574      0.026     -9.778      0.000      -
0.309      -0.206
weathersit_Mist+cloudy    -0.0611      0.010     -5.854      0.000      -
0.082      -0.041
==========================================================================
====
Omnibus:                       84.475   Durbin-Watson:
2.040
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              23
5.382
Skew:                          -0.804   Prob(JB):                    7.72
e-52
Kurtosis:                       5.914   Cond. No.                    1.05
e+15
==========================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 1.43e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                  Features    vif
2              workingday   86.81
20         weekday_mondata   19.72
24        weekday_tuesdata   18.35
6            season_spring   10.79
8            season_winter    9.50
7            season_summer    8.29
3                     temp    7.12
17                mnth_Nov    6.80
18                mnth_Oct    6.59
9                 mnth_Aug    6.43
12                mnth_Jan    5.90
10                mnth_Dec    5.68
13                mnth_Jul    5.66
19                mnth_Sep    4.94
1                  holiday    4.59
11                mnth_Feb    4.39
15                mnth_Mar    3.47
14                mnth_Jun    2.83
16                mnth_May    2.22
```

```
4                      hum   2.05
25       weekday_wednesdata   1.78
21        weekday_saturdata   1.62
22          weekday_sundata   1.61
23        weekday_thursdata   1.61
27   weathersit_Mist+cloudy   1.60
26    weathersit_Light snow   1.29
5                 windspeed   1.24
0                        yr   1.06
```

**Here the no of variables to be eliminated are too high. So we use automated Feature selection method i.e: RFE**

For that we need to build a model using Sklearn

In [74]:

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression

sk_model = LinearRegression()
sk_model = sk_model.fit( X_train, y_train)

rfe = RFE(sk_model , 15)
rfe = rfe.fit( X_train, y_train)

list(zip(X_train.columns,rfe.support_,rfe.ranking_))
```

Out[74]:

```
[('yr', True, 1),
 ('holiday', True, 1),
 ('workingday', True, 1),
 ('temp', True, 1),
 ('hum', True, 1),
 ('windspeed', True, 1),
 ('season_spring', True, 1),
 ('season_summer', True, 1),
 ('season_winter', True, 1),
 ('mnth_Aug', False, 9),
 ('mnth_Dec', False, 4),
 ('mnth_Feb', False, 5),
 ('mnth_Jan', False, 2),
 ('mnth_Jul', True, 1),
 ('mnth_Jun', False, 13),
 ('mnth_Mar', False, 14),
 ('mnth_May', False, 6),
 ('mnth_Nov', False, 3),
 ('mnth_Oct', False, 11),
 ('mnth_Sep', True, 1),
 ('weekday_mondata', True, 1),
 ('weekday_saturdata', False, 12),
 ('weekday_sundata', False, 10),
 ('weekday_thursdata', False, 8),
 ('weekday_tuesdata', True, 1),
 ('weekday_wednesdata', False, 7),
 ('weathersit_Light snow', True, 1),
 ('weathersit_Mist+cloudy', True, 1)]
```

In [75]:

```
# The selected features from the RFE process are

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

## Re-Build the model using the new training set using statsmodel

In [76]:

```
# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())



# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.847
Model:                            OLS   Adj. R-squared:
0.843
Method:                 Least Squares   F-statistic:                     1
96.3
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          1.13e
-191
Time:                        18:38:25   Log-Likelihood:                 51
7.87
No. Observations:                 510   AIC:                            -1
006.
Df Residuals:                     495   BIC:                            -9
42.2
Df Model:                          14
Covariance Type:            nonrobust
==========================================================================
================
                             coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                      0.1989      0.028      7.181      0.000
0.144       0.253
yr                         0.2297      0.008     28.660      0.000
0.214       0.245
holiday                   -0.0190      0.021     -0.892      0.373      -
0.061       0.023
workingday                 0.0837      0.010      8.672      0.000
0.065       0.103
temp                       0.5278      0.033     15.897      0.000
0.463       0.593
hum                       -0.1595      0.037     -4.268      0.000      -
0.233      -0.086
windspeed                 -0.1806      0.025     -7.110      0.000      -
0.231      -0.131
season_spring             -0.0554      0.021     -2.694      0.007      -
0.096      -0.015
season_summer              0.0526      0.015      3.553      0.000
0.024       0.082
season_winter              0.1003      0.017      5.890      0.000
0.067       0.134
mnth_Jul                  -0.0549      0.018     -3.035      0.003      -
0.090      -0.019
mnth_Sep                   0.0818      0.016      4.956      0.000
0.049       0.114
weekday_mondata            0.0937      0.012      8.045      0.000
0.071       0.117
weekday_tuesdata           0.0405      0.012      3.304      0.001
0.016       0.065
weathersit_Light snow     -0.2463      0.026     -9.449      0.000      -
0.298      -0.195
weathersit_Mist+cloudy    -0.0578      0.010     -5.559      0.000      -
0.078      -0.037
==========================================================================
====
Omnibus:                       64.879   Durbin-Watson:
2.065
```

```
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             15
8.454
Skew:                          -0.661   Prob(JB):                    3.91
e-35
Kurtosis:                       5.390   Cond. No.                    1.00
e+15
==========================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 1.5e-27. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
                  Features    vif
2              workingday  50.69
11        weekday_mondata  12.20
12       weekday_tuesdata  11.79
6           season_spring   5.02
3                    temp   3.62
8           season_winter   3.49
1                 holiday   2.91
7           season_summer   2.61
4                     hum   1.91
14  weathersit_Mist+cloudy   1.57
9                mnth_Jul   1.49
10               mnth_Sep   1.30
13   weathersit_Light snow   1.25
5               windspeed   1.20
0                      yr   1.03
```

### The holiday variable has high p value and low VIF value. So, lets eliminate it andsee the changes.

In [77]:

```
# Dropping the extra feature
X_train.drop('holiday', inplace = True, axis=1)


# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())


# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.847
Model:                            OLS   Adj. R-squared:
0.843
Method:                 Least Squares   F-statistic:                     1
96.3
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          1.13e
-191
Time:                        18:38:25   Log-Likelihood:                 51
7.87
No. Observations:                 510   AIC:                            -1
006.
Df Residuals:                     495   BIC:                            -9
42.2
Df Model:                          14
Covariance Type:            nonrobust
==========================================================================
================
                            coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                     0.1799      0.042      4.276      0.000
0.097       0.263
yr                        0.2297      0.008     28.660      0.000
0.214       0.245
workingday                0.1027      0.025      4.047      0.000
0.053       0.152
temp                      0.5278      0.033     15.897      0.000
0.463       0.593
hum                      -0.1595      0.037     -4.268      0.000       -
0.233      -0.086
windspeed                -0.1806      0.025     -7.110      0.000       -
0.231      -0.131
season_spring            -0.0554      0.021     -2.694      0.007       -
0.096      -0.015
season_summer             0.0526      0.015      3.553      0.000
0.024       0.082
season_winter             0.1003      0.017      5.890      0.000
0.067       0.134
mnth_Jul                 -0.0549      0.018     -3.035      0.003       -
0.090      -0.019
mnth_Sep                  0.0818      0.016      4.956      0.000
0.049       0.114
weekday_mondata           0.1126      0.027      4.202      0.000
0.060       0.165
weekday_tuesdata          0.0594      0.027      2.206      0.028
0.006       0.112
weathersit_Light snow    -0.2463      0.026     -9.449      0.000       -
0.298      -0.195
weathersit_Mist+cloudy   -0.0578      0.010     -5.559      0.000       -
0.078      -0.037
==========================================================================
====
Omnibus:                       64.879   Durbin-Watson:
2.065
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              15
8.454
```

```
Skew:                          -0.661   Prob(JB):                         3.91
e-35
Kurtosis:                       5.390   Cond. No.
25.2
==========================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
                  Features    vif
3                      hum  32.14
2                     temp  18.98
1               workingday  18.41
10         weekday_mondata   4.91
4                windspeed   4.90
5            season_spring   4.80
11        weekday_tuesdata   4.76
7            season_winter   3.71
6            season_summer   3.03
13  weathersit_Mist+cloudy   2.31
0                       yr   2.09
8                 mnth_Jul   1.60
9                 mnth_Sep   1.38
12   weathersit_Light snow   1.25
```

**Now all the P-values are set but the VIF value of the variable "hum" > 10 which is a bad sign. So, drop it.**

In [78]:

```
# Dropping the extra feature
X_train.drop('hum', inplace = True, axis=1)


# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())


# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.842
Model:                            OLS   Adj. R-squared:
0.838
Method:                 Least Squares   F-statistic:                     2
03.0
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          5.73e
-189
Time:                        18:38:25   Log-Likelihood:                 50
8.65
No. Observations:                 510   AIC:                            -9
89.3
Df Residuals:                     496   BIC:                            -9
30.0
Df Model:                          13
Covariance Type:            nonrobust
==========================================================================
================
                             coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                      0.1005      0.038      2.618      0.009
0.025       0.176
yr                         0.2336      0.008     28.839      0.000
0.218       0.250
workingday                 0.1034      0.026      4.008      0.000
0.053       0.154
temp                       0.4920      0.033     15.056      0.000
0.428       0.556
windspeed                 -0.1491      0.025     -6.032      0.000      -
0.198      -0.101
season_spring             -0.0653      0.021     -3.139      0.002      -
0.106      -0.024
season_summer              0.0465      0.015      3.101      0.002
0.017       0.076
season_winter              0.0859      0.017      5.058      0.000
0.053       0.119
mnth_Jul                  -0.0500      0.018     -2.723      0.007      -
0.086      -0.014
mnth_Sep                   0.0758      0.017      4.532      0.000
0.043       0.109
weekday_mondata            0.1152      0.027      4.225      0.000
0.062       0.169
weekday_tuesdata           0.0571      0.027      2.085      0.038
0.003       0.111
weathersit_Light snow     -0.2904      0.024    -11.931      0.000      -
0.338      -0.243
weathersit_Mist+cloudy    -0.0835      0.009     -9.669      0.000      -
0.100      -0.067
==========================================================================
====
Omnibus:                       66.977   Durbin-Watson:
2.059
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               16
3.728
Skew:                          -0.681   Prob(JB):                     2.80
e-36
```

```
Kurtosis:                       5.419   Cond. No.
22.6
==========================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
                  Features    vif
1               workingday  16.19
2                     temp  12.73
3                windspeed   4.75
9          weekday_mondata   4.45
10        weekday_tuesdata   4.21
4            season_spring   3.82
6            season_winter   2.80
5            season_summer   2.75
0                       yr   2.07
7                 mnth_Jul   1.60
12  weathersit_Mist+cloudy   1.58
8                 mnth_Sep   1.35
11   weathersit_Light snow   1.09
```

### Still there are 2 features 'workingday' and 'temp' that has VIF>10. So, drop them one by one checking the changes in VIF.

In [79]:

```
# Dropping the extra feature
X_train.drop('workingday', inplace = True, axis=1)


# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())


# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.837
Model:                            OLS   Adj. R-squared:
0.833
Method:                 Least Squares   F-statistic:                     2
12.1
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          1.01e
-186
Time:                        18:38:25   Log-Likelihood:                 50
0.52
No. Observations:                 510   AIC:                            -9
75.0
Df Residuals:                     497   BIC:                            -9
20.0
Df Model:                          12
Covariance Type:            nonrobust
==========================================================================
================
                             coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                      0.2005      0.030      6.771      0.000
0.142       0.259
yr                         0.2341      0.008     28.476      0.000
0.218       0.250
temp                       0.4934      0.033     14.874      0.000
0.428       0.559
windspeed                 -0.1513      0.025     -6.031      0.000      -
0.201      -0.102
season_spring             -0.0679      0.021     -3.217      0.001      -
0.109      -0.026
season_summer              0.0469      0.015      3.081      0.002
0.017       0.077
season_winter              0.0829      0.017      4.818      0.000
0.049       0.117
mnth_Jul                  -0.0492      0.019     -2.639      0.009      -
0.086      -0.013
mnth_Sep                   0.0721      0.017      4.257      0.000
0.039       0.105
weekday_mondata            0.0157      0.011      1.371      0.171      -
0.007       0.038
weekday_tuesdata          -0.0422      0.012     -3.562      0.000      -
0.066      -0.019
weathersit_Light snow     -0.2858      0.025    -11.578      0.000      -
0.334      -0.237
weathersit_Mist+cloudy    -0.0816      0.009     -9.323      0.000      -
0.099      -0.064
==========================================================================
====
Omnibus:                       80.763   Durbin-Watson:
2.010
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               21
0.920
Skew:                          -0.791   Prob(JB):                     1.58
e-46
Kurtosis:                       5.725   Cond. No.
17.5
```

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

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
                  Features   vif
1                     temp  5.17
2                windspeed  4.62
4            season_summer  2.23
3            season_spring  2.11
0                       yr  2.07
5            season_winter  1.82
6                 mnth_Jul  1.59
11  weathersit_Mist+cloudy  1.55
7                 mnth_Sep  1.33
8          weekday_mondata  1.22
9         weekday_tuesdata  1.21
10   weathersit_Light snow  1.08
```

## Now weekday_mondata has high p value. drop it

In [80]:

```
# Dropping the extra feature
X_train.drop('weekday_mondata', inplace = True, axis=1)


# Building a model from statsmodel
X_train_lm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_lm).fit()
print(model.summary())


# Calcilating VIF values
VIF = pd.DataFrame()
VIF['Features'] = X_train.columns
VIF['vif'] = [variance_inflation_factor(X_train.values, i) for i in range(X_train.shape
[1])]
VIF['vif'] = round(VIF['vif'],2)
VIF = VIF.sort_values(by = "vif", ascending = False)
print(VIF)
```

```
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    cnt   R-squared:
0.836
Model:                            OLS   Adj. R-squared:
0.832
Method:                 Least Squares   F-statistic:                     2
30.8
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          1.65e
-187
Time:                        18:38:25   Log-Likelihood:                 49
9.56
No. Observations:                 510   AIC:                            -9
75.1
Df Residuals:                     498   BIC:                            -9
24.3
Df Model:                          11
Covariance Type:            nonrobust
==========================================================================
================
                             coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                      0.2036      0.030      6.889      0.000
0.146       0.262
yr                         0.2338      0.008     28.423      0.000
0.218       0.250
temp                       0.4923      0.033     14.832      0.000
0.427       0.557
windspeed                 -0.1498      0.025     -5.970      0.000      -
0.199      -0.100
season_spring             -0.0680      0.021     -3.219      0.001      -
0.109      -0.026
season_summer              0.0467      0.015      3.067      0.002
0.017       0.077
season_winter              0.0831      0.017      4.824      0.000
0.049       0.117
mnth_Jul                  -0.0486      0.019     -2.607      0.009      -
0.085      -0.012
mnth_Sep                   0.0721      0.017      4.253      0.000
0.039       0.105
weekday_tuesdata          -0.0451      0.012     -3.862      0.000      -
0.068      -0.022
weathersit_Light snow     -0.2856      0.025    -11.560      0.000      -
0.334      -0.237
weathersit_Mist+cloudy    -0.0816      0.009     -9.311      0.000      -
0.099      -0.064
==========================================================================
====
Omnibus:                       76.151   Durbin-Watson:
2.009
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              20
7.716
Skew:                          -0.733   Prob(JB):                     7.85
e-46
Kurtosis:                       5.762   Cond. No.
17.4
==========================================================================
====
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
                Features   vif
1                   temp  5.13
2              windspeed  4.60
4          season_summer  2.22
3          season_spring  2.09
0                     yr  2.07
5          season_winter  1.80
6               mnth_Jul  1.59
10  weathersit_Mist+cloudy  1.55
7               mnth_Sep  1.33
8        weekday_tuesdata  1.17
9    weathersit_Light snow  1.08
```

### Now lets calculate the $R^2$ value for the actual and predicted values for the training set.

In [81]:

```
# Predicted value of the training data.
y_train_pred = model.predict(X_train_lm)
```

In [82]:

```
# Calculating the R2 value for the
from sklearn.metrics import r2_score

r2_score(y_true=y_train , y_pred=y_train_pred)
```

Out[82]:

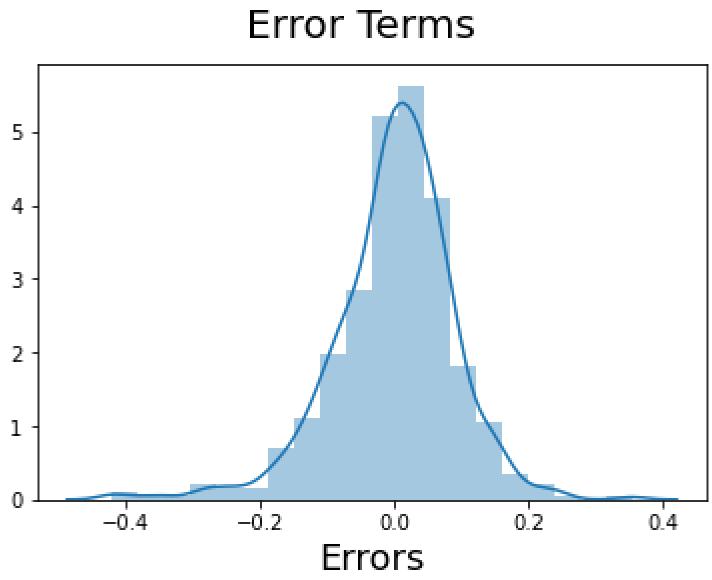
```
0.8360233701515918
```

In [83]:

```
# Plot the histogram of the error terms
fig = plt.figure()
sns.distplot((y_train - y_train_pred), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)                  # Plot heading
plt.xlabel('Errors', fontsize = 18)
```

Out[83]:

```
Text(0.5, 0, 'Errors')
```

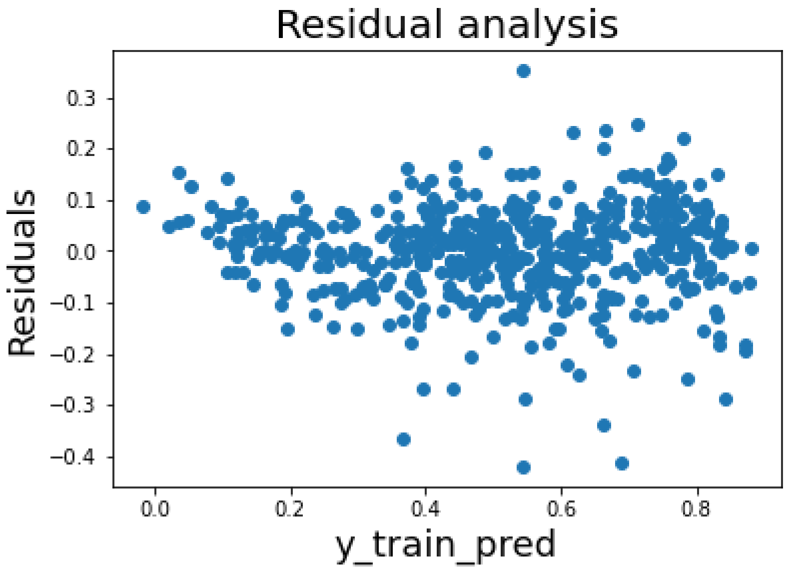


In [99]:

```
# Error terms are independent of each other
residuals = y_train - y_train_pred
plt.scatter( y_train_pred,residuals)
plt.title('Residual analysis', fontsize = 20)                    # Plot heading
plt.xlabel('y_train_pred', fontsize = 18)
plt.ylabel('Residuals', fontsize = 18)
```

Out[99]:

```
Text(0, 0.5, 'Residuals')
```




In [ ]:

```

```

# Testing and Evaluation

In [84]:

```
data_test[data_test.columns] = scaler.transform(data_test[data_test.columns])
```

In [85]:

```
data_test['yr']
```

Out[85]:

```
184    0.0
535    1.0
299    0.0
221    0.0
152    0.0
      ...
400    1.0
702    1.0
127    0.0
640    1.0
72     0.0
Name: yr, Length: 219, dtype: float64
```

In [86]:

```
# Splitting X,y

y_test = data_test.pop('cnt')
X_test = data_test
```

In [87]:

```
X_test_lm = sm.add_constant(X_test)

X_test_lm = X_test_lm[X_train_lm.columns]
X_test_lm
```

Out[87]:

| | const | yr | temp | windspeed | season_spring | season_summer | season_winter | mnth_ |
|---|---|---|---|---|---|---|---|---|
| **184** | 1.0 | 0.0 | 0.831783 | 0.084219 | 0.0 | 0.0 | 0.0 | |
| **535** | 1.0 | 1.0 | 0.901354 | 0.153728 | 0.0 | 1.0 | 0.0 | |
| **299** | 1.0 | 0.0 | 0.511964 | 0.334206 | 0.0 | 0.0 | 1.0 | |
| **221** | 1.0 | 0.0 | 0.881625 | 0.339570 | 0.0 | 0.0 | 0.0 | |
| **152** | 1.0 | 0.0 | 0.817246 | 0.537414 | 0.0 | 1.0 | 0.0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **400** | 1.0 | 1.0 | 0.257562 | 0.287411 | 1.0 | 0.0 | 0.0 | |
| **702** | 1.0 | 1.0 | 0.519232 | 0.283397 | 0.0 | 0.0 | 1.0 | |
| **127** | 1.0 | 0.0 | 0.584649 | 0.069510 | 0.0 | 1.0 | 0.0 | |
| **640** | 1.0 | 1.0 | 0.745598 | 0.052115 | 0.0 | 0.0 | 1.0 | |
| **72** | 1.0 | 0.0 | 0.331557 | 0.203418 | 1.0 | 0.0 | 0.0 | |

219 rows × 12 columns

In [88]:

```
y_test_pred = model.predict(X_test_lm)

r2_score(y_true = y_test, y_pred = y_test_pred)
```

Out[88]:

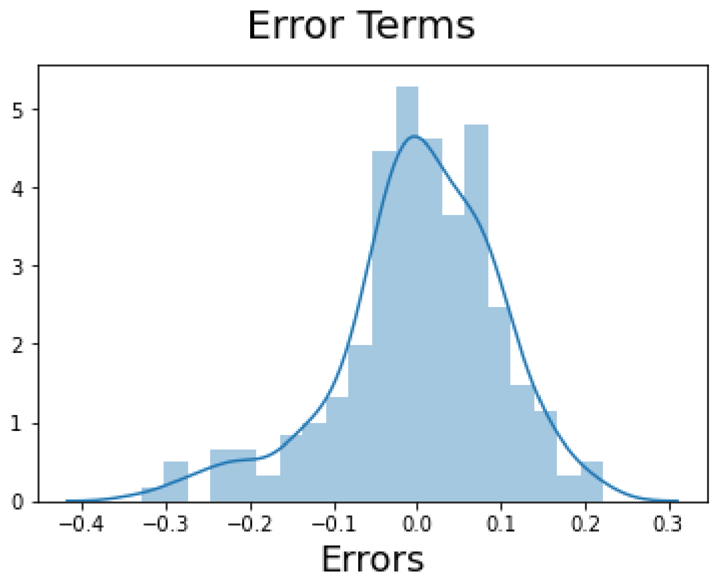
0.805407680173852

In [101]:

```
# Plot the histogram of the error terms
fig = plt.figure()
sns.distplot((y_test - y_test_pred), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)                  # Plot heading
plt.xlabel('Errors', fontsize = 18)
```

Out[101]:

Text(0.5, 0, 'Errors')

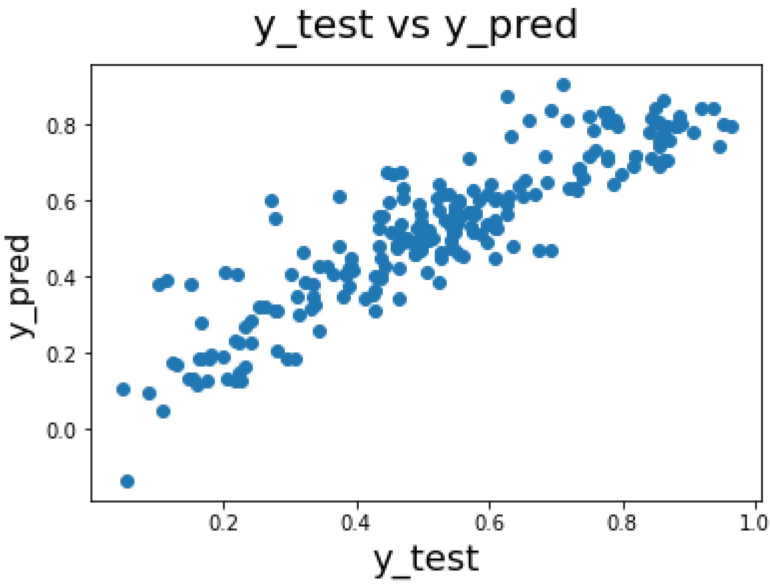


In [90]:

```
fig = plt.figure()
plt.scatter(y_test, y_test_pred)
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)
```

Out[90]:

```
Text(0, 0.5, 'y_pred')
```




In [ ]:

```
```

In [ ]:

```
```