handwritten-digit-recognition-knn

June 19, 2024

- Made By:

Ashmeet Kaur

102203893

3CO 30 (in Sem - 5)

Results of all the required scenarios (Accuracy, and Confusion Matrix)

[1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
```

Load the data

[3]:
```
data = pd.read_csv('data.csv')
```

Split features and labels

[5]:
```
y=data['label']
x=data.drop('label',axis=1)
```

Defining Parameters

[7]:
```
k_values = [2, 4, 5, 6, 7, 10]
split_ratios = [(60, 40), (70, 30), (75, 25), (80, 20), (90, 10), (95, 5)]
results = []
```

Train the model and predict the values based on the given k values and Train - Test split ratios

[9]:
```
for k in k_values:
    for train_size, test_size in split_ratios:
        # Train-test split
        x_train, x_test, y_train, y_test = train_test_split(x, y,␣
↪test_size=test_size/100, random_state=42)

        # Train the model
        classifier = KNeighborsClassifier(n_neighbors=k)
        classifier.fit(x_train, y_train)
```

```
        # Predict
        y_pred = classifier.predict(x_test)

        # Evaluate
        accuracy = accuracy_score(y_test, y_pred)
        cm = confusion_matrix(y_test, y_pred)

        # Append results
        results.append((k, train_size, test_size, accuracy, cm))

        # Print results
        print(f"k: {k}, Train Size: {train_size}%, Test Size: {test_size}%")
        print(f"Accuracy: {accuracy}")
        print(f"Confusion Matrix:\n{cm}")
        print("\n")
```

```
k: 2, Train Size: 60%, Test Size: 40%
Accuracy: 0.9567261904761905
Confusion Matrix:
[[1621    0    1    0    0    0    4    0    0    0]
 [   0 1847    2    0    1    0    0    1    0    0]
 [  17   27 1627    3    2    2    0   12    3    1]
 [   2    9   17 1711    0   15    1    8    9    4]
 [   3   19    0    0 1587    0    3    3    0   18]
 [   5    5    1   56    5 1375   12    0    3    5]
 [  24    5    1    1    4    9 1650    0    0    0]
 [   1   27   12    0    6    1    0 1714    0   12]
 [  14   22   19   55    9   45    8    7 1450   10]
 [  11    8    2   19   50    6    1   55    4 1491]]


k: 2, Train Size: 70%, Test Size: 30%
Accuracy: 0.9600793650793651
Confusion Matrix:
[[1197    0    0    0    0    0    3    0    0    0]
 [   0 1387    1    0    1    0    0    0    0    0]
 [  15   20 1240    3    1    1    1   10    3    0]
 [   1    6   15 1305    0    9    1    6    9    3]
 [   2   12    0    0 1190    0    4    3    0   11]
 [   1    3    0   36    4 1031    5    0    2    3]
 [  13    2    1    1    4    7 1228    0    0    0]
 [   0   24    9    0    3    1    0 1311    0   11]
 [   5   13   14   35    3   35    6    5 1085    8]
 [   8    4    1   16   36    3    0   38    2 1123]]
```

```
k: 2, Train Size: 75%, Test Size: 25%
Accuracy: 0.961047619047619
Confusion Matrix:
[[1023    0    0    0    0    1    1    0    0    0]
 [   0 1145    0    0    1    0    0    0    0    0]
 [  13   19 1027    2    1    1    1    7    1    0]
 [   0    6    7 1114    0    7    1    6    7    3]
 [   2    8    0    0 1000    0    4    1    0    9]
 [   1    3    0   29    3  853    5    0    2    2]
 [  11    2    0    2    1    6  988    0    0    0]
 [   0   22    8    0    2    0    0 1095    0    8]
 [   4   15   13   27    3   31    4    2  900    6]
 [   6    3    0   14   32    2    0   29    2  946]]


k: 2, Train Size: 80%, Test Size: 20%
Accuracy: 0.9621428571428572
Confusion Matrix:
[[814   0   0   0   0   1   1   0   0   0]
 [  0 908   0   0   1   0   0   0   0   0]
 [ 10  16 809   1   1   1   1   7   0   0]
 [  0   4   7 907   0   7   0   6   4   2]
 [  1   6   0   0 822   0   2   1   0   7]
 [  0   2   0  21   1 670   5   0   1   2]
 [  8   0   0   0   0   4 773   0   0   0]
 [  0  15   6   0   3   0   0 863   0   6]
 [  3  13  13  21   2  27   3   2 745   6]
 [  4   3   0  10  19   2   0  27   2 771]]


k: 2, Train Size: 90%, Test Size: 10%
Accuracy: 0.9633333333333334
Confusion Matrix:
[[407   0   0   0   0   0   1   0   0   0]
 [  0 471   0   0   0   0   0   0   0   0]
 [  4   6 405   0   1   0   1   3   0   0]
 [  0   2   5 491   0   2   0   3   1   2]
 [  0   1   0   0 391   0   1   0   0   4]
 [  0   0   0   9   0 328   1   0   1   0]
 [  7   0   0   0   0   3 392   0   0   0]
 [  0  10   2   0   2   0   0 422   0   2]
 [  2   7  10  12   2  12   3   2 350   3]
 [  3   1   0   7   5   0   0  10   1 389]]


k: 2, Train Size: 95%, Test Size: 5%
Accuracy: 0.9676190476190476
Confusion Matrix:
```

```
[[215   0   0   0   0   0   1   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   6 212   0   1   0   1   1   0   0]
 [  0   1   1 255   0   0   0   2   0   1]
 [  0   1   0   0 192   0   0   0   0   3]
 [  0   0   0   3   0 156   0   0   1   0]
 [  3   0   0   0   0   1 195   0   0   0]
 [  0   5   0   0   2   0   0 221   0   2]
 [  1   3   6   5   1   6   1   0 168   0]
 [  1   0   0   2   2   0   0   3   0 184]]
```

```
k: 4, Train Size: 60%, Test Size: 40%
Accuracy: 0.9618452380952381
Confusion Matrix:
[[1618    1    0    0    0    0    6    0    0    1]
 [   0 1847    2    0    0    0    1    1    0    0]
 [  13   23 1612    4    2    1    4   31    3    1]
 [   3    8    9 1709    0   18    2   10    9    8]
 [   2   18    0    0 1585    0    5    2    0   21]
 [   3    5    1   47    2 1382   13    0    5    9]
 [  20    4    1    0    3    3 1660    0    3    0]
 [   1   30    6    0    5    0    0 1712    0   19]
 [  11   20   11   33    7   33    8    3 1496   17]
 [   8    7    3   16   23    7    2   40    3 1538]]
```

```
k: 4, Train Size: 70%, Test Size: 30%
Accuracy: 0.9631746031746031
Confusion Matrix:
[[1196    0    0    0    0    0    4    0    0    0]
 [   0 1386    1    0    0    0    2    0    0    0]
 [  13   16 1229    4    3    1    2   22    3    1]
 [   1    6   10 1300    0   12    2    9    9    6]
 [   2   12    0    0 1185    0    4    2    0   17]
 [   1    2    0   31    0 1037    8    0    1    5]
 [  13    3    1    0    3    6 1230    0    0    0]
 [   0   27    3    0    3    0    0 1311    0   15]
 [   5   13    7   21    3   28    5    2 1115   10]
 [   8    3    2   13   24    1    1   31    1 1147]]
```

```
k: 4, Train Size: 75%, Test Size: 25%
Accuracy: 0.9648571428571429
Confusion Matrix:
[[1023    0    0    0    0    0    2    0    0    0]
 [   0 1144    0    0    0    0    2    0    0    0]
 [  10   15 1021    3    3    1    2   15    1    1]
```

```
 [   1    7    3 1110    0    8    2    7    8    5]
 [   2    8    0    0  996    0    3    1    0   14]
 [   1    2    0   21    0  861    8    0    1    4]
 [  11    3    0    0    1    7  988    0    0    0]
 [   0   26    3    0    1    0    0 1093    0   12]
 [   4   14    5   19    2   25    3    1  925    7]
 [   4    3    2   12   18    2    1   21    1  970]]


k: 4, Train Size: 80%, Test Size: 20%
Accuracy: 0.9644047619047619
Confusion Matrix:
[[813   0   0   0   0   0   3   0   0   0]
 [  0 907   0   0   0   0   2   0   0   0]
 [  6  14 805   3   2   1   1  13   0   1]
 [  1   6   3 901   0   8   1   6   7   4]
 [  1   7   0   0 816   0   3   1   0  11]
 [  0   1   0  18   0 671   8   0   1   3]
 [  8   1   0   0   0   5 771   0   0   0]
 [  0  18   3   0   1   0   0 862   0   9]
 [  1  12   5  14   2  23   4   1 766   7]
 [  2   3   3   9  12   1   0  18   1 789]]


k: 4, Train Size: 90%, Test Size: 10%
Accuracy: 0.9652380952380952
Confusion Matrix:
[[408   0   0   0   0   0   0   0   0   0]
 [  0 470   0   0   0   0   1   0   0   0]
 [  3   7 403   1   1   0   0   5   0   0]
 [  1   3   3 486   0   4   0   4   3   2]
 [  0   1   0   0 388   0   1   0   0   7]
 [  0   0   0   6   0 327   4   0   1   1]
 [  6   0   0   0   0   2 394   0   0   0]
 [  0  14   2   0   0   0   0 419   0   3]
 [  1   6   4   7   1  11   3   1 365   4]
 [  1   1   1   6   5   0   0   8   0 394]]


k: 4, Train Size: 95%, Test Size: 5%
Accuracy: 0.9671428571428572
Confusion Matrix:
[[216   0   0   0   0   0   0   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   7 210   0   1   0   0   3   0   0]
 [  0   1   1 254   0   0   0   2   1   1]
 [  0   1   0   0 192   0   0   0   0   3]
 [  0   0   0   3   0 154   1   0   1   1]
```

```
 [  4   0   0   0   0   0 195   0   0   0]
 [  0   8   0   0   0   0   0 219   0   3]
 [  1   3   4   3   0   6   1   0 173   0]
 [  0   0   1   2   2   0   0   3   0 184]]


k: 5, Train Size: 60%, Test Size: 40%
Accuracy: 0.9619642857142857
Confusion Matrix:
[[1616    0    0    0    0    1    8    0    0    1]
 [   0 1845    2    0    0    0    3    1    0    0]
 [  12   25 1604    4    2    2    3   34    6    2]
 [   4    7    9 1701    0   23    1   12    9   10]
 [   3   18    0    0 1570    0    6    3    0   33]
 [   4    7    1   35    5 1380   19    1    5   10]
 [  14    4    1    0    2    3 1668    0    2    0]
 [   1   30    4    0    3    0    0 1712    0   23]
 [   9   20    6   27   10   28    7    3 1511   18]
 [   7    9    2   17   14    5    2   31    6 1554]]


k: 5, Train Size: 70%, Test Size: 30%
Accuracy: 0.9655555555555555
Confusion Matrix:
[[1196    0    0    0    0    0    4    0    0    0]
 [   0 1384    1    0    0    0    4    0    0    0]
 [  10   19 1223    4    2    2    2   28    4    0]
 [   3    5    9 1299    0   14    2    8    7    8]
 [   2   11    0    0 1181    0    4    2    0   22]
 [   1    3    0   30    1 1034   11    0    1    4]
 [   9    3    1    0    1    2 1239    0    1    0]
 [   0   26    2    0    2    0    0 1311    0   18]
 [   2   14    3   19    3   20    5    2 1127   14]
 [   8    3    1   13   13    1    0   18    2 1172]]


k: 5, Train Size: 75%, Test Size: 25%
Accuracy: 0.9666666666666667
Confusion Matrix:
[[1022    0    0    0    0    0    3    0    0    0]
 [   0 1143    0    0    0    0    3    0    0    0]
 [   9   17 1014    3    2    2    2   22    1    0]
 [   2    5    4 1110    0   10    2    6    5    7]
 [   2    7    0    0  993    0    3    1    0   18]
 [   2    2    0   19    1  861    9    0    1    3]
 [   8    2    0    0    0    2  997    0    1    0]
 [   0   26    2    0    2    0    0 1088    0   17]
 [   1   14    2   18    3   18    4    1  936    8]
```

```
 [   6    3    1   12   10    0    0   14    2  986]]


k: 5, Train Size: 80%, Test Size: 20%
Accuracy: 0.9648809523809524
Confusion Matrix:
[[812   0   0   0   0   0   4   0   0   0]
 [  0 906   0   0   0   0   3   0   0   0]
 [  6  15 796   3   2   2   1  20   1   0]
 [  2   4   4 900   0   9   1   6   5   6]
 [  1   5   0   0 811   0   3   1   0  18]
 [  1   2   0  15   1 669   9   0   1   4]
 [  5   0   0   0   0   3 777   0   0   0]
 [  0  18   2   0   1   0   0 857   0  15]
 [  1  12   2  12   3  15   4   1 777   8]
 [  3   3   1  10   6   1   0  12   2 800]]


k: 5, Train Size: 90%, Test Size: 10%
Accuracy: 0.9654761904761905
Confusion Matrix:
[[408   0   0   0   0   0   0   0   0   0]
 [  0 469   0   0   0   0   2   0   0   0]
 [  3   6 402   0   2   0   0   7   0   0]
 [  1   2   4 485   0   5   0   3   3   3]
 [  0   1   0   0 383   0   2   0   0  11]
 [  0   0   0   6   0 325   5   0   1   2]
 [  4   0   0   0   0   2 396   0   0   0]
 [  0  14   1   0   1   0   0 416   0   6]
 [  1   6   1   7   2   7   3   1 372   3]
 [  1   1   1   6   2   0   0   6   0 399]]


k: 5, Train Size: 95%, Test Size: 5%
Accuracy: 0.9671428571428572
Confusion Matrix:
[[216   0   0   0   0   0   0   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   6 210   0   2   0   0   3   0   0]
 [  0   1   1 253   0   0   0   2   1   2]
 [  0   1   0   0 190   0   0   0   0   5]
 [  0   0   0   3   0 153   1   0   1   2]
 [  3   0   0   0   0   1 195   0   0   0]
 [  0   7   0   0   1   0   0 218   0   4]
 [  1   3   1   2   1   5   1   0 177   0]
 [  0   0   1   3   1   0   0   2   0 185]]
```

```
k: 6, Train Size: 60%, Test Size: 40%
Accuracy: 0.9598214285714286
Confusion Matrix:
[[1618    0    0    0    0    1    6    0    0    1]
 [   0 1845    3    0    0    0    2    1    0    0]
 [  12   27 1603    5    2    1    3   32    8    1]
 [   4    8    8 1705    0   20    2   12    9    8]
 [   2   19    0    0 1571    0    8    2    0   31]
 [   5    9    1   36    4 1379   18    0    4   11]
 [  21    5    1    0    2    2 1660    0    3    0]
 [   1   32    5    0    3    0    0 1711    0   21]
 [   7   26    6   35   13   33    7    2 1493   17]
 [   9    9    2   17   19    6    2   38    5 1540]]


k: 6, Train Size: 70%, Test Size: 30%
Accuracy: 0.9625396825396826
Confusion Matrix:
[[1197    0    0    0    0    0    3    0    0    0]
 [   0 1384    1    0    0    0    4    0    0    0]
 [  13   19 1225    4    2    1    2   26    2    0]
 [   3    7   11 1296    0   13    1    9    8    7]
 [   2   12    0    0 1184    0    4    2    0   18]
 [   2    4    0   27    0 1035    9    0    2    6]
 [  15    3    1    0    1    5 1230    0    1    0]
 [   0   29    3    0    2    0    0 1310    0   15]
 [   2   19    3   22    7   24    6    2 1113   11]
 [   8    3    1   14   19    1    0   29    2 1154]]


k: 6, Train Size: 75%, Test Size: 25%
Accuracy: 0.9646666666666667
Confusion Matrix:
[[1022    0    0    0    0    0    3    0    0    0]
 [   0 1143    0    0    0    0    3    0    0    0]
 [  12   16 1016    3    1    1    2   20    1    0]
 [   2    5    6 1108    0    9    2    6    6    7]
 [   2    8    0    0  996    0    3    1    0   14]
 [   1    3    0   18    0  862    7    0    2    5]
 [  14    4    0    0    0    3  988    0    1    0]
 [   0   27    3    0    2    0    0 1092    0   11]
 [   1   20    3   19    4   21    4    1  926    6]
 [   6    3    1   11   14    1    0   20    2  976]]


k: 6, Train Size: 80%, Test Size: 20%
Accuracy: 0.9636904761904762
Confusion Matrix:
```

```
[[812   0   0   0   0   0   4   0   0   0]
 [  0 906   0   0   0   0   3   0   0   0]
 [  7  13 802   3   1   1   1  17   1   0]
 [  3   4   5 898   0   9   1   5   6   6]
 [  1   6   0   0 815   0   3   1   0  13]
 [  0   3   0  14   0 672   7   0   2   4]
 [  8   2   0   0   0   4 771   0   0   0]
 [  0  20   3   0   1   0   0 859   0  10]
 [  1  14   2  15   3  21   3   1 770   5]
 [  3   3   1   9  11   2   0  18   1 790]]
```

```
k: 6, Train Size: 90%, Test Size: 10%
Accuracy: 0.9661904761904762
Confusion Matrix:
[[408   0   0   0   0   0   0   0   0   0]
 [  0 469   0   0   0   0   2   0   0   0]
 [  3   6 403   0   1   0   0   7   0   0]
 [  1   2   4 487   0   4   0   3   3   2]
 [  0   1   0   0 388   0   2   0   0   6]
 [  0   0   0   7   0 327   3   0   1   1]
 [  5   1   0   0   0   2 394   0   0   0]
 [  0  14   1   0   0   0   0 418   0   5]
 [  1   7   1   6   2  11   3   1 368   3]
 [  2   1   1   6   3   0   0   7   0 396]]
```

```
k: 6, Train Size: 95%, Test Size: 5%
Accuracy: 0.9685714285714285
Confusion Matrix:
[[216   0   0   0   0   0   0   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   6 211   0   1   0   0   3   0   0]
 [  0   1   1 254   0   0   0   2   1   1]
 [  0   1   0   0 191   0   0   0   0   4]
 [  0   0   0   3   0 154   1   0   1   1]
 [  3   1   0   0   0   1 194   0   0   0]
 [  0   8   0   0   0   0   0 219   0   3]
 [  1   3   1   3   1   5   1   0 176   0]
 [  1   0   1   2   1   0   0   2   0 185]]
```

```
k: 7, Train Size: 60%, Test Size: 40%
Accuracy: 0.9602380952380952
Confusion Matrix:
[[1617    1    0    0    0    1    6    0    0    1]
 [   0 1845    2    0    0    0    3    1    0    0]
 [  11   29 1595    6    2    1    4   37    7    2]
```

```
 [   4    8    8 1695    1   25    2   14    9   10]
 [   3   19    0    0 1563    0    8    2    0   38]
 [   6    7    1   30    4 1382   21    2    2   12]
 [  14    4    0    0    2    4 1668    0    2    0]
 [   1   32    3    0    2    0    0 1708    0   27]
 [   8   23    6   29   12   28    8    2 1502   21]
 [   8   10    2   15   13    7    2   27    6 1557]]


k: 7, Train Size: 70%, Test Size: 30%
Accuracy: 0.9633333333333334
Confusion Matrix:
[[1196    0    0    0    0    0    4    0    0    0]
 [   0 1384    1    0    0    0    4    0    0    0]
 [  10   19 1218    6    2    1    2   31    4    1]
 [   4    7    9 1292    0   14    1   11    8    9]
 [   2   12    0    0 1175    0    5    2    0   26]
 [   2    2    0   24    1 1033   12    1    2    8]
 [  10    3    1    0    1    4 1236    0    1    0]
 [   0   28    3    0    2    0    0 1311    0   15]
 [   1   18    4   19    5   20    5    1 1122   14]
 [   8    3    2   13   11    1    0   20    2 1171]]


k: 7, Train Size: 75%, Test Size: 25%
Accuracy: 0.9642857142857143
Confusion Matrix:
[[1023    0    0    0    0    0    2    0    0    0]
 [   0 1143    0    0    0    0    3    0    0    0]
 [   9   16 1011    5    1    1    2   23    3    1]
 [   3    5    5 1103    0   11    2    7    6    9]
 [   2    8    0    0  986    0    4    1    0   23]
 [   1    2    0   16    1  860   11    0    2    5]
 [   9    3    0    0    0    3  994    0    1    0]
 [   0   26    3    0    2    0    0 1091    0   13]
 [   1   19    4   18    4   18    4    1  927    9]
 [   6    3    1   11   10    1    0   13    2  987]]


k: 7, Train Size: 80%, Test Size: 20%
Accuracy: 0.9632142857142857
Confusion Matrix:
[[814   0   0   0   0   0   2   0   0   0]
 [  0 906   0   0   0   0   3   0   0   0]
 [  7  14 796   4   1   1   1  19   2   1]
 [  2   4   5 895   0  10   1   6   6   8]
 [  1   6   0   0 806   0   4   1   0  21]
 [  0   2   0  14   0 670  10   0   2   4]
```

```
 [  5   1   0   0   0   4 775   0   0   0]
 [  0  20   3   0   1   0   0 858   0  11]
 [  1  14   3  14   4  17   3   1 771   7]
 [  3   3   1   9   8   1   0  12   1 800]]


k: 7, Train Size: 90%, Test Size: 10%
Accuracy: 0.9647619047619047
Confusion Matrix:
[[408   0   0   0   0   0   0   0   0   0]
 [  0 469   0   0   0   0   2   0   0   0]
 [  3   7 398   1   1   0   0   9   1   0]
 [  1   2   3 487   0   4   0   3   3   3]
 [  0   1   0   0 380   0   2   0   0  14]
 [  0   0   0   5   0 326   6   0   1   1]
 [  3   0   0   0   0   2 397   0   0   0]
 [  0  14   1   0   0   0   0 417   0   6]
 [  1   7   2   6   2   8   3   1 370   3]
 [  2   1   1   5   1   0   0   6   0 400]]


k: 7, Train Size: 95%, Test Size: 5%
Accuracy: 0.9671428571428572
Confusion Matrix:
[[216   0   0   0   0   0   0   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   6 209   0   1   0   0   4   1   0]
 [  0   1   1 253   0   0   0   2   1   2]
 [  0   1   0   0 190   0   0   0   0   5]
 [  0   0   0   2   0 155   1   0   1   1]
 [  2   0   0   0   0   1 196   0   0   0]
 [  0   8   0   0   0   0   0 219   0   3]
 [  1   3   2   3   1   6   1   0 174   0]
 [  1   0   1   2   1   0   0   2   0 185]]


k: 10, Train Size: 60%, Test Size: 40%
Accuracy: 0.9576785714285714
Confusion Matrix:
[[1616    1    0    0    0    1    7    0    0    1]
 [   0 1846    2    0    0    0    3    0    0    0]
 [  13   35 1579    6    4    2    6   35    9    5]
 [   4   10    7 1697    1   21    1   15   10   10]
 [   2   21    0    0 1565    0    9    2    0   34]
 [   4   10    0   31    6 1381   20    2    1   12]
 [  17    5    1    0    4    3 1661    0    3    0]
 [   1   36    3    0    4    0    0 1706    0   23]
 [   7   26    6   31    9   32    9    3 1493   23]
```

```
 [   9   11    4   18   13    5    3   35    4 1545]]


k: 10, Train Size: 70%, Test Size: 30%
Accuracy: 0.9608730158730159
Confusion Matrix:
[[1196    0    0    0    0    0    4    0    0    0]
 [   0 1384    1    0    0    0    4    0    0    0]
 [  13   24 1211    5    3    2    4   24    7    1]
 [   4    8    7 1295    0   13    1   12    8    7]
 [   2   14    0    0 1174    0    6    2    0   24]
 [   2    4    0   26    2 1028   13    1    1    8]
 [  10    3    1    0    2    2 1236    0    2    0]
 [   0   30    2    0    3    0    0 1306    0   18]
 [   1   20    4   20    7   24    7    1 1110   15]
 [   8    4    4   13    9    0    1   23    2 1167]]


k: 10, Train Size: 75%, Test Size: 25%
Accuracy: 0.962
Confusion Matrix:
[[1022    0    0    0    0    0    3    0    0    0]
 [   0 1142    0    0    1    0    3    0    0    0]
 [  11   22 1001    4    2    2    4   20    5    1]
 [   3    7    4 1106    0    8    1    9    6    7]
 [   2   10    0    0  988    0    5    2    0   17]
 [   2    3    0   18    2  853   12    1    1    6]
 [  11    3    0    0    0    2  993    0    1    0]
 [   0   28    2    0    2    0    0 1091    0   12]
 [   1   19    3   17    6   19    5    1  924   10]
 [   5    4    3   11    7    2    1   18    2  981]]


k: 10, Train Size: 80%, Test Size: 20%
Accuracy: 0.9616666666666667
Confusion Matrix:
[[814   0   0   0   0   0   2   0   0   0]
 [  0 905   0   0   1   0   3   0   0   0]
 [  7  17 793   4   2   1   3  15   3   1]
 [  3   5   4 897   0   7   1   8   6   6]
 [  1   7   0   0 809   0   3   2   0  17]
 [  1   1   0  15   0 668   9   1   2   5]
 [  7   1   0   0   0   3 774   0   0   0]
 [  0  22   2   0   1   0   0 859   0   9]
 [  1  15   3  13   6  20   5   1 762   9]
 [  3   4   3   8   6   1   0  14   2 797]]
```

```
k: 10, Train Size: 90%, Test Size: 10%
Accuracy: 0.960952380952381
Confusion Matrix:
[[406   0   0   0   0   0   2   0   0   0]
 [  0 469   0   0   0   0   2   0   0   0]
 [  3   9 396   0   2   0   0   9   1   0]
 [  1   3   3 485   1   4   0   4   2   3]
 [  0   1   0   0 383   0   2   1   0  10]
 [  1   0   0   7   0 326   4   0   0   1]
 [  5   0   0   0   0   2 394   0   1   0]
 [  0  15   1   0   0   0   0 419   0   3]
 [  1   7   1  10   5   8   4   1 363   3]
 [  2   2   1   5   2   0   0   8   1 395]]


k: 10, Train Size: 95%, Test Size: 5%
Accuracy: 0.9642857142857143
Confusion Matrix:
[[214   0   0   0   0   0   2   0   0   0]
 [  0 234   0   0   0   0   0   0   0   0]
 [  1   7 209   0   1   0   0   3   1   0]
 [  0   1   1 253   0   0   0   2   1   2]
 [  0   1   0   0 190   0   0   0   0   5]
 [  0   0   0   4   0 153   1   0   0   2]
 [  3   0   0   0   0   2 194   0   0   0]
 [  0   8   0   0   0   0   0 220   0   2]
 [  1   4   0   3   1   6   1   0 175   0]
 [  1   1   1   2   1   0   0   3   0 183]]
```

Save results to a DataFrame

[10]:
```
results_df = pd.DataFrame(results, columns=['k', 'Train Size', 'Test Size',␣
↪'Accuracy', 'Confusion Matrix'])
```

Save results to CSV

[11]:
```
results_df.to_csv('knn_results.csv', index=False)
```

Analysis regarding the dependency of the performance of model over training-testing split and k values

[12]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the results from CSV
results_df = pd.read_csv('knn_results.csv')
```

```python
# Create a Summary Table
summary_table = results_df.groupby(['k', 'Train Size', 'Test Size']).agg({
    'Accuracy': ['mean', 'std']
}).reset_index()
print(summary_table)

# Plot Accuracy vs. Train Size for different k values
plt.figure(figsize=(9, 6))
for k in results_df['k'].unique():
    subset = results_df[results_df['k'] == k]
    plt.plot(subset['Train Size'], subset['Accuracy'], label=f'k={k}')

plt.xlabel('Train Size (%)')
plt.ylabel('Accuracy')
plt.title('Accuracy vs. Train Size for different k values')
plt.legend()
plt.grid(True)
plt.savefig('accuracy_vs_train_size.png')
plt.show()
```

```
     k Train Size Test Size  Accuracy
                                 mean std
0    2         60        40  0.956726 NaN
1    2         70        30  0.960079 NaN
2    2         75        25  0.961048 NaN
3    2         80        20  0.962143 NaN
4    2         90        10  0.963333 NaN
5    2         95         5  0.967619 NaN
6    4         60        40  0.961845 NaN
7    4         70        30  0.963175 NaN
8    4         75        25  0.964857 NaN
9    4         80        20  0.964405 NaN
10   4         90        10  0.965238 NaN
11   4         95         5  0.967143 NaN
12   5         60        40  0.961964 NaN
13   5         70        30  0.965556 NaN
14   5         75        25  0.966667 NaN
15   5         80        20  0.964881 NaN
16   5         90        10  0.965476 NaN
17   5         95         5  0.967143 NaN
18   6         60        40  0.959821 NaN
19   6         70        30  0.962540 NaN
20   6         75        25  0.964667 NaN
21   6         80        20  0.963690 NaN
22   6         90        10  0.966190 NaN
23   6         95         5  0.968571 NaN
```

24	7	60	40	0.960238	NaN
25	7	70	30	0.963333	NaN
26	7	75	25	0.964286	NaN
27	7	80	20	0.963214	NaN
28	7	90	10	0.964762	NaN
29	7	95	5	0.967143	NaN
30	10	60	40	0.957679	NaN
31	10	70	30	0.960873	NaN
32	10	75	25	0.962000	NaN
33	10	80	20	0.961667	NaN
34	10	90	10	0.960952	NaN
35	10	95	5	0.964286	NaN