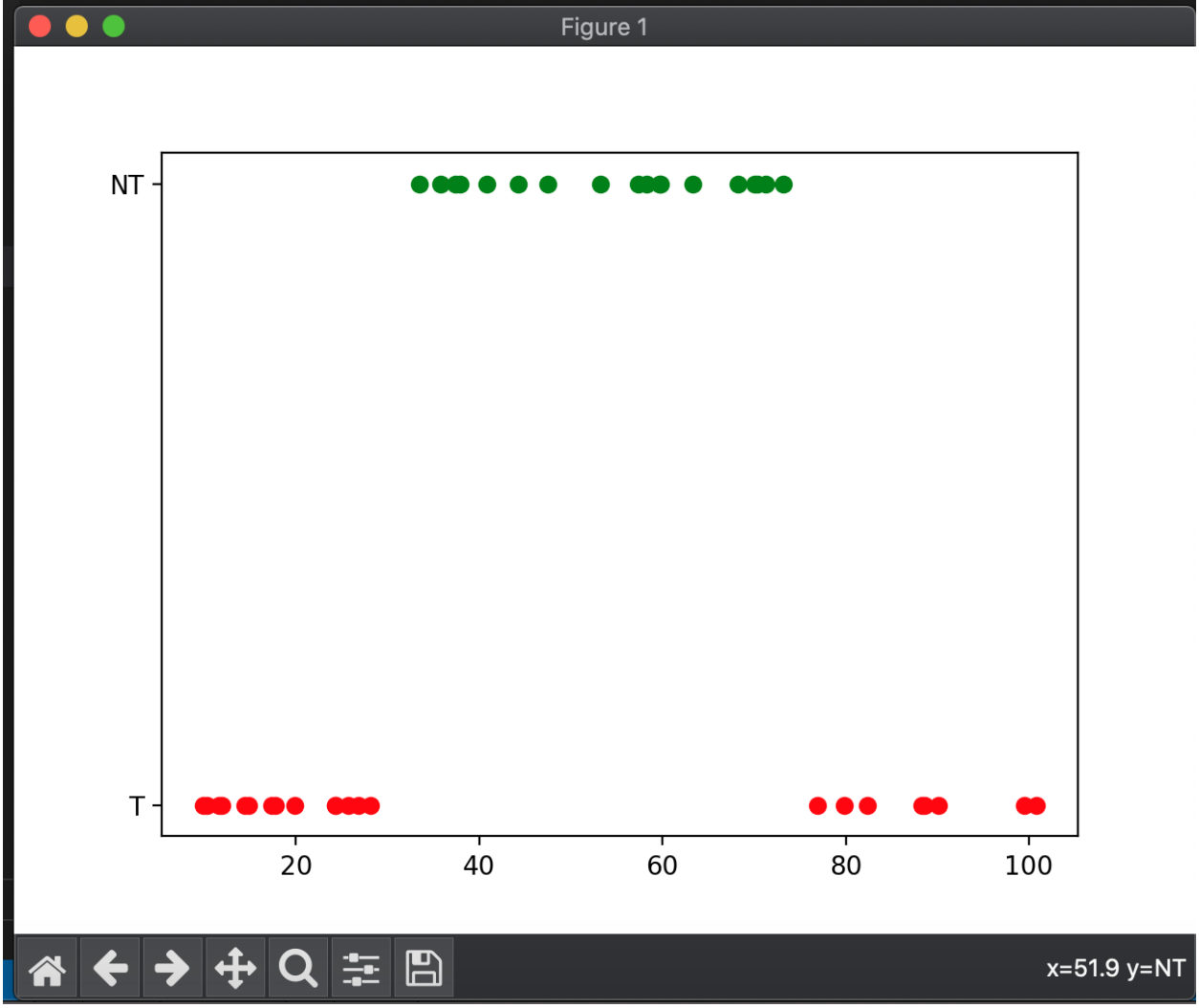
```python
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, ConfusionMatrixDisplay

df = pd.read_csv('speedLimits.csv')
X = np.array(df.Speed)
y = np.array(df.Ticket)
X = X.reshape(-1,1)

X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.1, random_state=0)
modelSVC = SVC(kernel='linear').fit(X_train, y_train)
predL = modelSVC.predict(X_test)
print(f'Accuracy score using linear kernel: {accuracy_score(y_test, predL)}')
modelSVC = SVC(kernel='poly').fit(X_train, y_train)
predP = modelSVC.predict(X_test)
print(f'Accuracy score using poly kernel: {accuracy_score(y_test, predP)}')
modelSVC = SVC(kernel='rbf').fit(X_train, y_train)
predRBF = modelSVC.predict(X_test)
print(f'Accuracy score using rbf kernel: {accuracy_score(y_test, predRBF)}')
modelSVC = SVC(kernel='sigmoid').fit(X_train, y_train)
predS = modelSVC.predict(X_test)
print(f'Accuracy score using sigmoid kernel: {accuracy_score(y_test, predS)}')
print(f'RBF is the best kernel')
for i in range(len(X)):
    if y[i]=="T":
        plt.scatter(X[i],y[i],c='r')
    else:
        plt.scatter(X[i],y[i],c='g')
plt.show()
```

```
/usr/local/bin/python3 "/Users/raulrodriguez/Library/Mobile Documents/com~apple~CloudDoc
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/bin/python3 "/Users/raulr
n/Lab10_1ML.py"
Accuracy score using linear kernel: 0.5
Accuracy score using poly kernel: 0.75
Accuracy score using rbf kernel: 1.0
Accuracy score using sigmoid kernel: 0.5
RBF is the best kernel
```



Lab10_2ML.py > ...

```python
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, ConfusionMatrixDisplay
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
names = ['Sample code number', 'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal Adhesion',
'Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses','Class' ]
df = pd.read_csv('breast-cancer-wisconsin.data.csv', names=names)
df=df.replace('?',np.nan)
df=df.dropna()
df=df.drop("Sample code number",axis=1)
X = np.array(df.loc[:, 'Clump Thickness' : 'Mitoses'])
y = np.array(df['Class'])
X = StandardScaler().fit_transform(X)
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(X)
pc1=principalComponents[0:,1]
pc2=principalComponents[0:,0]
X_train, X_test, y_train, y_test = train_test_split(principalComponents, y,test_size=0.1, random_state=0)
modelSVC = SVC(kernel='linear').fit(X_train, y_train)
y_pred = modelSVC.predict(X_test)
confusionMatrix = confusion_matrix(y_test, y_pred)
print(confusionMatrix)
print(f'Accuracy score: {accuracy_score(y_test, y_pred)}')
for i in range(len(y)):
    if y[i]==2:
        plt.scatter(pc1[i],pc2[i],color='r')
    else:
        plt.scatter(pc1[i],pc2[i],color='g')

plt.xlabel("pc1")
plt.ylabel("pc2")
plt.show()
```

```
/usr/local/bin/python3 "/Users/raulrodriguez/Library/Mobile Documents/com~apple~CloudDocs/D
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/bin/python3 "/Users/raulrodr
n/Lab10_2ML.py"
[[44  2]
 [ 0 23]]
Accuracy score: 0.9710144927536232
```