wine-quality-prediction

February 3, 2024

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

```
[2]: df = pd.read_csv('wine-quality.csv')
     df.head()
```

```
[2]:    fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
     0            7.0              0.27         0.36            20.7      0.045
     1            6.3              0.30         0.34             1.6      0.049
     2            8.1              0.28         0.40             6.9      0.050
     3            7.2              0.23         0.32             8.5      0.058
     4            7.2              0.23         0.32             8.5      0.058

        free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
     0                 45.0                 170.0   1.0010  3.00       0.45
     1                 14.0                 132.0   0.9940  3.30       0.49
     2                 30.0                  97.0   0.9951  3.26       0.44
     3                 47.0                 186.0   0.9956  3.19       0.40
     4                 47.0                 186.0   0.9956  3.19       0.40

        alcohol  quality
     0      8.8        6
     1      9.5        6
     2     10.1        6
     3      9.9        6
     4      9.9        6
```

```
[3]: df.shape
```

```
[3]: (4898, 12)
```

```
[4]: df.columns
```

```
[4]: Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
            'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
            'pH', 'sulphates', 'alcohol', 'quality'],
           dtype='object')
```

[4]:

```
df.isnull().sum()
```

[4]:

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

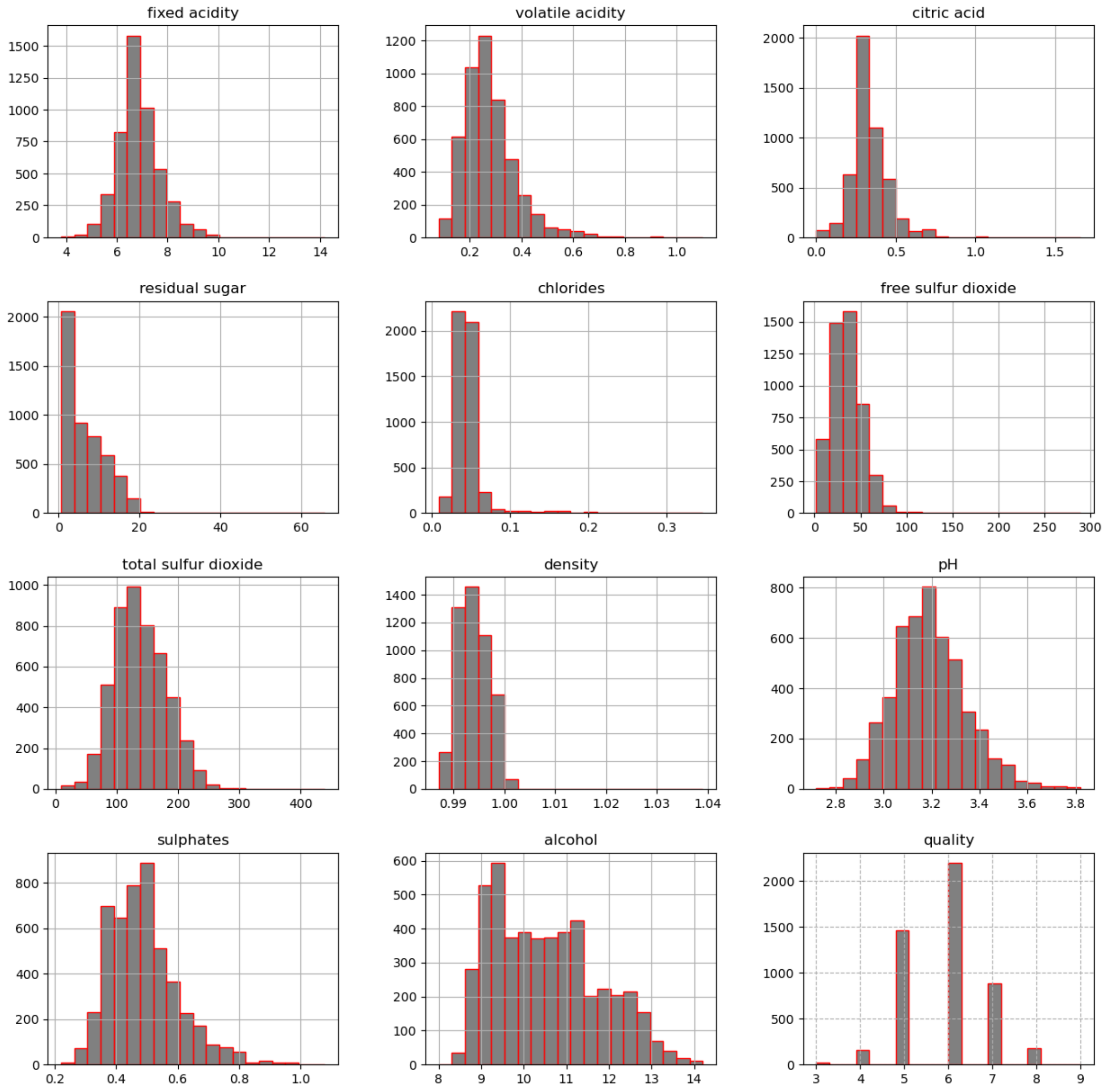
[5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         4898 non-null   float64
 1   volatile acidity      4898 non-null   float64
 2   citric acid           4898 non-null   float64
 3   residual sugar        4898 non-null   float64
 4   chlorides             4898 non-null   float64
 5   free sulfur dioxide   4898 non-null   float64
 6   total sulfur dioxide  4898 non-null   float64
 7   density               4898 non-null   float64
 8   pH                    4898 non-null   float64
 9   sulphates             4898 non-null   float64
 10  alcohol               4898 non-null   float64
 11  quality               4898 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 459.3 KB
```

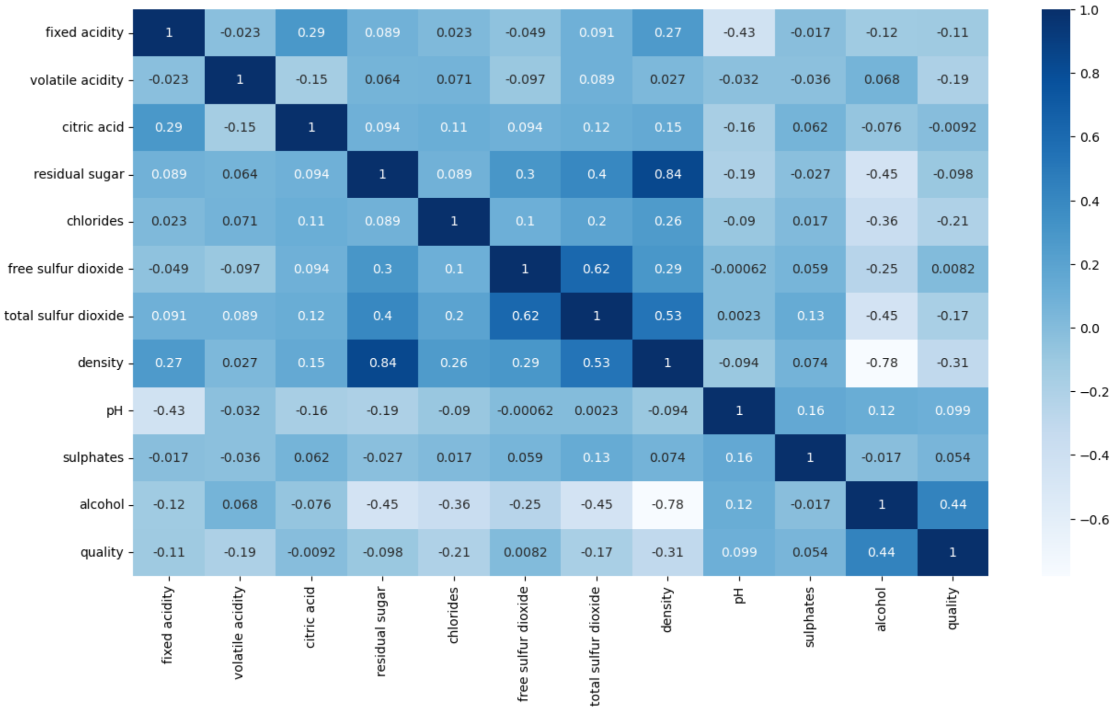
[6]:

```
# plt.figure(figsize=(15,15))
df.hist(bins=20,figsize=(15,15), color='grey', edgecolor='red')
plt.grid(linestyle='--')
plt.show()
```



[7]:
```
plt.figure(figsize=(15,8))
sns.heatmap(df.corr(), annot=True, cmap='Blues')
```

[7]: <Axes: >



[8]:
```
x = df.drop('quality', axis=1)
y = df['quality']
```

[9]:
```
x.head()
```

[9]:
```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.0              0.27         0.36            20.7      0.045
1            6.3              0.30         0.34             1.6      0.049
2            8.1              0.28         0.40             6.9      0.050
3            7.2              0.23         0.32             8.5      0.058
4            7.2              0.23         0.32             8.5      0.058

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 45.0                 170.0   1.0010  3.00       0.45
1                 14.0                 132.0   0.9940  3.30       0.49
2                 30.0                  97.0   0.9951  3.26       0.44
3                 47.0                 186.0   0.9956  3.19       0.40
4                 47.0                 186.0   0.9956  3.19       0.40

   alcohol
0      8.8
1      9.5
2     10.1
3      9.9
```

```
4      9.9
```

```
[10]: y.head()
```

```
[10]: 0    6
      1    6
      2    6
      3    6
      4    6
      Name: quality, dtype: int64
```

0.1 splitting the data

```
[11]: from sklearn.model_selection import train_test_split
      x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2,␣
       ↪random_state=101)
      print('x_train: ', x_train.shape)
      print('x_test: ', x_test.shape)
      print('y_train: ', y_train.shape)
      print('y_test: ', y_test.shape)
```

```
x_train:  (3918, 11)
x_test:  (980, 11)
y_train:  (3918,)
y_test:  (980,)
```

0.2 preproceswing the data

```
[12]: from sklearn.preprocessing import StandardScaler
      scaler = StandardScaler()
      x_train_scaled = scaler.fit_transform(x_train)
      x_test_scaled = scaler.fit_transform(x_test)
```

```
[13]: x_train_scaled,x_test_scaled
```

```
[13]: (array([[-1.59549189,  1.43374398, -2.01364076, …,  0.94099026,
                0.18077608,  0.70930052],
              [-0.88939471, -1.17914458,  0.05692741, …,  0.8748854 ,
                0.79651933, -0.30440681],
              [-0.18329752, -0.47567458, -0.02589531, …,  0.8748854 ,
               -0.78682047,  0.222721  ],
              …,
              [ 0.05206821, -0.87765744,  0.55386377, …, -0.71163105,
                1.67615255,  0.79039711],
              [-1.00707757, -1.48063172, -0.44000895, …,  2.65971642,
               -0.5229305 ,  0.06052783],
              [-0.18329752,  0.22779541,  1.29926831, …, -0.44721164,
               -0.34700386, -0.50714827]]),
       array([[-1.16034599,  0.17156545, -0.39719192, …, -0.0048365 ,
               -0.96229349,  0.68094974],
              [-0.18415621,  1.68586113, -1.29898824, …, -0.7391665 ,
               -0.4441829 ,  1.74737515],
              [ 0.91405729, -1.24808675,  0.5046044 , …, -0.33862286,
                2.31907355, -0.38547568],
              …,
              [ 1.15810474, -0.01772151, -0.72511786, …, -1.40673924,
               -1.04864525,  2.07550605],
              [-1.28236971,  0.31353067, -0.56115489, …,  1.19679442,
               -0.78958996,  1.17314608],
              [-0.06213249, -0.11236499, -0.97106231, …, -0.80592378,
                0.33298298, -1.04173747]]]))
```

[14]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
```

[15]:
```
models = {
    ('Logistic Regression',LogisticRegression()),
    ('Decision Tree',DecisionTreeClassifier()),
    ('Random Forest',RandomForestClassifier()),
    ('Gradient Boosting',GradientBoostingClassifier()),
    ('SVM',SVC()),
    ('KNeighborsClassifier',KNeighborsClassifier())
}
```

[16]:
```
models
```

[16]:
```
{('Decision Tree', DecisionTreeClassifier()),
 ('Gradient Boosting', GradientBoostingClassifier()),
 ('KNeighborsClassifier', KNeighborsClassifier()),
 ('Logistic Regression', LogisticRegression()),
 ('Random Forest', RandomForestClassifier()),
 ('SVM', SVC())}
```

[17]:
```
result = pd.DataFrame(columns=['Model','Accuracy_score'])
```

[18]:
```
result
```

[18]:
```
Empty DataFrame
Columns: [Model, Accuracy_score]
Index: []
```

[19]:
```
for model_name , model in models:
    try: #exception
        model.fit(x_train_scaled, y_train)
        prediction = model.predict(x_test_scaled)

        accuracy = accuracy_score(y_test, prediction)

        result = result.append({'Model':model_name,
                                'Accuracy_score':accuracy,
                                },ignore_index=True)

        print(f'\nModel: {model_name}')
        print('Classification reports\n',classification_report(y_test,␣
 ↪prediction))
        print('Confusion Matrix\n',confusion_matrix(y_test, prediction))

    except Exception as e:
        print(f'Error occurred while processing {model_name}: {str(e)}')
print(result)
```

```
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\linear_model\_logistic.py:460: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  result = result.append({'Model':model_name,
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
```

```
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  result = result.append({'Model':model_name,


Model: Logistic Regression
Classification reports
               precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.75      0.07      0.13        41
           5       0.59      0.54      0.56       306
           6       0.53      0.74      0.62       433
           7       0.42      0.27      0.32       158
           8       0.00      0.00      0.00        37

    accuracy                           0.54       980
   macro avg       0.38      0.27      0.27       980
weighted avg       0.52      0.54      0.51       980

Confusion Matrix
 [[  0   0   3   1   1   0]
 [  0   3  25  13   0   0]
 [  0   1 164 137   4   0]
 [  0   0  71 319  43   0]
 [  0   0  12 104  42   0]
 [  0   0   2  24  11   0]]

Model: KNeighborsClassifier
Classification reports
               precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.26      0.12      0.17        41
           5       0.53      0.59      0.56       306
           6       0.59      0.61      0.60       433
           7       0.48      0.44      0.46       158
           8       0.24      0.11      0.15        37

    accuracy                           0.54       980
   macro avg       0.35      0.31      0.32       980
weighted avg       0.52      0.54      0.53       980

Confusion Matrix
 [[  0   0   3   1   1   0]
 [  1   5  25   8   2   0]
 [  1  10 182  96  14   3]
 [  0   3 115 266  42   7]
 [  0   0  16  69  70   3]
 [  1   1   4   9  18   4]]
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  result = result.append({'Model':model_name,
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
Model: Random Forest
Classification reports
               precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.75      0.15      0.24        41
           5       0.70      0.69      0.69       306
           6       0.66      0.78      0.71       433
           7       0.65      0.60      0.62       158
           8       0.81      0.35      0.49        37

    accuracy                           0.67       980
   macro avg       0.59      0.43      0.46       980
weighted avg       0.68      0.67      0.66       980

Confusion Matrix
 [[  0   0   3   2   0   0]
 [  0   6  25  10   0   0]
 [  0   2 210  89   5   0]
 [  0   0  59 336  36   2]
 [  0   0   5  57  95   1]
 [  0   0   0  14  10  13]]

C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  result = result.append({'Model':model_name,
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  result = result.append({'Model':model_name,
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
```

```
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


Model: SVM
Classification reports
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       1.00      0.07      0.14        41
           5       0.63      0.57      0.60       306
           6       0.55      0.78      0.65       433
           7       0.49      0.26      0.34       158
           8       0.00      0.00      0.00        37

    accuracy                           0.57       980
   macro avg       0.45      0.28      0.29       980
weighted avg       0.56      0.57      0.53       980

Confusion Matrix
 [[  0   0   3   2   0   0]
 [  0   3  25  13   0   0]
 [  0   0 175 130   1   0]
 [  0   0  67 339  27   0]
 [  0   0   6 111  41   0]
 [  0   0   0  23  14   0]]

Model: Decision Tree
Classification reports
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.25      0.22      0.23        41
           5       0.62      0.57      0.60       306
           6       0.60      0.60      0.60       433
           7       0.47      0.56      0.51       158
           8       0.33      0.41      0.37        37
           9       0.00      0.00      0.00         0
    accuracy                           0.56       980
   macro avg       0.33      0.34      0.33       980
weighted avg       0.56      0.56      0.56       980

Confusion Matrix
 [[  0   0   3   2   0   0   0]
 [  0   9  18  11   2   1   0]
 [  0  14 175  98  17   1   1]
 [  0  13  73 258  68  21   0]
 [  0   0  11  52  88   7   0]
 [  0   0   1  10  11  15   0]
 [  0   0   0   0   0   0   0]]

Model: Gradient Boosting
Classification reports
              precision    recall  f1-score   support

           3       0.17      0.20      0.18         5
           4       0.60      0.15      0.24        41
           5       0.63      0.58      0.61       306
           6       0.57      0.73      0.64       433
           7       0.47      0.36      0.41       158
           8       0.50      0.14      0.21        37
           9       0.00      0.00      0.00         0

    accuracy                           0.57       980
   macro avg       0.42      0.31      0.33       980
weighted avg       0.57      0.57      0.56       980

Confusion Matrix
 [[  1   0   3   1   0   0   0]
 [  2   6  23  10   0   0   0]
 [  2   2 179 115   6   1   1]
 [  1   2  71 315  42   2   0]
 [  0   0   8  90  57   2   1]
 [  0   0   0  17  15   5   0]
 [  0   0   0   0   0   0   0]]
                 Model  Accuracy_score
0   Logistic Regression        0.538776
1  KNeighborsClassifier        0.537755
2         Random Forest        0.673469
3                   SVM        0.569388
4         Decision Tree        0.556122
5     Gradient Boosting        0.574490

C:\Users\Admin\AppData\Local\Temp\ipykernel_10844\164645047.py:8: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
```

```
  result = result.append({'Model':model_name,
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Admin\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[20]:

```
data= pd.DataFrame(result)
```

[21]:

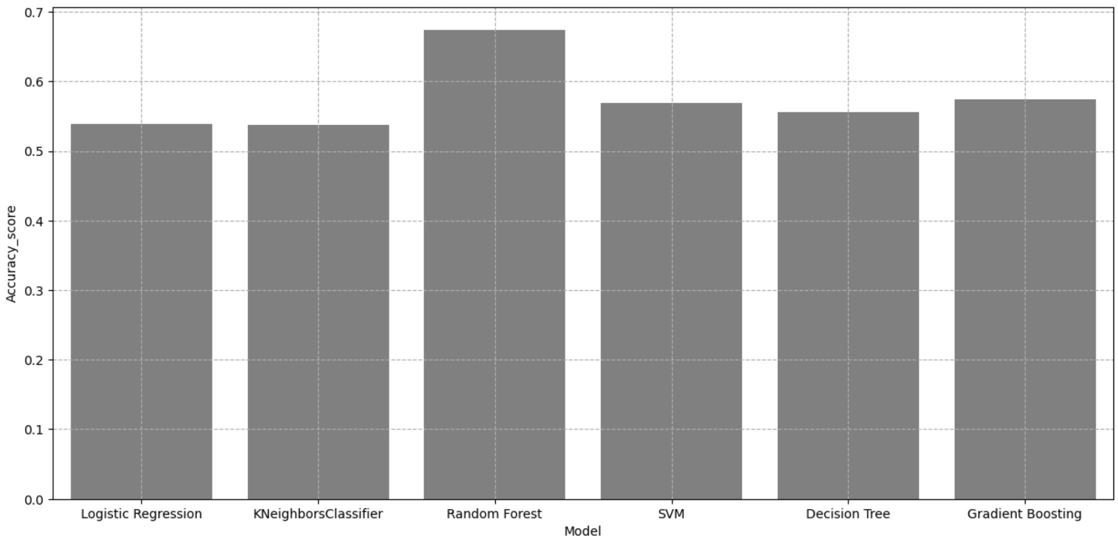
```
data
```

[21]:

```
                   Model  Accuracy_score
0    Logistic Regression        0.538776
1   KNeighborsClassifier        0.537755
2          Random Forest        0.673469
3                    SVM        0.569388
4          Decision Tree        0.556122
5      Gradient Boosting        0.574490
```

[26]:

```
plt.figure(figsize=(15,7))
sns.barplot(data=data, x='Model', y='Accuracy_score', color='grey')
plt.grid(linestyle='--')
```



0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Accuracy_score
Logistic Regression
KNeighborsClassifier
Random Forest
SVM
Decision Tree
Gradient Boosting
Model

```
[ ]:
```