

```

12 #KNN CLASSIFIER
13 from sklearn.neighbors import KNeighborsClassifier
14 clf = KNeighborsClassifier()
15
16 train_x = images[:10000]
17 train_y = labels[:10000]
18
19 print("Train model")
20 clf.fit(train_x, train_y)
21
22 test_x = images[10000:11000]
23 expected = labels[10000:11000].tolist()
24
25 print("Compute predictions")
26 predicted = clf.predict(test_x)
27
28 print("Accuracy KNN: ", accuracy_score(expected, predicted))
29 print("Confusion Matrix: ")
30 print(confusion_matrix(expected, predicted))
31 print("Classification Report:")
32 print(classification_report(expected, predicted))

```

```

Train model
Compute predictions
Accuracy KNN: 0.956
Confusion Matrix:
[[ 95  0  0  0  0  0  2  0  0  0]
 [  0 111  1  0  0  0  0  0  0  0]
 [  0  1 94  0  1  0  0  1  0  0]
 [  0  0  3 97  0  2  0  1  0  0]
 [  0  1  0  0 98  0  1  0  0  6]
 [  0  0  0  1  0 93  0  0  1  0]
 [  2  0  0  0  0  1 100  0  0  0]
 [  0  0  0  0  0  0  0 96  1  0]
 [  0  3  0  6  1  0  1  1 73  1]
 [  0  0  0  1  2  0  0  2  0 99]]

```

```

Classification Report:
              precision    recall  f1-score   support

0             0.98         0.98         0.98         97
1             0.96         0.99         0.97        112
2             0.96         0.97         0.96         97
3             0.92         0.94         0.93        103
4             0.96         0.92         0.94        106
5             0.97         0.98         0.97         95
6             0.96         0.97         0.97        103
7             0.95         0.99         0.97         97
8             0.97         0.85         0.91         86
9             0.93         0.95         0.94        104

accuracy              0.96        1000
macro avg             0.96         0.95         0.96        1000
weighted avg          0.96         0.96         0.96        1000

```

```

38 #RANDOM FOREST
39 from sklearn.ensemble import RandomForestClassifier
40 clf2 = RandomForestClassifier(n_estimators=100)
41
42 train_x = images[:10000]
43 train_y = labels[:10000]
44
45 print("Train model")
46 clf2.fit(train_x, train_y)
47
48 test_x = images[10000:11000]
49 expected = labels[10000:11000].tolist()
50
51 print("Compute predictions")
52 predicted = clf2.predict(test_x)
53
54 print("Accuracy Random Tree: ", accuracy_score(expected, predicted))
55 print("Confusion Matrix: ")
56 print(confusion_matrix(expected, predicted))
57 print("Classification Report:")
58 print(classification_report(expected, predicted))

```

```

Train model
Compute predictions
Accuracy Random Tree: 0.955
Confusion Matrix:
[[ 94  0  1  0  1  0  1  0  0  0]
 [  0 110  0  1  0  0  1  0  0  0]
 [  0  0 96  0  0  0  0  0  1  0]
 [  0  0  2 97  0  1  1  1  1  0]
 [  1  0  0  0 99  0  1  0  2  3]
 [  1  0  0  1  0 88  1  0  4  0]
 [  2  0  0  0  1  2 98  0  0  0]
 [  0  0  2  0  1  0  0 93  1  0]
 [  0  2  1  2  1  0  0  0 78  2]
 [  0  0  0  1  0  0  0  0  1 102]

```

```

Classification Report:
              precision    recall  f1-score   support

0             0.96         0.97         0.96         97
1             0.98         0.98         0.98        112
2             0.94         0.99         0.96         97
3             0.95         0.94         0.95        103
4             0.96         0.93         0.95        106
5             0.97         0.93         0.95         95
6             0.95         0.95         0.95        103
7             0.99         0.96         0.97         97
8             0.89         0.91         0.90         86
9             0.95         0.98         0.97        104

accuracy              0.95        1000
macro avg             0.95         0.95         0.95        1000
weighted avg          0.96         0.95         0.96        1000

```

```

60 #LINEAR SVC
61 from sklearn.svm import LinearSVC
62 clf3 = LinearSVC()
63
64 train_x = images[:10000]
65 train_y = labels[:10000]
66
67 print("Train model")
68 clf3.fit(train_x, train_y)
69
70 test_x = images[10000:11000]
71 expected = labels[10000:11000].tolist()
72
73 print("Compute predictions")
74 predicted = clf3.predict(test_x)
75
76 print("Accuracy Linear SVC: ", accuracy_score(expected, predicted))
77 print("Confusion Matrix: ")
78 print(confusion_matrix(expected, predicted))
79 print("Classification Report:")
80 print(classification_report(expected, predicted))

```

```

Train model
Compute predictions
Accuracy Linear SVC: 0.879
Confusion Matrix:
[[ 94  0  1  0  0  0  2  0  0  0]
 [  0 105  4  0  0  0  1  0  1  1]
 [  0  0 90  0  0  1  0  0  6  0]
 [  2  0  3 87  0  4  1  2  4  0]
 [  0  0  1  2 90  2  5  1  3  2]
 [  1  0  1  5  1 76  4  0  7  0]
 [  1  0  2  1  0  4 95  0  0  0]
 [  0  0  1  0  2  0  0 86  2  6]
 [  0  6  0  3  1  0  1  0 73  2]
 [  0  1  2  0  5  1  1  6  5 83]]

```

```

Classification Report:
              precision    recall  f1-score   support

0               0.96      0.97      0.96         97
1               0.94      0.94      0.94        112
2               0.86      0.93      0.89         97
3               0.89      0.84      0.87        103
4               0.91      0.85      0.88        106
5               0.86      0.80      0.83         95
6               0.86      0.92      0.89        103
7               0.91      0.89      0.90         97
8               0.72      0.85      0.78         86
9               0.88      0.80      0.84        104

accuracy               0.88      1000
macro avg              0.88      0.88      0.88      1000
weighted avg           0.88      0.88      0.88      1000

```

```

82 #GAUSSIAN NB
83 from sklearn.naive_bayes import GaussianNB
84 clf4 = GaussianNB()
85
86 train_x = images[:10000]
87 train_y = labels[:10000]
88
89 print("Train model")
90 clf4.fit(train_x, train_y)
91
92 test_x = images[10000:11000]
93 expected = labels[10000:11000].tolist()
94
95 print("Compute predictions")
96 predicted = clf4.predict(test_x)
97
98 print("Accuracy Naive Bayes: ", accuracy_score(expected, predicted))
99 print("Confusion Matrix: ")
100 print(confusion_matrix(expected, predicted))
101 print("Classification Report:")
102 print(classification_report(expected, predicted))

```

```

Train model
Compute predictions
Accuracy Naive Bayes: 0.564
Confusion Matrix:
[[ 91  0  1  1  0  0  2  0  1  1]
 [  1 105  0  0  0  1  2  0  2  1]
 [  2  5 45  6  0  0 15  0 24  0]
 [ 10  5  2 30  0  1  5  2 41  7]
 [  1  4  2  0 19  1  4  0 23 52]
 [ 18  4  0  3  0  5  3  0 55  7]
 [  1  3  1  0  0  2 94  0  2  0]
 [  0  2  0  0  0  1  0 27  5 62]
 [  0 18  0  2  0  0  1  1 50 14]
 [  0  3  0  0  2  0  0  1  0 98]]

```

```

Classification Report:
              precision    recall  f1-score   support

0               0.73      0.94      0.82         97
1               0.70      0.94      0.80        112
2               0.88      0.46      0.61         97
3               0.71      0.29      0.41        103
4               0.90      0.18      0.30        106
5               0.45      0.05      0.09         95
6               0.75      0.91      0.82        103
7               0.87      0.28      0.42         97
8               0.25      0.58      0.35         86
9               0.40      0.94      0.57        104

accuracy               0.56      1000
macro avg              0.67      0.56      0.52      1000
weighted avg           0.67      0.56      0.53      1000

```