Crop2

January 25, 2024

[18]:
```
# Importing libraries

from __future__ import print_function
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn import tree
from sklearn.metrics import confusion_matrix
import warnings
warnings.filterwarnings('ignore')
```

[19]:
```
df = pd.read_csv('Crop.csv')
```

[20]:
```
df.head()
```

[20]:

	N	P	K	ph	EC	S	Cu	Fe	Mn	Zn	B	\
0	143	69	217	5.9	0.58	0.23	10.20	116.35	59.96	54.85	21.29	\
1	170	36	216	5.9	0.15	0.28	15.69	114.20	56.87	31.28	28.62	
2	158	66	219	6.8	0.34	0.20	15.29	65.87	51.81	57.12	27.59	
3	133	45	207	6.4	0.94	0.21	8.48	103.10	43.81	68.50	47.29	
4	132	48	218	6.7	0.54	0.19	5.59	63.40	56.40	46.71	31.04	

	label
0	pomegranate
1	pomegranate
2	pomegranate
3	pomegranate
4	pomegranate

[21]:
```
df.tail()
```

[21]:

	N	P	K	ph	EC	S	Cu	Fe	Mn	Zn	B	\
615	41	23	135	5.0	1.67	0.10655	26.0	39.2	206.89	31.09	20.64	\
616	49	45	90	5.8	1.98	0.09229	19.0	40.2	91.12	32.68	14.91	
617	131	24	121	4.9	2.24	0.08775	22.0	40.0	94.34	24.93	23.74	

```
618  131  55  130  5.3  2.48  0.08983  15.0  41.0   92.58  45.73  21.48
619  129  34  160  4.8  1.08  0.08869  25.0  39.0  259.93  33.49  14.16

      label
615  potato
616  potato
617  potato
618  potato
619  potato
```

[22]:
```
df.size
```

[22]:
```
7440
```

[23]:
```
df.shape
```

[23]:
```
(620, 12)
```

[24]:
```
df.columns
```

[24]:
```
Index(['N', 'P', 'K', 'ph', 'EC', 'S', 'Cu', 'Fe', 'Mn', 'Zn', 'B', 'label'],
dtype='object')
```

[25]:
```
df['label'].unique()
```

[25]:
```
array(['pomegranate', 'mango', 'grapes', 'mulberry', 'ragi', 'potato'],
      dtype=object)
```

[26]:
```
df.dtypes
```

[26]:
```
N          int64
P          int64
K          int64
ph       float64
EC       float64
S        float64
Cu       float64
Fe       float64
Mn       float64
Zn       float64
B        float64
label     object
dtype: object
```

[27]:
```
df['label'].value_counts()
```

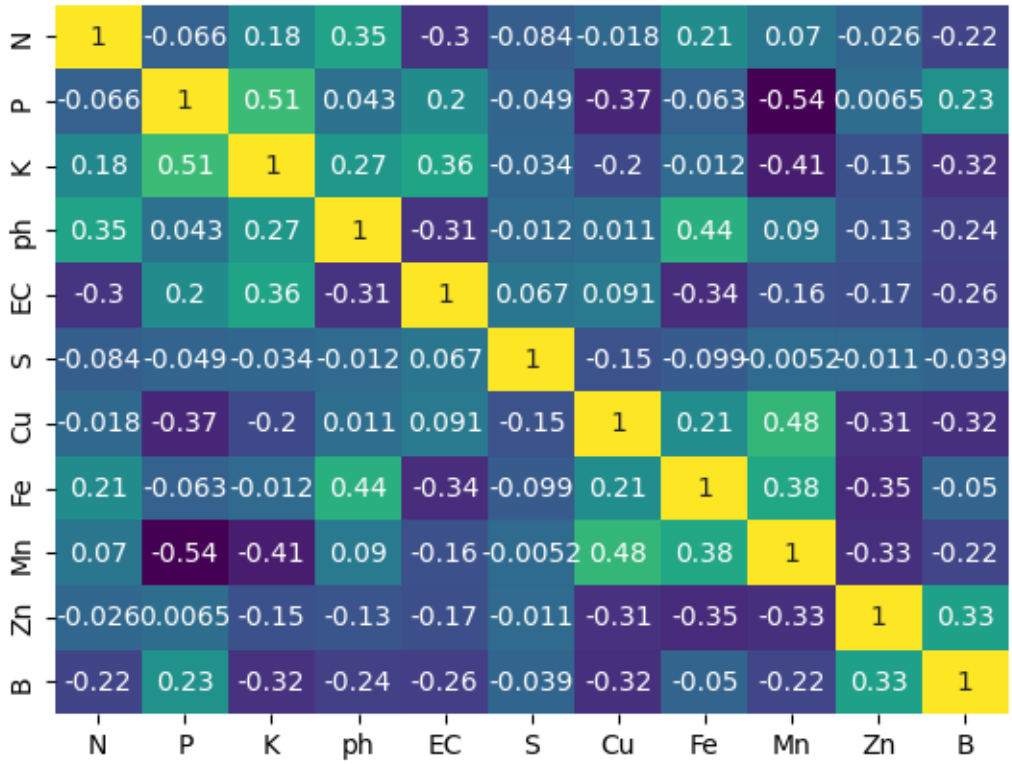
[27]:
```
label
pomegranate    104
```

```
mango          104
grapes         104
mulberry       104
ragi           104
potato         100
Name: count, dtype: int64
```

[28]:
```
df_new = df.copy()

df_new.drop('label', axis=1, inplace=True)
```

[29]:
```
sns.heatmap(df_new.corr(),annot=True, cbar=False, cmap='viridis')
```

[29]: <Axes: >



Seperating features and target label

[30]:
```
features = df[['N', 'P', 'K', 'ph', 'EC', 'S', 'Cu', 'Fe', 'Mn', 'Zn', 'B']]
target = df['label']
labels = df['label']
```

[31]:
```
# Initializing empty lists to append all model's name and corresponding name
acc = []
model = []
```

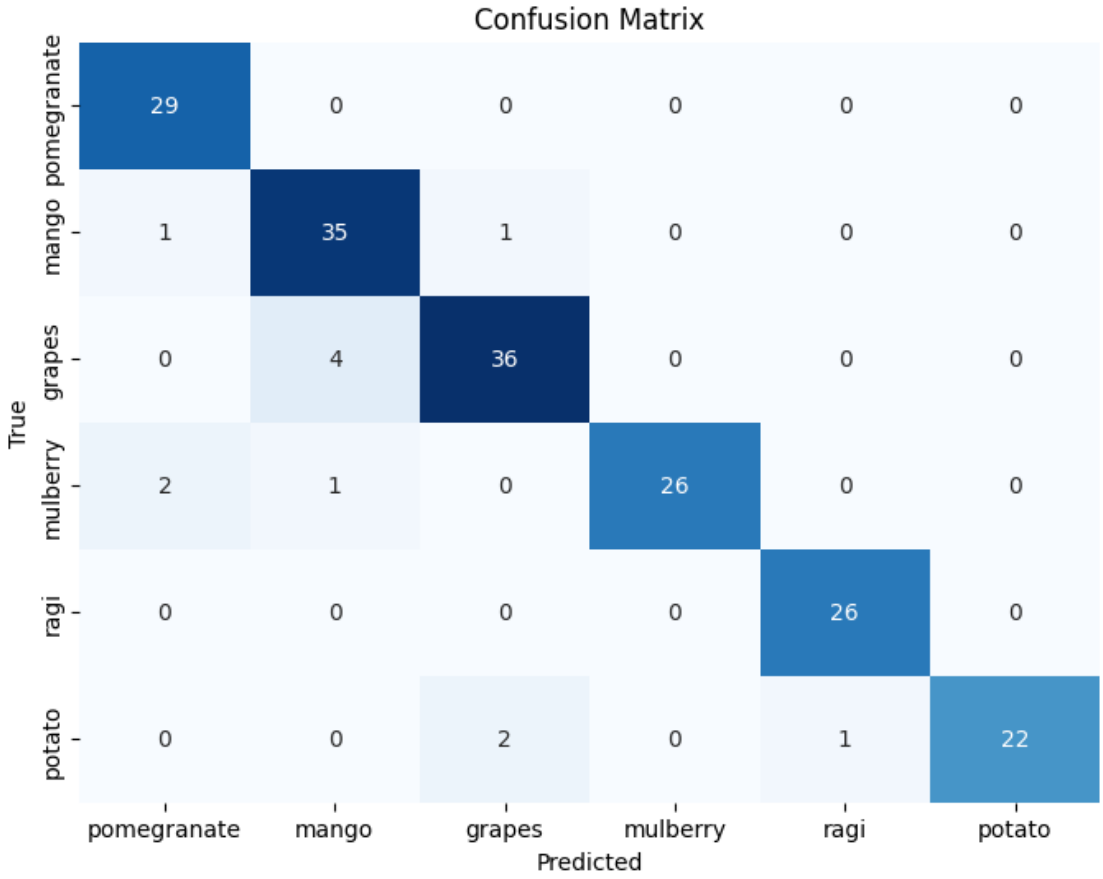
[32]:
```
# Splitting into train and test data

from sklearn.model_selection import train_test_split
Xtrain, Xtest, Ytrain, Ytest = train_test_split(features,target,test_size = 0.
 ↪3,random_state =3)
```

1 Decision Tree

[33]:
```
from sklearn.tree import DecisionTreeClassifier

DecisionTree =␣
 ↪DecisionTreeClassifier(criterion="entropy",random_state=2,max_depth=5)

DecisionTree.fit(Xtrain,Ytrain)

predicted_values = DecisionTree.predict(Xtest)
x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('Decision Tree')
print("DecisionTrees's Accuracy is: ", x*100)

print(classification_report(Ytest,predicted_values))

# Create confusion matrix
cm = confusion_matrix(Ytest,predicted_values)

# Plot confusion matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',cbar=False,␣
 ↪xticklabels=df['label'].unique(), yticklabels=df['label'].unique())
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

```
DecisionTrees's Accuracy is:  93.54838709677419
              precision    recall  f1-score   support

      grapes       0.91      1.00      0.95        29
       mango       0.88      0.95      0.91        37
    mulberry       0.92      0.90      0.91        40
 pomegranate       1.00      0.90      0.95        29
      potato       0.96      1.00      0.98        26
```

ragi	1.00	0.88	0.94	25
accuracy			0.94	186
macro avg	0.94	0.94	0.94	186
weighted avg	0.94	0.94	0.94	186

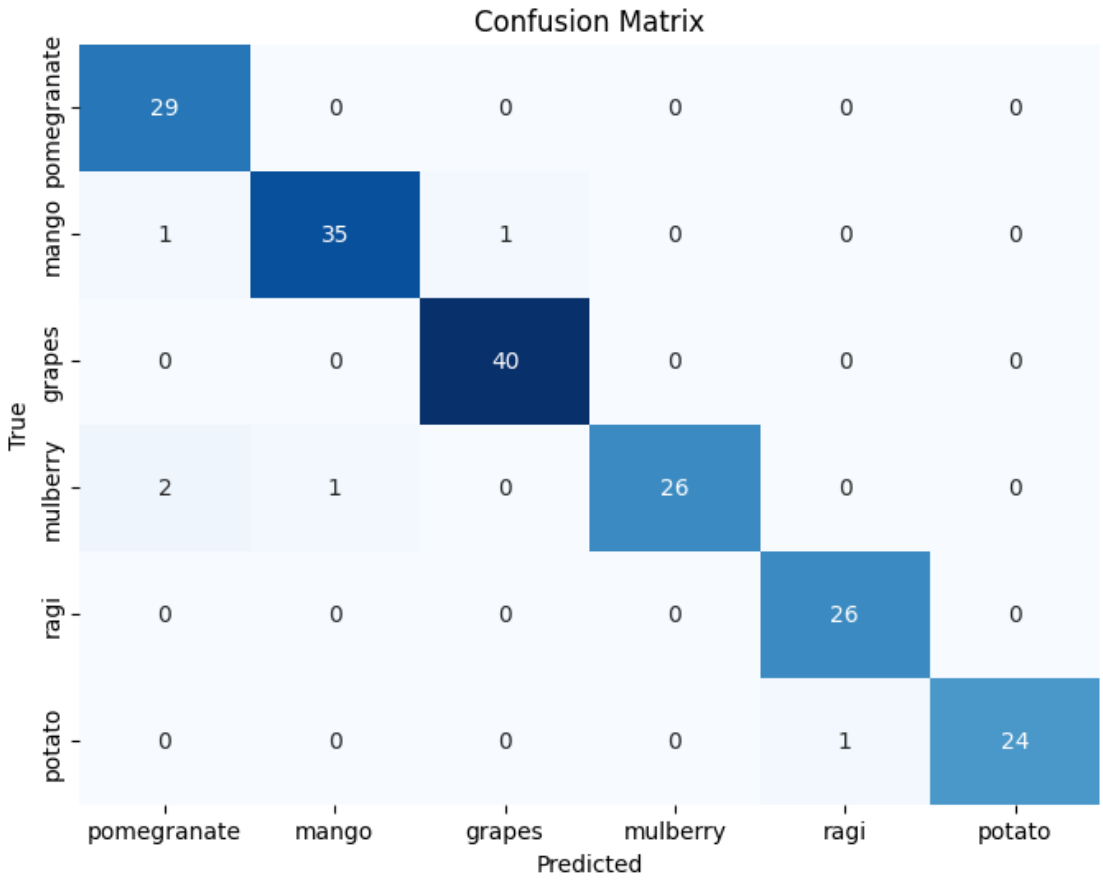


2 Guassian Naive Bayes

[34]:
```
from sklearn.naive_bayes import GaussianNB

NaiveBayes = GaussianNB()

NaiveBayes.fit(Xtrain,Ytrain)

predicted_values = NaiveBayes.predict(Xtest)
x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
```

```
model.append('Naive Bayes')
print("Naive Bayes's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))

# Create confusion matrix
cm = confusion_matrix(Ytest,predicted_values)

# Plot confusion matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',␣
 ↪cbar=False,xticklabels=df['label'].unique(), yticklabels=df['label'].
 ↪unique())
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

```
Naive Bayes's Accuracy is:  0.967741935483871
              precision    recall  f1-score   support

      grapes       0.91      1.00      0.95        29
       mango       0.97      0.95      0.96        37
    mulberry       0.98      1.00      0.99        40
 pomegranate       1.00      0.90      0.95        29
      potato       0.96      1.00      0.98        26
        ragi       1.00      0.96      0.98        25

    accuracy                           0.97       186
   macro avg       0.97      0.97      0.97       186
weighted avg       0.97      0.97      0.97       186
```

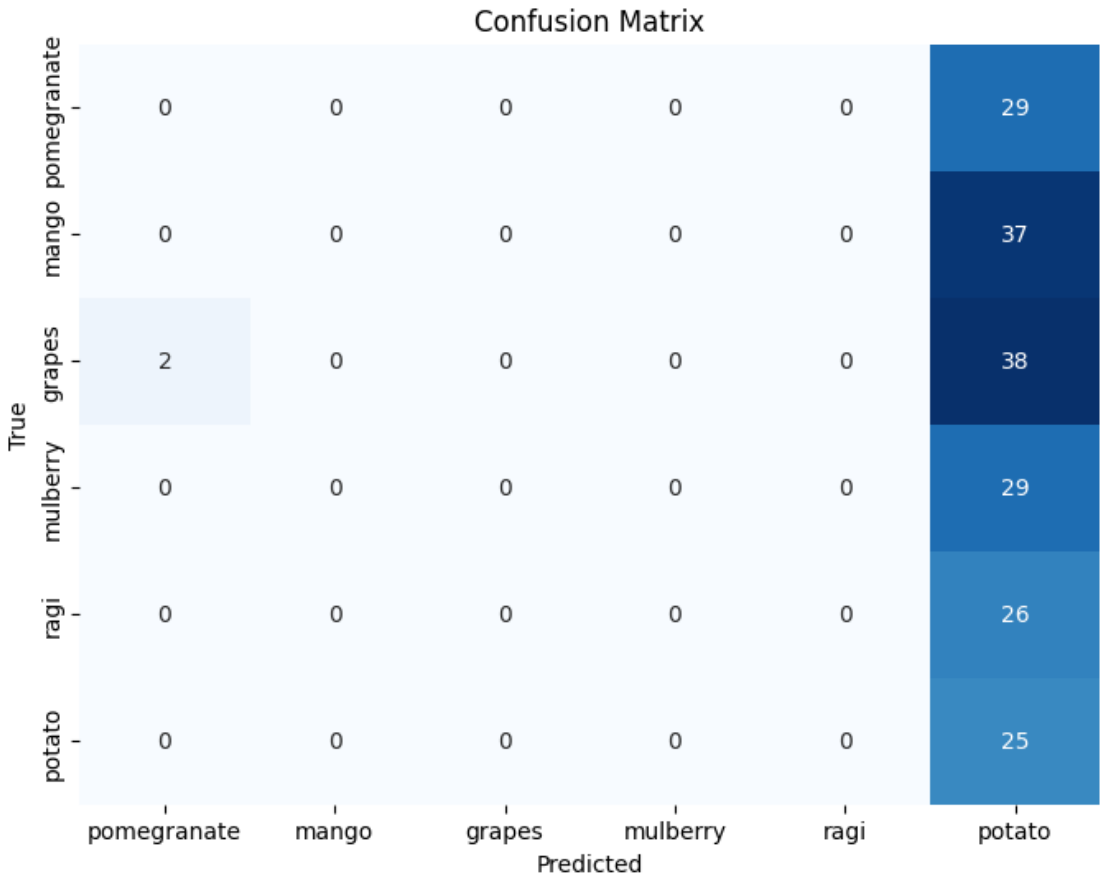


3 Support Vector Machine (SVM)

[35]:
```
from sklearn.svm import SVC

SVM = SVC(gamma='auto')

SVM.fit(Xtrain,Ytrain)

predicted_values = SVM.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('SVM')
print("SVM's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))

# Create confusion matrix
```

```
cm = confusion_matrix(Ytest,predicted_values)

# Plot confusion matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',cbar=False,␣
 ↪xticklabels=df['label'].unique(), yticklabels=df['label'].unique())
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

```
SVM's Accuracy is:  0.13440860215053763
              precision    recall  f1-score   support

      grapes       0.00      0.00      0.00        29
       mango       0.00      0.00      0.00        37
    mulberry       0.00      0.00      0.00        40
 pomegranate       0.00      0.00      0.00        29
      potato       0.00      0.00      0.00        26
        ragi       0.14      1.00      0.24        25

    accuracy                           0.13       186
   macro avg       0.02      0.17      0.04       186
weighted avg       0.02      0.13      0.03       186
```

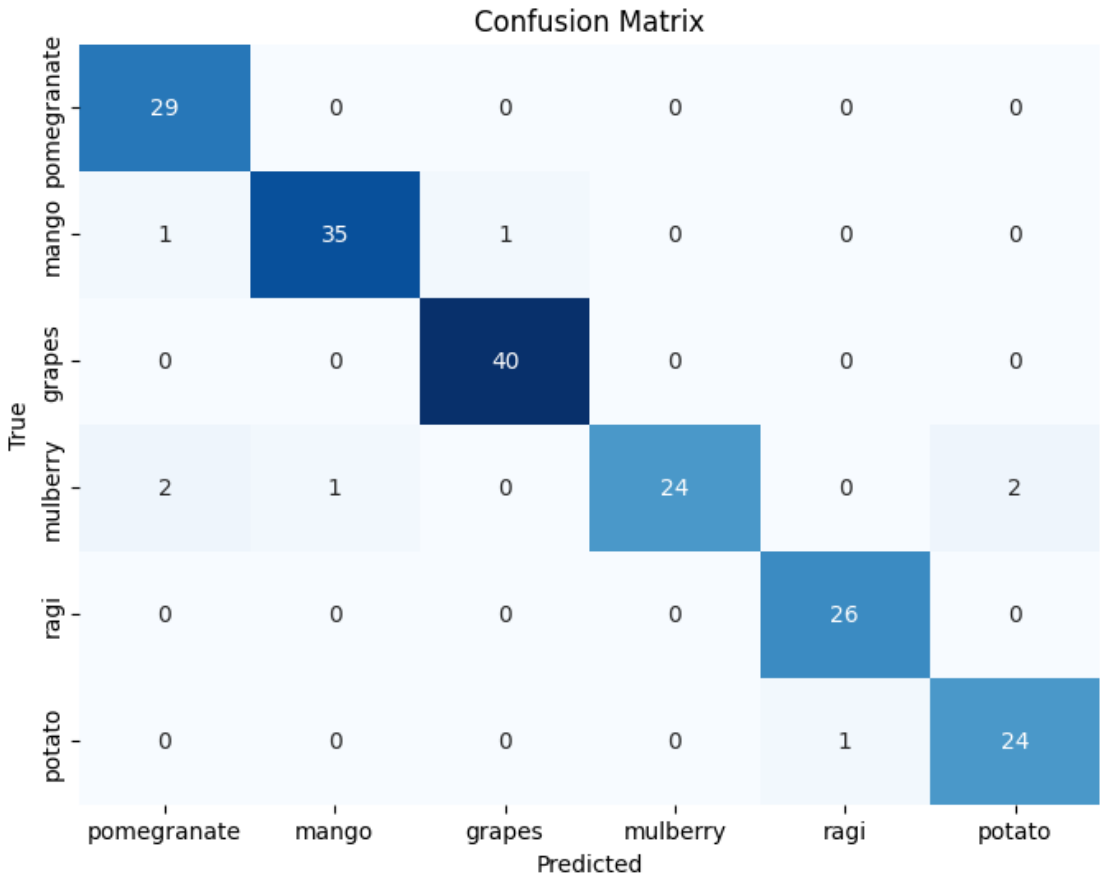


4 Logistic Regression

[36]:
```
from sklearn.linear_model import LogisticRegression

LogReg = LogisticRegression(random_state=2)

LogReg.fit(Xtrain,Ytrain)

predicted_values = LogReg.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('Logistic Regression')
print("Logistic Regression's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))

# Create confusion matrix
```

```
cm = confusion_matrix(Ytest,predicted_values)

# Plot confusion matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',cbar=False,␣
 ↪xticklabels=df['label'].unique(), yticklabels=df['label'].unique())
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

```
Logistic Regression's Accuracy is:  0.956989247311828
              precision    recall  f1-score   support

      grapes       0.91      1.00      0.95        29
       mango       0.97      0.95      0.96        37
    mulberry       0.98      1.00      0.99        40
 pomegranate       1.00      0.83      0.91        29
      potato       0.96      1.00      0.98        26
        ragi       0.92      0.96      0.94        25

    accuracy                           0.96       186
   macro avg       0.96      0.96      0.95       186
weighted avg       0.96      0.96      0.96       186
```

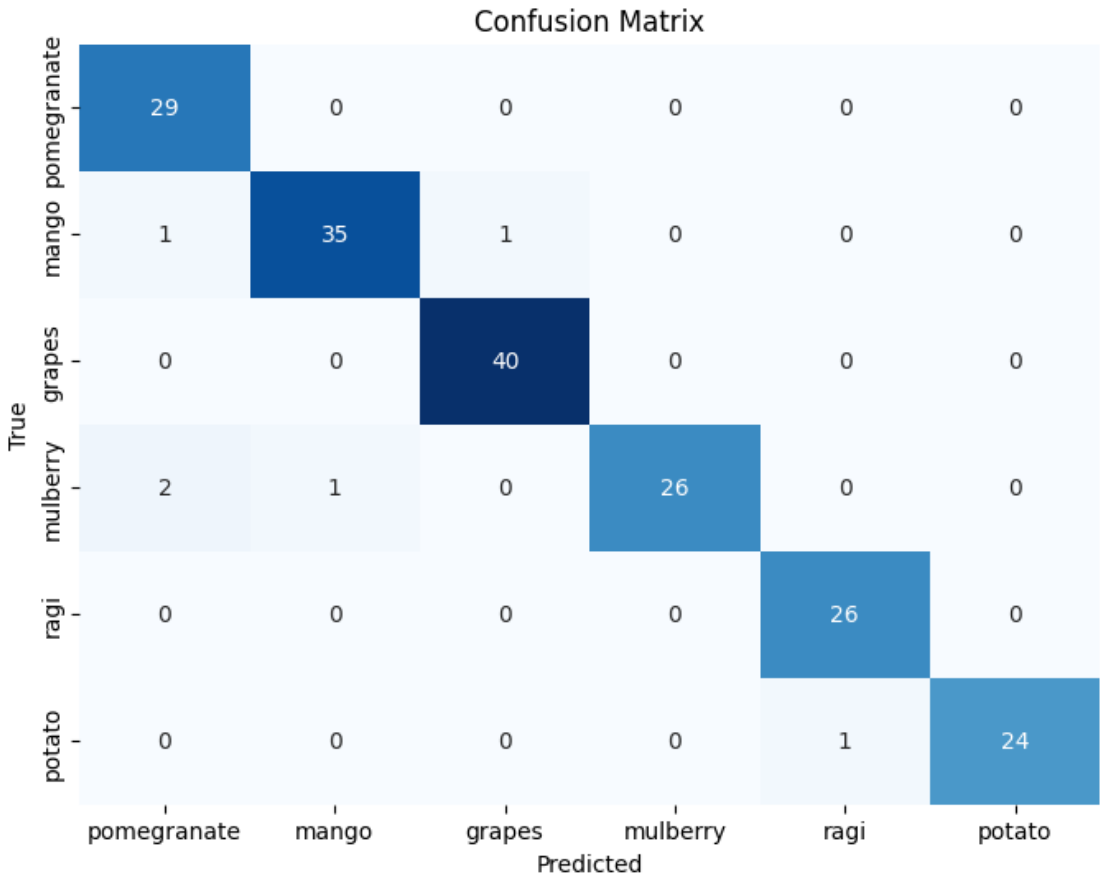


5 Random Forest

[37]:
```
from sklearn.ensemble import RandomForestClassifier

RF = RandomForestClassifier(n_estimators=20, random_state=0)
RF.fit(Xtrain,Ytrain)

predicted_values = RF.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('RF')
print("RF's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))

# Create confusion matrix
cm = confusion_matrix(Ytest,predicted_values)
```

```
# Plot confusion matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',␣
 ↪cbar=False,xticklabels=df['label'].unique(), yticklabels=df['label'].
 ↪unique())
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

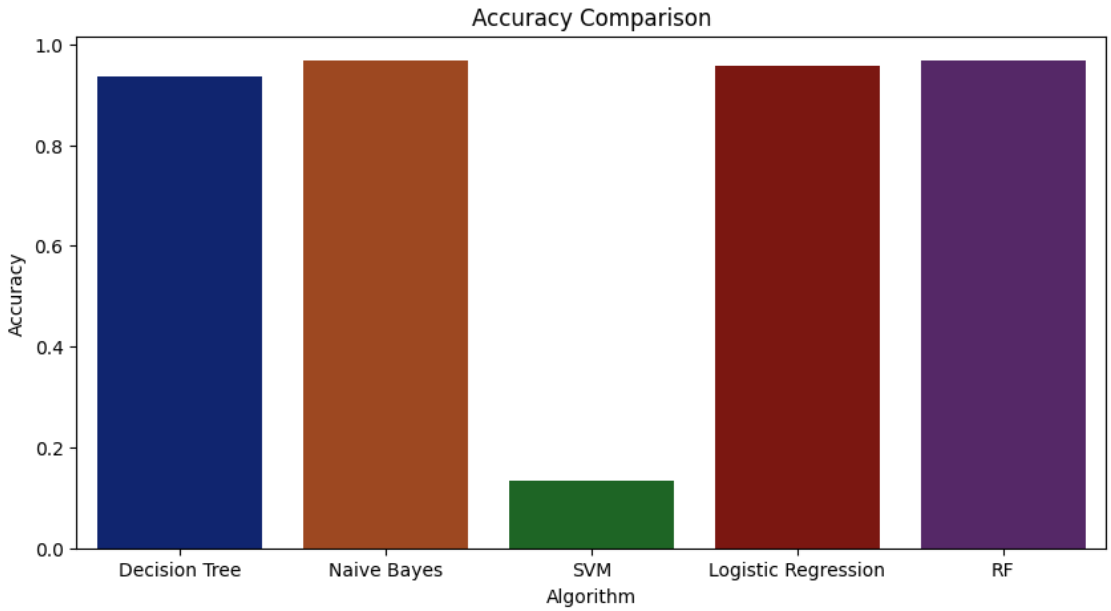
```
RF's Accuracy is:  0.967741935483871
              precision    recall  f1-score   support

      grapes       0.91      1.00      0.95        29
       mango       0.97      0.95      0.96        37
    mulberry       0.98      1.00      0.99        40
 pomegranate       1.00      0.90      0.95        29
      potato       0.96      1.00      0.98        26
        ragi       1.00      0.96      0.98        25

    accuracy                           0.97       186
   macro avg       0.97      0.97      0.97       186
weighted avg       0.97      0.97      0.97       186
```



6 Accuracy Comparison

[38]:
```
plt.figure(figsize=[10,5],dpi = 100)
plt.title('Accuracy Comparison')
plt.xlabel('Algorithm')
plt.ylabel('Accuracy')
sns.barplot(x = model,y = acc,palette='dark')
```

[38]:
```
<Axes: title={'center': 'Accuracy Comparison'}, xlabel='Algorithm',
ylabel='Accuracy'>
```



[39]:
```
accuracy_models = dict(zip(model, acc))

from prettytable import PrettyTable
table = PrettyTable()
table.field_names = ["Name", "Accuracy"]

for k,v in accuracy_models.items():
    v = round(v, 2)
    table.add_row([k,v])

print(table)
```

```
+---------------------+----------+
|         Name        | Accuracy |
+---------------------+----------+
|    Decision Tree    |   0.94   |
|     Naive Bayes     |   0.97   |
|         SVM         |   0.13   |
| Logistic Regression |   0.96   |
|          RF         |   0.97   |
+---------------------+----------+
```

7 Making Predictions

[41]:
```
data = np.array([[150,70,217,6,0.6,0.25,10,116,60,55,22]])
prediction = RF.predict(data)
print(prediction)
```

```
['pomegranate']
```

[42]:
```
data = np.array([[50,20,130,5,2,0.1,26,40,206,32,20]])
prediction = RF.predict(data)
print(prediction)
```

```
['potato']
```