

```
In [8]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectKBest, f_classif, SelectFromModel
from sklearn.metrics import classification_report, accuracy_score

print("تم استيراد المكتبات بنجاح")
```

تم استيراد المكتبات بنجاح

```
In [9]: df = pd.read_csv('WineQT.csv')

if 'Id' in df.columns:
    df = df.drop(columns=['Id'])

print("شكل البيانات الحالي:", df.shape)
df.head()
```

شكل البيانات الحالي: (12, 1143)

Out [9]:

	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 [10]: missing_values = df.isnull().sum()
print("عدد القيم المفقودة في كل عمود")

df.fillna(df.mean(), inplace=True)

print("\nتمت معالجة القيم المفقودة بنجاح")
```

عدد القيم المفقودة في كل عمود:

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

تمت معالجة القيم المفقودة بنجاح

```
In [11]: X = df.drop('quality', axis=1)
y = df['quality']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

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

print("تم تقسيم وتجهيز البيانات للتدريب")
```

تم تقسيم وتجهيز البيانات للتدريب

```
In [12]: rf = RandomForestClassifier(random_state=42)
param_grid = {
    'n_estimators': [50, 100],
    'max_depth': [None, 10, 20],
    'criterion': ['gini', 'entropy']
}

grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=3)
grid_search.fit(X_train_scaled, y_train)

print("أفضل المعاملات المكتشفة:", grid_search.best_params_)
y_pred = grid_search.predict(X_test_scaled)
print(f"(Accuracy) دقة النموذج: {accuracy_score(y_test, y_pred):.2f}")
```

{'criterion': 'gini', 'max_depth': 20, 'n_estimators': 50}: أفضل المعاملات المكتشفة
(Accuracy) دقة النموذج: 0.67

```
In [13]: selector_kbest = SelectKBest(score_func=f_classif, k=5)
X_train_kbest = selector_kbest.fit_transform(X_train_scaled, y_train)
X_test_kbest = selector_kbest.transform(X_test_scaled)

selected_features_kbest = X.columns[selector_kbest.get_support()]
```

```
print("الميزات المختارة باستخدام SelectKBest:\n", list(selected_features_kbest))
```

SelectKBest: الميزات المختارة باستخدام
['volatile acidity', 'citric acid', 'total sulfur dioxide', 'sulphates', 'alcohol']

In [14]: selector_sfm = SelectFromModel(RandomForestClassifier(n_estimators=100, random_state=42))
X_train_sfm = selector_sfm.fit_transform(X_train_scaled, y_train)
X_test_sfm = selector_sfm.transform(X_test_scaled)

selected_features_sfm = X.columns[selector_sfm.get_support()]
print("الميزات المختارة باستخدام SelectFromModel:\n", list(selected_features_sfm))

SelectFromModel: الميزات المختارة باستخدام
['volatile acidity', 'total sulfur dioxide', 'sulphates', 'alcohol']

In [15]: model_final = RandomForestClassifier(random_state=42)
model_final.fit(X_train_sfm, y_train)
y_pred_final = model_final.predict(X_test_sfm)

print("--- تقرير الأداء النهائي بعد اختيار الميزات ---")
print(classification_report(y_test, y_pred_final))
print(f"الدقة النهائية: {accuracy_score(y_test, y_pred_final):.2f}")

--- تقرير الأداء النهائي بعد اختيار الميزات ---

	precision	recall	f1-score	support
4	0.00	0.00	0.00	6
5	0.72	0.76	0.74	96
6	0.64	0.65	0.64	99
7	0.62	0.62	0.62	26
8	0.00	0.00	0.00	2
accuracy			0.67	229
macro avg	0.40	0.40	0.40	229
weighted avg	0.65	0.67	0.66	229

الدقة النهائية: 0.67

In [16]: print("Project Submitted by: Roba Ahmed Abusaifin")

Project Submitted by: Roba Ahmed Abusaifin

In []: