Wine Quality Prediction

March 11, 2024

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

[2]:
```
data = pd.read_excel("WineQT.xlsx")
```

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

[3]:

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

Data Preprocessing

[4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1143 entries, 0 to 1142
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
```

```
 0   fixed acidity         1143 non-null   float64
 1   volatile acidity      1143 non-null   float64
 2   citric acid           1143 non-null   float64
 3   residual sugar        1143 non-null   float64
 4   chlorides             1143 non-null   float64
 5   free sulfur dioxide   1143 non-null   float64
 6   total sulfur dioxide  1143 non-null   float64
 7   density               1143 non-null   float64
 8   pH                    1143 non-null   float64
 9   sulphates             1143 non-null   float64
 10  alcohol               1143 non-null   float64
 11  quality               1143 non-null   int64
 12  Id                    1143 non-null   int64
dtypes: float64(11), int64(2)
memory usage: 116.2 KB
```

[6]:
```
data.isnull().sum()
```

[6]:
```
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
Id                      0
dtype: int64
```

Feature Selection

[7]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

[8]:
```
# Split the dataset into features (X) and target variable (y)
X = data.drop(["quality", "Id"], axis=1)
y = data["quality"]
```

[10]:
```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.
↪2,random_state=42)
```

[11]:
```
# Feature Scaling
scaler = StandardScaler()
```

```
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

MODEL BUILDING

[12]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import SGDClassifier
from sklearn.svm import SVC
```

[13]:
```
# Initialize classifier models
rf_model = RandomForestClassifier()
sgd_model = SGDClassifier()
svc_model = SVC()
```

[14]:
```
# Train the random forest model
rf_model.fit(X_train_scaled, y_train)
```

[14]:
```
RandomForestClassifier()
```

[15]:
```
# Train the sgd model
sgd_model.fit(X_train_scaled, y_train)
```

[15]:
```
SGDClassifier()
```

[16]:
```
 # Train the svc model
svc_model.fit(X_train_scaled, y_train)
```

[16]:
```
SVC()
```

MODEL EVALUATION

[17]:
```
from sklearn.metrics import classification_report
```

[18]:
```
# Evaluate Random Forest model
rf_predictions = rf_model.predict(X_test_scaled)
print("\033[1mRandom Forest Classifier:\033[0m")
print(classification_report(y_test, rf_predictions))
```

```
Random Forest Classifier:
              precision    recall  f1-score   support

           4       0.00      0.00      0.00         6
           5       0.69      0.78      0.74        96
           6       0.64      0.63      0.63        99
           7       0.67      0.62      0.64        26
           8       0.00      0.00      0.00         2

    accuracy                           0.67       229
   macro avg       0.40      0.40      0.40       229
```

```
weighted avg       0.64      0.67      0.65       229
```

```
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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))
```

[19]:

```
# Evaluate SGD model
sgd_predictions = sgd_model.predict(X_test_scaled)
print("\033[1mStochastic Gradient Descent Classifier:\033[0m")
print(classification_report(y_test, sgd_predictions))
```

Stochastic Gradient Descent Classifier:

```
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         0
           4       0.00      0.00      0.00         6
           5       0.67      0.55      0.61        96
           6       0.54      0.67      0.60        99
           7       0.28      0.27      0.27        26
           8       0.00      0.00      0.00         2

    accuracy                           0.55       229
   macro avg       0.25      0.25      0.25       229
weighted avg       0.55      0.55      0.54       229
```

```
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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]:
```
 # Evaluate SVC model
svc_predictions = svc_model.predict(X_test_scaled)
print("\033[1mSupport Vector Classifier:\033[0m")
print(classification_report(y_test, svc_predictions))
```

Support Vector Classifier:

	precision	recall	f1-score	support
4	0.00	0.00	0.00	6
5	0.70	0.74	0.72	96
6	0.59	0.69	0.64	99
7	0.54	0.27	0.36	26
8	0.00	0.00	0.00	2
accuracy			0.64	229
macro avg	0.37	0.34	0.34	229
weighted avg	0.61	0.64	0.62	229

```
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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))
```

[21]:
```
# Define classes and models
classes = np.unique(y_test)
models = ['Random Forest', 'Stochastic Gradient Descent', 'Support Vector␣
↪Classifier']
```

[22]:
```
# Initialize empty lists to store F1-scores for each class and model
f1_scores = {model: [] for model in models}
```

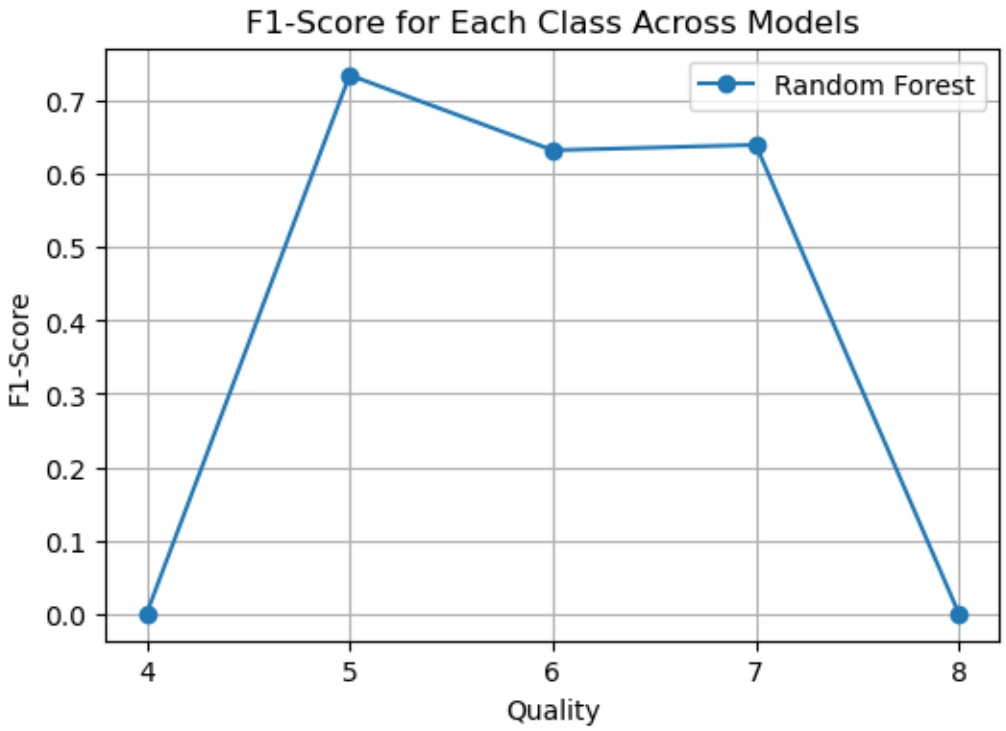
[23]:
```
 # Calculate F1-score for each model
for model_name, predictions in zip(models, [rf_predictions,␣
↪sgd_predictions,svc_predictions]):
    report = classification_report(y_test, predictions, output_dict=True)
    for class_label in classes:
        f1_scores[model_name].append(report[str(class_label)]['f1-score'])
```

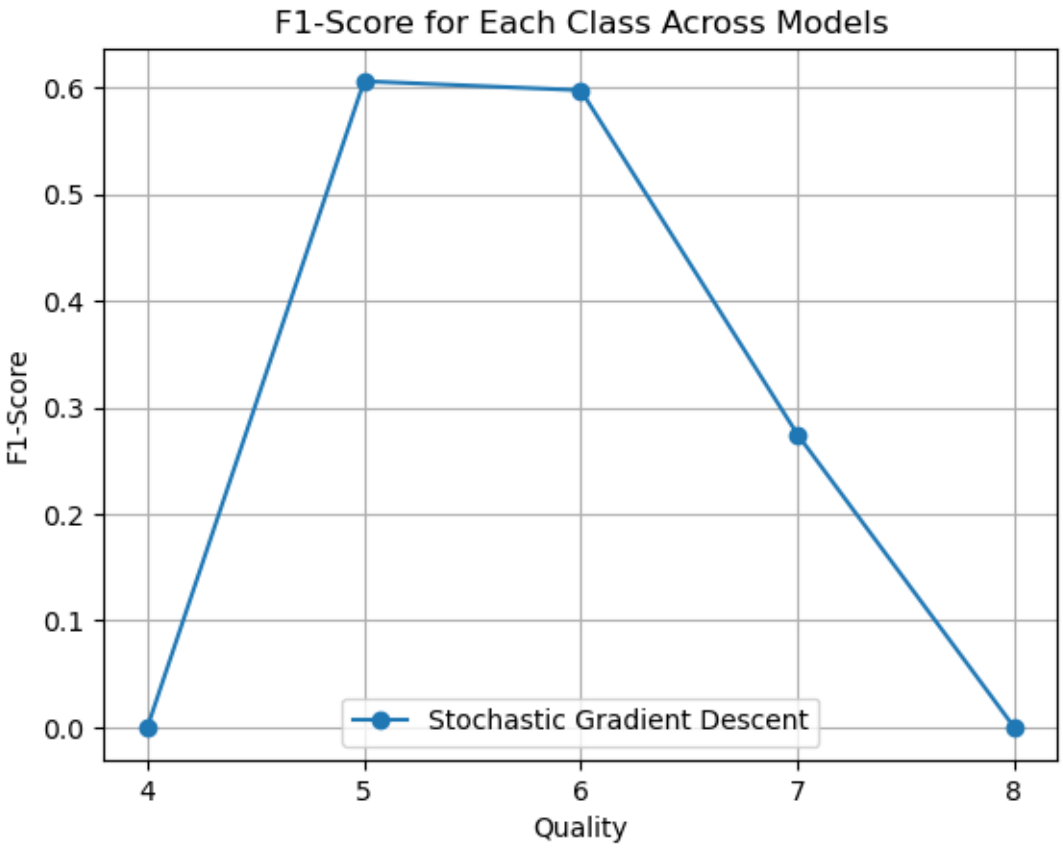
```
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
```

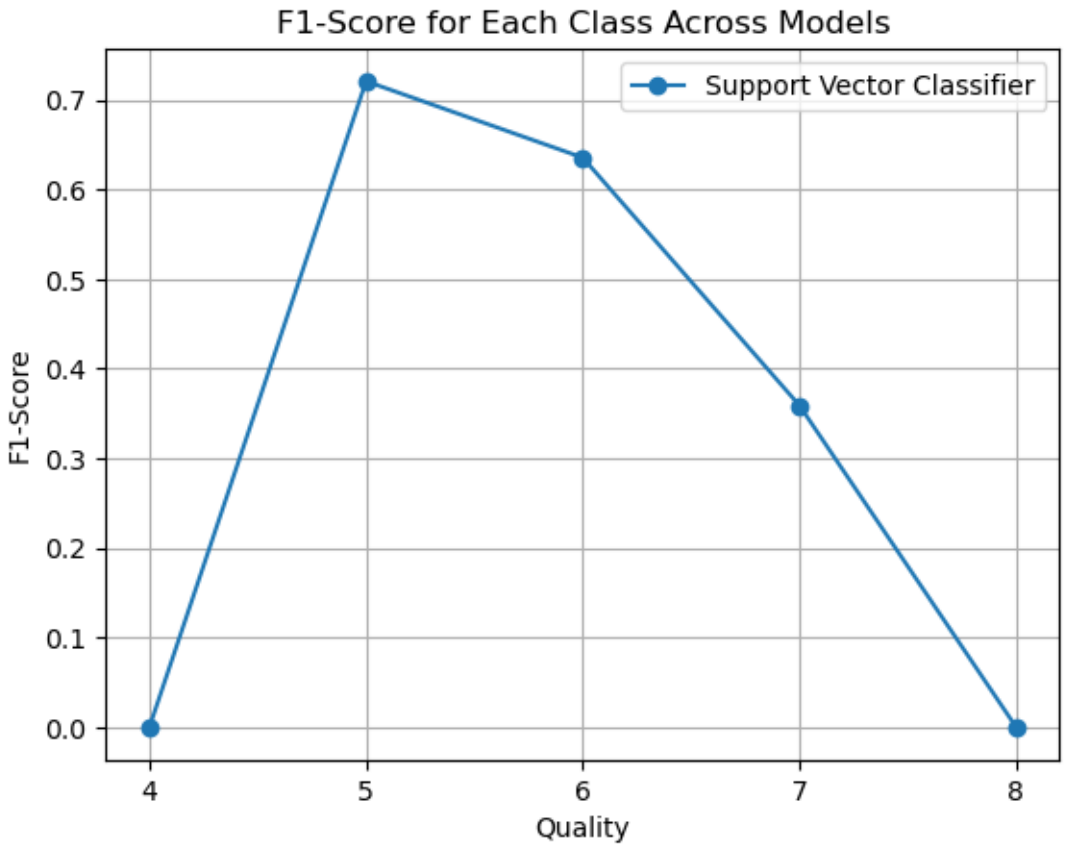
```
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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:\ProgramData\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: 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))
```

[24]:

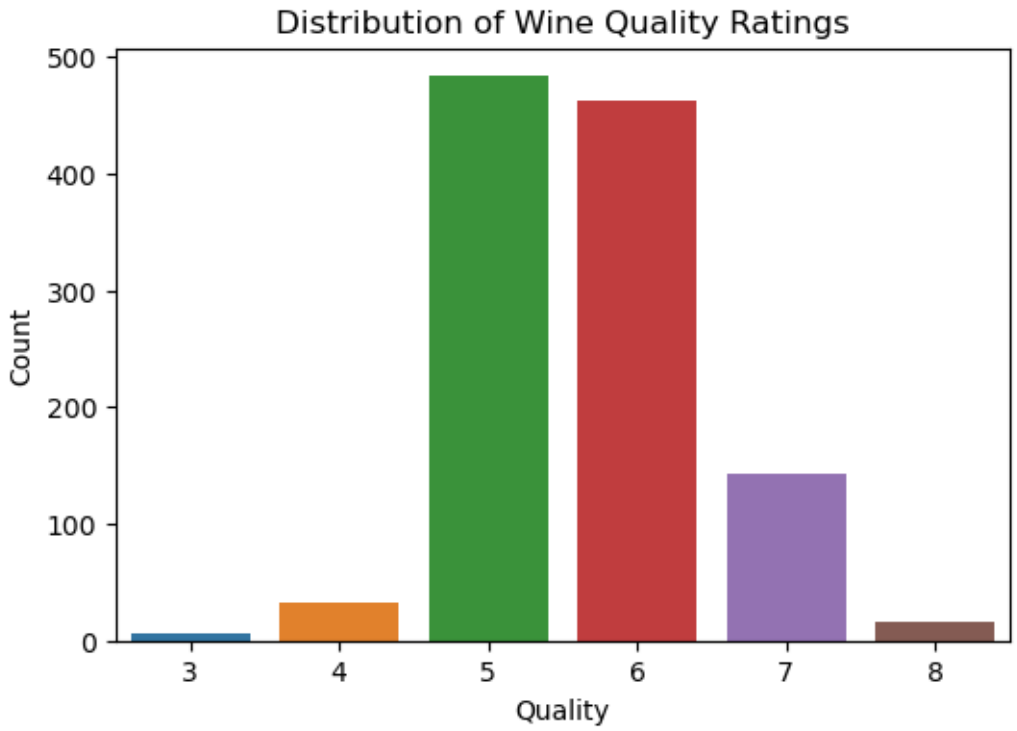
```
# Plot F1-scores for each class across models
plt.figure(figsize=(6, 4))
for model_name in models:
    plt.plot(classes, f1_scores[model_name], marker='o', label=model_name)
    plt.title('F1-Score for Each Class Across Models')
    plt.xlabel('Quality')
    plt.ylabel('F1-Score')
    plt.legend()
    plt.xticks(classes)
    plt.grid(True)
    plt.show()
```


F1-Score for Each Class Across Models
Random Forest
F1-Score
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
4
5
6
7
8
Quality

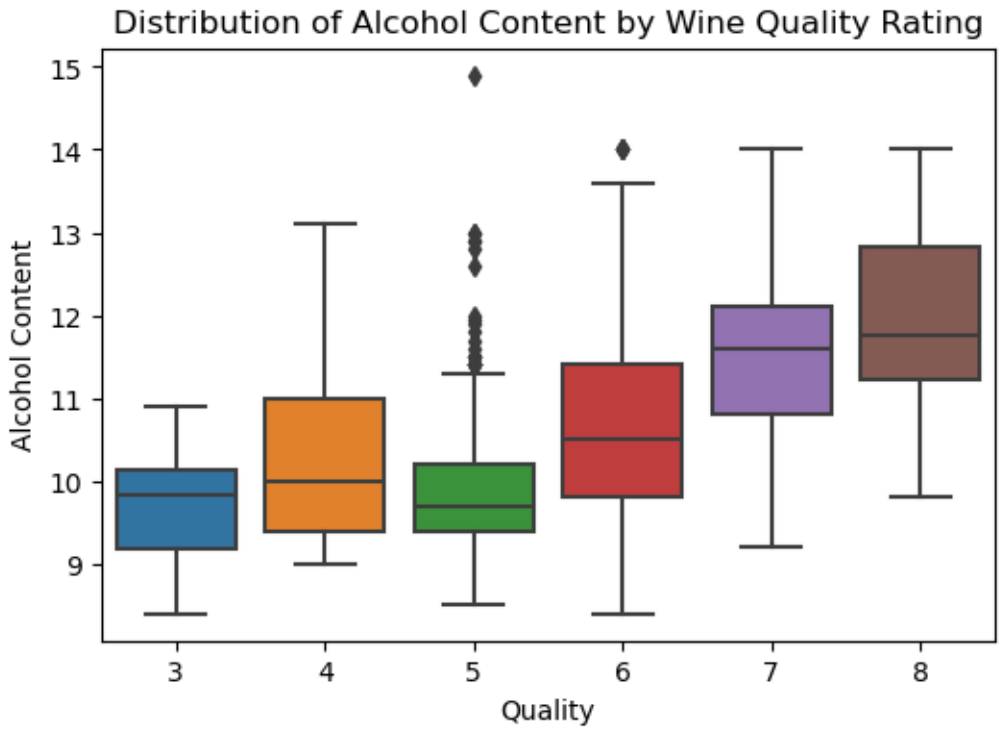

F1-Score for Each Class Across Models
F1-Score
0.0
0.1
0.2
0.3
0.4
0.5
0.6
4
5
6
7
8
Quality
Stochastic Gradient Descent



[25]:
```
# Visualize the distribution of wine quality ratings
plt.figure(figsize=(6, 4))
sns.countplot(x='quality', data=data)
plt.title('Distribution of Wine Quality Ratings')
plt.xlabel('Quality')
plt.ylabel('Count')
plt.show()
```



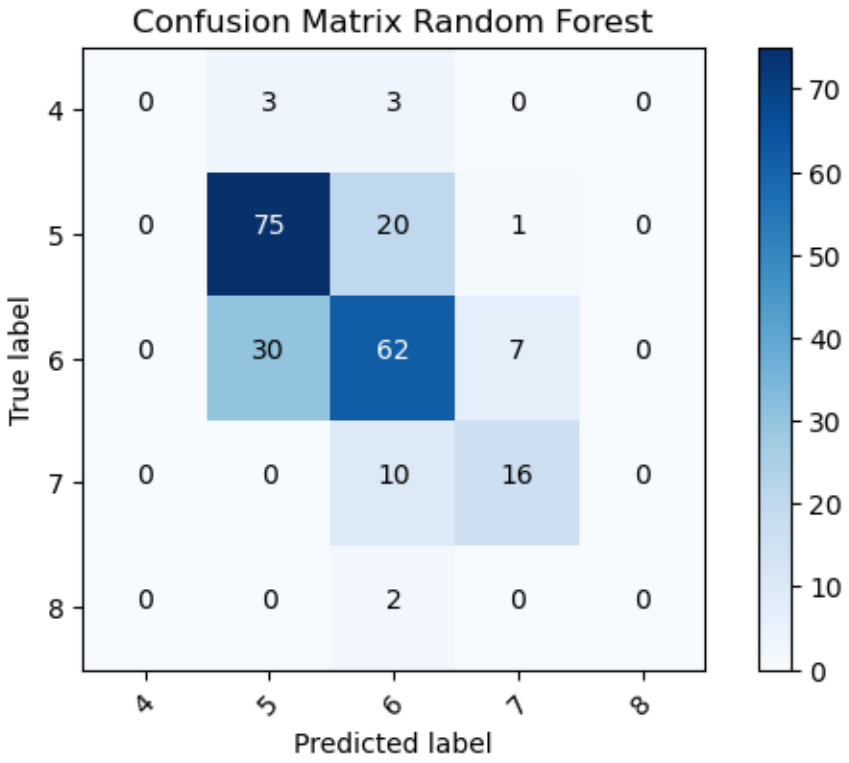
[26]:
```
# Boxplot to visualize distribution of features by quality rating
plt.figure(figsize=(6, 4))
sns.boxplot(x='quality', y='alcohol', data=data)
plt.title('Distribution of Alcohol Content by Wine Quality Rating')
plt.xlabel('Quality')
plt.ylabel('Alcohol Content')
plt.show()
```



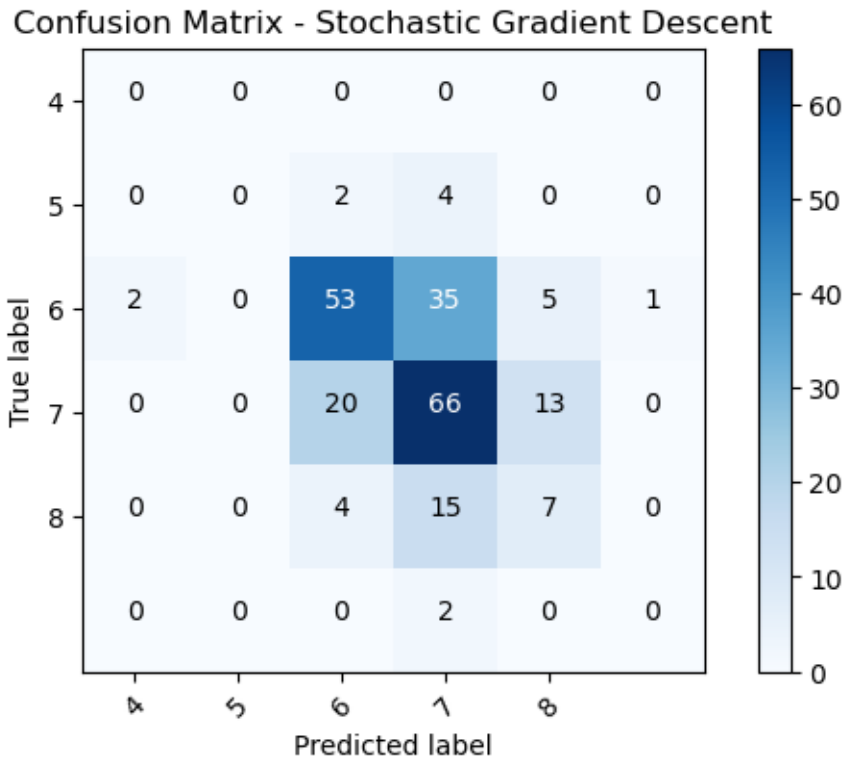
[27]:
```
from sklearn.metrics import confusion_matrix
import itertools
# Function to plot confusion matrix
def plot_confusion_matrix(cm, classes, title='Confusion matrix', cmap=plt.cm.
 ↪Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)
    fmt = 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j],␣
 ↪fmt),horizontalalignment="center",color="white" if cm[i, j] > thresh else␣
 ↪"black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

[28]:
```
 # Calculate confusion matrix for Random Forest model
rf_cm = confusion_matrix(y_test, rf_predictions)
```

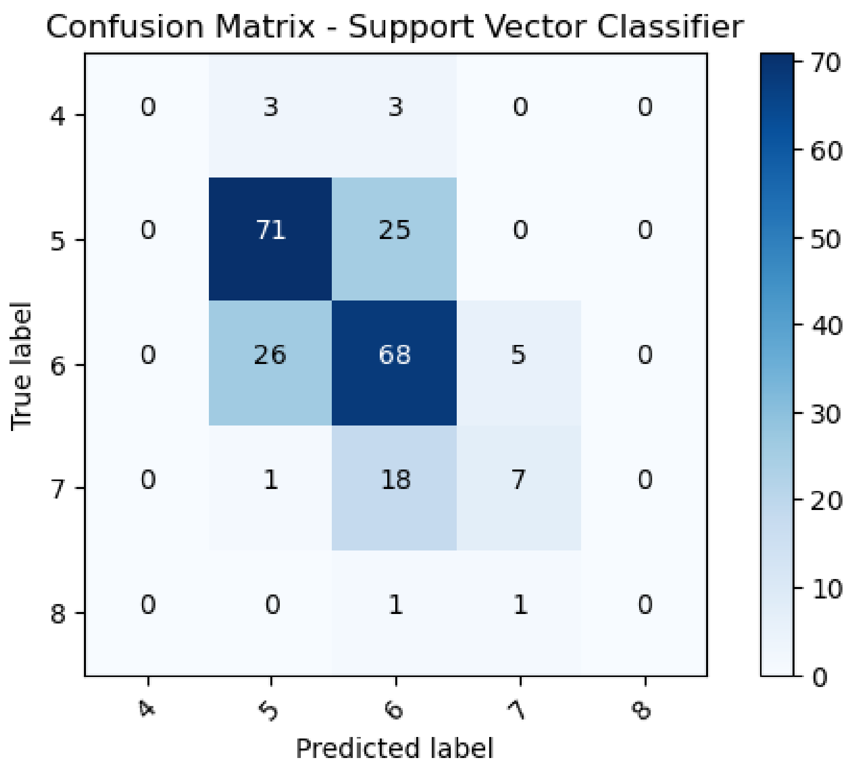
```
# Plot confusion matrix
plt.figure(figsize=(6, 4))
plot_confusion_matrix(rf_cm, classes=np.unique(y_test), title='Confusion Matrix␣
 ↪Random Forest')
plt.show()
```



[29]:
```
# Calculate confusion matrix for SGD model
sgd_cm = confusion_matrix(y_test, sgd_predictions)
# Plot confusion matrix
plt.figure(figsize=(6, 4))
plot_confusion_matrix(sgd_cm, classes=np.unique(y_test), title='Confusion␣
 ↪Matrix - Stochastic Gradient Descent')
plt.show()
```



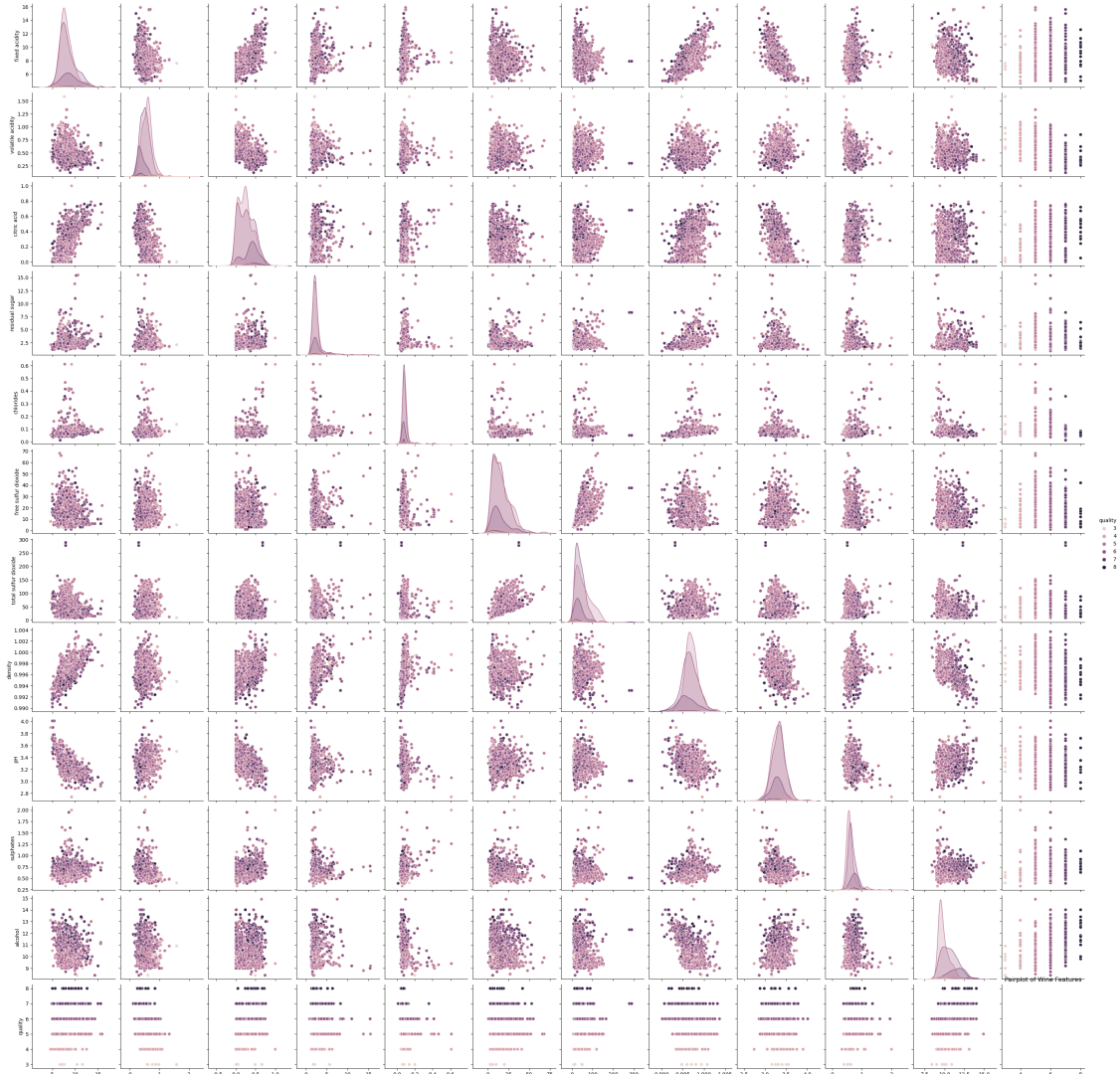
[30]:
```
 # Calculate confusion matrix for SVC model
svc_cm = confusion_matrix(y_test, svc_predictions)
# Plot confusion matrix
plt.figure(figsize=(6, 4))
plot_confusion_matrix(svc_cm, classes=np.unique(y_test), title='Confusion␣
 ↪Matrix - Support Vector Classifier')
plt.show()
```



[35]:
```
# Pairplot to visualize relationships between features
sns.pairplot(data=data, vars=['fixed acidity', 'volatile acidity', 'citric␣
↪acid', 'residual sugar','chlorides', 'free sulfur dioxide', 'total sulfur␣
↪dioxide', 'density', 'pH','sulphates', 'alcohol', 'quality'], hue='quality')
plt.title('Pairplot of Wine Features')
plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



Pairplot of Wine Features