Submitted by : Aadarsha Chapagain

Student id: C0825975

Week 13 Assignment 2 W22

Data Visualization in python

Importing libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import missingno as msno
import seaborn as sns
from scipy import stats
import missingno as msno
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error, accuracy_score
from scipy.stats import shapiro
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from statsmodels.stats.diagnostic import normal_ad, het_breuschpagan
import statsmodels.api as sm
from sklearn.preprocessing import minmax_scale

%matplotlib inline

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

Datasets

Datasets

In [4]:
```
df = pd.read_csv('winequality-red.csv')
```

In [5]:
```
df.head()
```

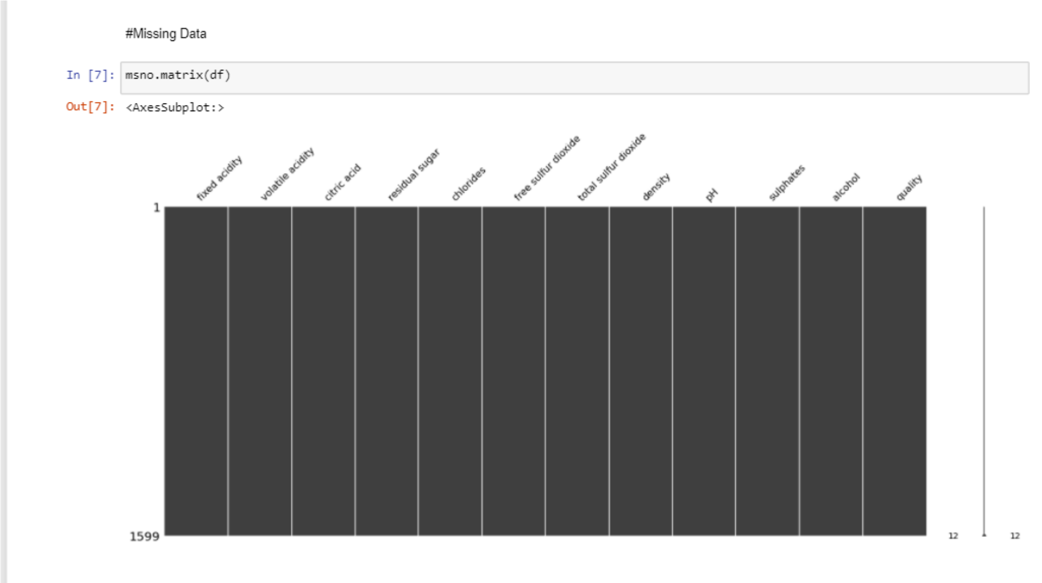
Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

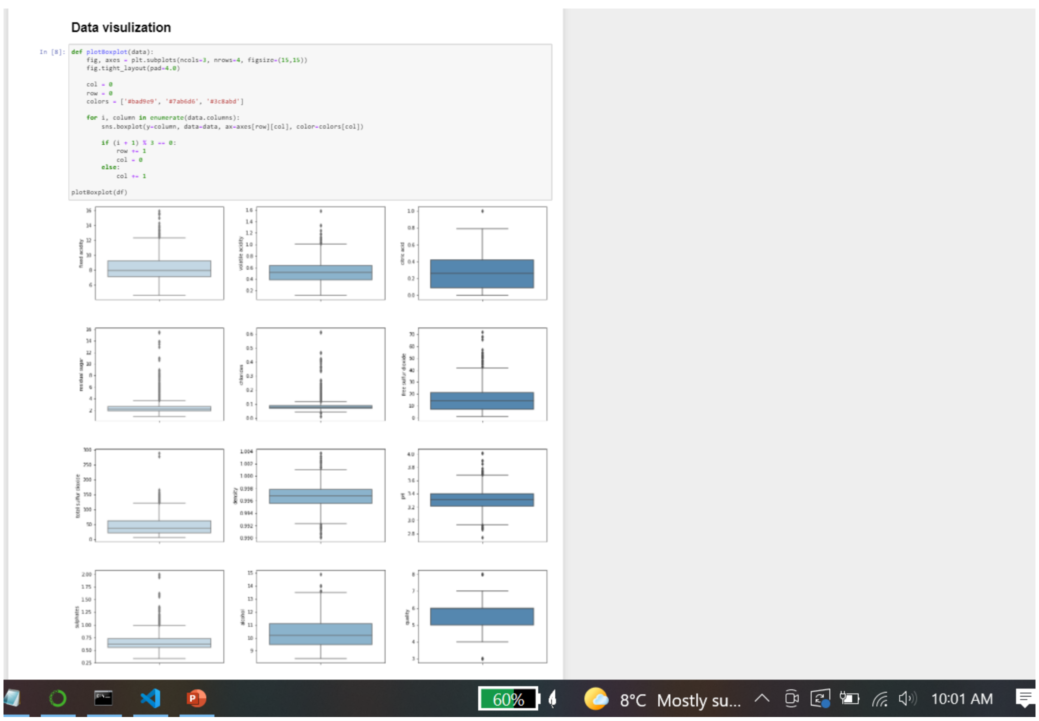
In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Missing Data



Data Visualization

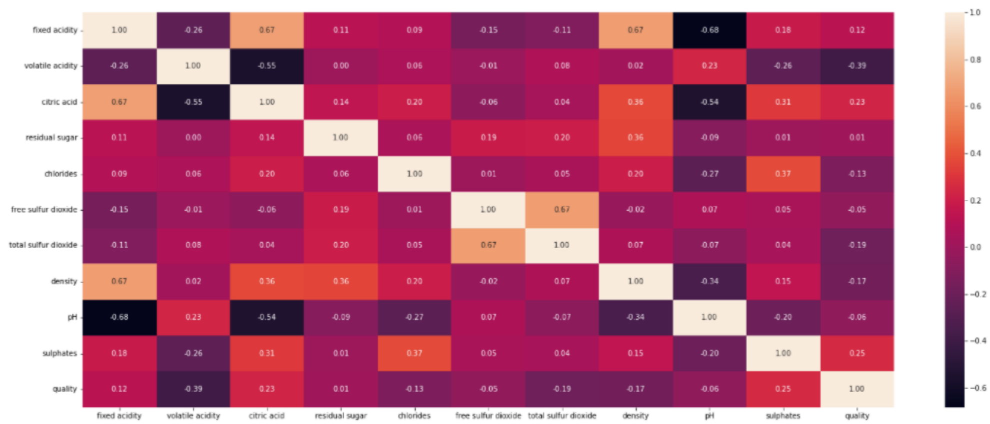


Heat Maps

No Multicollinearity among Independant Varaibles

[10]:
```python
plt.figure(figsize=(25, 10))
sns.heatmap(df.loc[:, df.columns != 'alcohol'].corr(), annot=True, fmt='.2f')
```

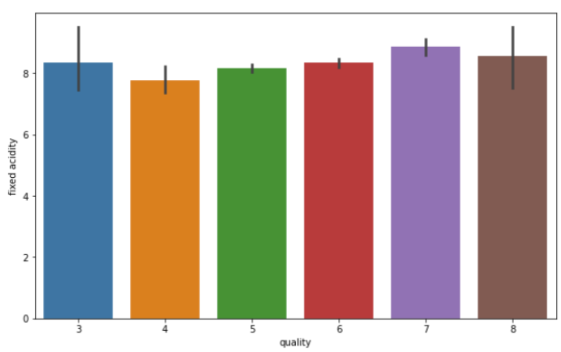
[10]: <AxesSubplot:>



Barchart

In [11]:
```python
#Here we see that fixed acidity does not give any specification to classify the quality.
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'fixed acidity', data = df)
```

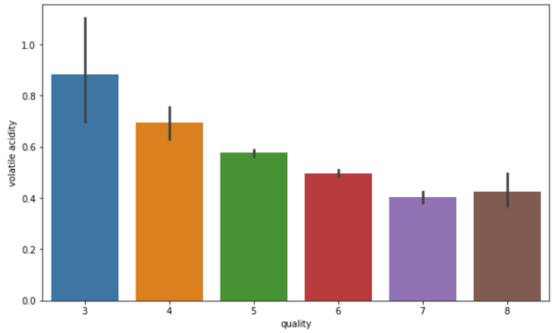
Out[11]: <AxesSubplot:xlabel='quality', ylabel='fixed acidity'>



In [12]:

```
#Here we see that its quite a downing trend in the volatile acidity as we go higher the quality
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'volatile acidity', data = df)
```

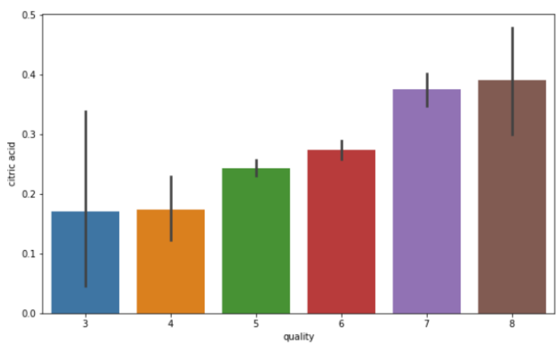
Out[12]: <AxesSubplot:xlabel='quality', ylabel='volatile acidity'>



In [13]:

```
#Composition of citric acid go higher as we go higher in the quality of the wine
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'citric acid', data = df)
```

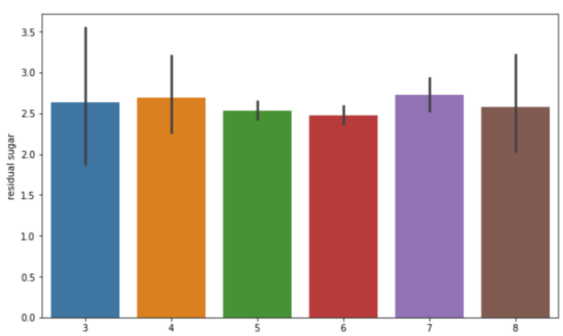
Out[13]: <AxesSubplot:xlabel='quality', ylabel='citric acid'>



In [14]:

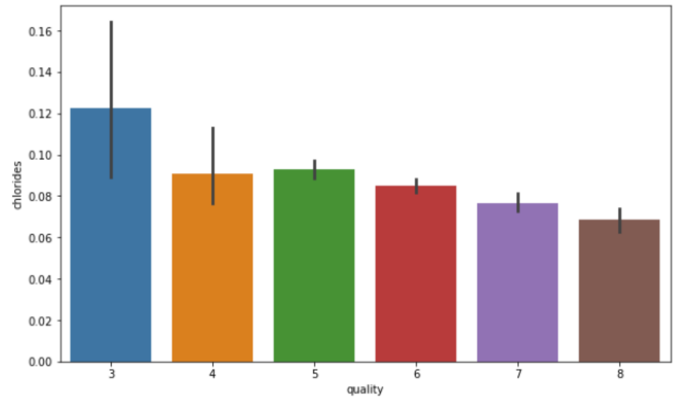
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'residual sugar', data = df)
```

Out[14]: <AxesSubplot:xlabel='quality', ylabel='residual sugar'>



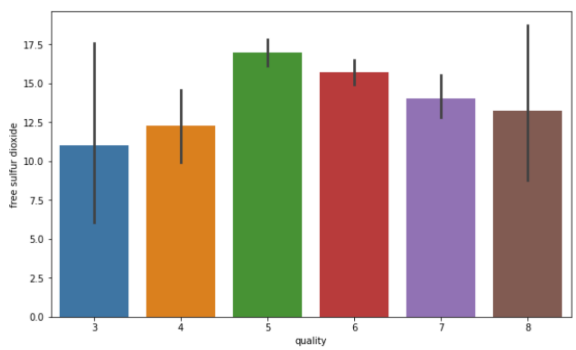
In [15]:
```
#Composition of chloride also go down as we go higher in the quality of the wine
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'chlorides', data = df)
```

Out[15]: <AxesSubplot:xlabel='quality', ylabel='chlorides'>



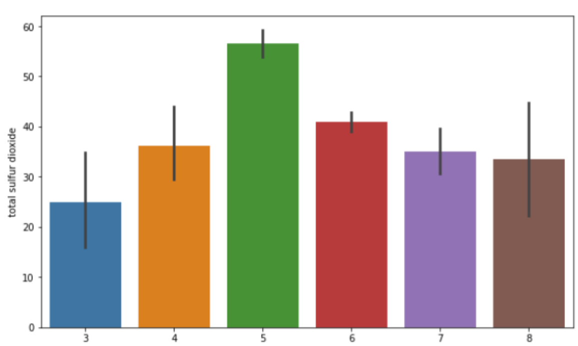
In [16]:
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'free sulfur dioxide', data = df)
```

Out[16]: <AxesSubplot:xlabel='quality', ylabel='free sulfur dioxide'>



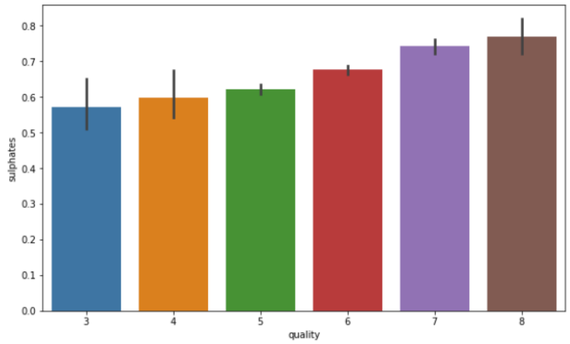
In [17]:
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'total sulfur dioxide', data = df)
```

Out[17]: <AxesSubplot:xlabel='quality', ylabel='total sulfur dioxide'>



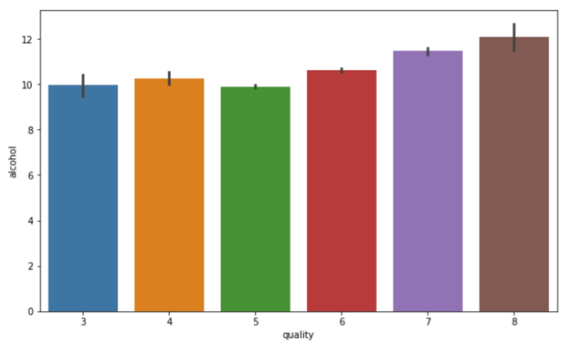
In [18]:
```
#Sulphates level goes higher with the quality of wine
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'sulphates', data = df)
```

Out[18]: <AxesSubplot:xlabel='quality', ylabel='sulphates'>



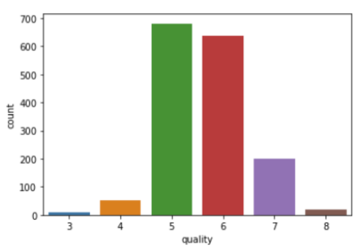
In [19]:
```
#Alcohol level also goes higher as te quality of wine increases
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'alcohol', data = df)
```

Out[19]: <AxesSubplot:xlabel='quality', ylabel='alcohol'>



In [20]:
```
sns.countplot(df['quality'])
```

Out[20]: <AxesSubplot:xlabel='quality', ylabel='count'>



Features and label

Features and Labels

In [21]:
```
#create tmp train/test split for assumptions test
X = df.drop(['alcohol'], axis=1)
y = df['alcohol']
```

In [19]:
```
print(X)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0               7.4             0.700         0.00             1.9      0.076
1               7.8             0.880         0.00             2.6      0.098
2               7.8             0.760         0.04             2.3      0.092
3              11.2             0.280         0.56             1.9      0.075
4               7.4             0.700         0.00             1.9      0.076
...             ...               ...          ...             ...        ...
1594            6.2             0.600         0.08             2.0      0.090
1595            5.9             0.550         0.10             2.2      0.062
1596            6.3             0.510         0.13             2.3      0.076
1597            5.9             0.645         0.12             2.0      0.075
1598            6.0             0.310         0.47             3.6      0.067

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                    11.0                  34.0  0.99780  3.51       0.56
1                    25.0                  67.0  0.99680  3.20       0.68
2                    15.0                  54.0  0.99700  3.26       0.65
3                    17.0                  60.0  0.99800  3.16       0.58
4                    11.0                  34.0  0.99780  3.51       0.56
...                   ...                   ...      ...   ...        ...
1594                 32.0                  44.0  0.99490  3.45       0.58
1595                 39.0                  51.0  0.99512  3.52       0.76
1596                 29.0                  40.0  0.99574  3.42       0.75
1597                 32.0                  44.0  0.99547  3.57       0.71
1598                 18.0                  42.0  0.99549  3.39       0.66

      quality
0           5
1           5
2           5
3           6
4           5
...       ...
1594        5
1595        6
1596        6
1597        5
1598        6

[1599 rows x 11 columns]
```

In [20]:
```
print(y)
```

```
0        9.4
1        9.8
2        9.8
3        9.8
4        9.4
        ...
1594    10.5
1595    11.2
1596    11.0
1597    10.2
1598    11.0
Name: alcohol, Length: 1599, dtype: float64
```

#Splitting the dataset

Splitting the dataset

#Splitting the dataset

In [22]:
```
x_train, x_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=50)
```

In [22]:
```
print(x_train)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
940             9.6             0.330         0.52             2.2      0.074
1287            8.0             0.600         0.08             2.6      0.056
1397            7.3             0.590         0.26             2.0      0.080
356            11.5             0.410         0.52             3.0      0.080
226             8.9             0.590         0.50             2.0      0.337
...             ...               ...          ...             ...        ...
70              7.7             0.630         0.08             1.9      0.076
132             5.6             0.500         0.09             2.3      0.049
1313            7.0             0.360         0.21             2.3      0.086
109             8.1             0.785         0.52             2.0      0.122
1504            7.5             0.380         0.57             2.3      0.106

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
940                  13.0                  25.0  0.99509  3.36       0.76
1287                  3.0                   7.0  0.99286  3.22       0.37
1397                 17.0                 104.0  0.99584  3.28       0.52
356                  29.0                  55.0  1.00010  3.26       0.88
226                  27.0                  81.0  0.99640  3.04       1.61
...                   ...                   ...      ...   ...        ...
70                   15.0                  27.0  0.99670  3.32       0.54
132                  17.0                  99.0  0.99370  3.63       0.63
1313                 20.0                  65.0  0.99558  3.40       0.54
109                  37.0                 153.0  0.99690  3.21       0.69
1504                  5.0                  12.0  0.99605  3.36       0.55

      quality
940         7
1287        5
1397        5
356         5
226         6
...       ...
70          6
132         5
1313        6
109         5
1504        6

[1119 rows x 11 columns]
```

In [23]:
```python
print(y_train)
```

```
940     12.4
1287    13.0
1397     9.9
356     11.0
226      9.5
        ... 
70       9.5
132     13.0
1313    10.1
109      9.3
1504    11.4
Name: alcohol, Length: 1119, dtype: float64
```

In [24]:
```python
print(x_test)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
453            10.4              0.33         0.63            2.80      0.084
1415            6.2              0.58         0.00            1.60      0.065
1242            9.0              0.40         0.41            2.00      0.058
885             8.9              0.75         0.14            2.50      0.086
488            11.6              0.32         0.55            2.80      0.081
...             ...               ...          ...             ...        ...
34              5.2              0.32         0.25            1.80      0.103
1493            7.7              0.54         0.26            1.90      0.089
501            10.4              0.44         0.73            6.55      0.074
1464            6.8              0.59         0.10            1.70      0.063
911             9.1              0.28         0.46            9.00      0.114

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
453                   5.0                  22.0  0.99980  3.26       0.74
1415                  8.0                  18.0  0.99660  3.56       0.84
1242                 15.0                  40.0  0.99414  3.22       0.60
885                   9.0                  30.0  0.99824  3.34       0.64
488                  35.0                  67.0  1.00020  3.32       0.92
...                   ...                   ...      ...   ...        ...
34                   13.0                  50.0  0.99570  3.38       0.55
1493                 23.0                 147.0  0.99636  3.26       0.59
501                  38.0                  76.0  0.99900  3.17       0.85
1464                 34.0                  53.0  0.99580  3.41       0.67
911                   3.0                   9.0  0.99901  3.18       0.60

      quality
453         7
1415        5
1242        6
885         5
488         7
...       ...
34          5
1493        5
501         7
1464        5
911         6

[480 rows x 11 columns]
```

Scaling Using Standard Scaler

```
(1119,)
```

In [26]:
```python
sc = StandardScaler()
X_train = sc.fit_transform(x_train)
X_test = sc.transform(x_test)
```

Model Setup

Model Setup

In [27]:
```
from sklearn.linear_model import LinearRegression
```

In [28]:
```
model = LinearRegression()
```

Training

Training

Training

In [30]:
```
model.fit(x_train, y_train)
```

Out[30]:
```
LinearRegression()
```

In [31]:
```
model.coef_
```

Out[31]:
```
array([ 5.05933116e-01,  7.02590521e-01,  8.23700660e-01,  2.72656929e-01,
       -5.33212379e-01, -5.68110422e-03, -5.96585162e-04, -5.87493452e+02,
        3.78523263e+00,  8.59410226e-01,  2.47684402e-01])
```

In [32]:
```
model.intercept_
```

Out[32]:
```
576.1956939510893
```

#Evaluating Model

In [33]:
```
print(model.score(X, y))
```

```
0.6895215542808213
```

Predicting on test data and visualization

Predicting on Test data

```
In [34]: y_pred = model.predict(x_test)
```

```
In [35]: r2_score(y_test, y_pred)
```

```
Out[35]: 0.6648714487925931
```

Visualisation

```
In [36]: #plot the actual vs predicted values
         sns.regplot(y_test, y_pred, line_kws={'color':'red'}, ci=None)

         plt.xlabel('Actual')
         plt.ylabel('Predictions')
         plt.title('Prediction vs Actual')
```