Homework 7

Shivangi Mundhra | SUID – 842548148 | 10/30/2022

SECTION 1 - INTRODUCTION

In this homework, I am going to try to recognize digits 0 to 9 in handwriting images using a k-Nearest Neighbors model, Support Vector Machine model and a Random Forest model. I am going to walk through all the aspects of this assignment including what the assignment is and what approaches I am using. In the analysis itself, I will try to explain all reasoning and thought process behind each significant step. The intent is that any person, especially one without technical knowledge and that is not familiar with this problem statement, should be able to read this report and follow along.

About the assignment –

The task at hand, in this assignment, is to be able to recognize the handwritten numbers 0 – 9, using 3 classification machine learning algorithms –

1. The k-Nearest Neighbors algorithm,
2. The Support Vector Machine algorithm, and,
3. The Random Forest algorithm

The data set consists of a training data set and a test data set, both these data sets consist of a "label" which tells us what number the row's data represents and 784 columns of data/features that describe that digit. We will use the training data set to train these models. After we create each model, we will examine the model's accuracy by predicting the labels on test data set. We will report on each of the model's accuracy and find which model best predicts the digits in test data.

About the k-Nearest Neighbor algorithm –

The kNN model is a supervised machine learning algorithm that assumes that similar things are located near to each other. The idea is to create a model that predicts the value of a target variable (in this case, label) by looking at its nearest labels. In this assignment, we want to create a model that predicts the label of handwritten images by analyzing the accompanying data for those labels. The kNN algorithm simply plots labels based on which it bases it decisions for classifying/predicting labels.

About the Support Vector Machine algorithm –

The SVM classification algorithm is a supervised machine learning classifier for two-group classification problems. It creates a divider/hyperplane that separates two groups of labels/targets. Since, this is a two-group classifier, it creates multiple hyperplanes between multiple target pairs for a model where we have more than 2 distinct values in target.

About the Random Forest algorithm –

Shivangi Mundhra
SUID - 842548148

Random Forest is a supervised machine learning classifier that works by constructing a multitude of decision trees while training the data. The output of the model is the label that is selected by most of these trees. Because Random Forests use multiple decision trees, their accuracy is usually better than a decision tree model.

Analysis and Data Pre-processing –

I used scikit-learn machine learning libraries to create these models in python.

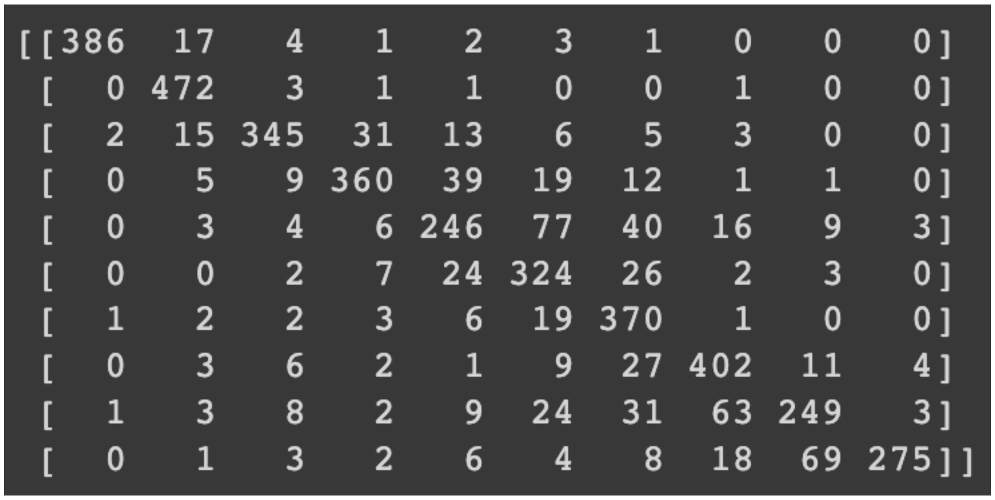
1. I start with importing different packages like sklearn, pandas, numpy, etc. Some of these, like sklearn, help us with creation and implementation of the models, some, like matplotlib and graphviz help in creating visualizations and others just help in data manipulation.
2. I store the contents of the data from digit-train.csv into a train dataframe and the data from digit-test.csv into a test data frame.
3. I split the training and testing dataset in features (x) and labels (y) data. The features data represents the columns that we want to train our data on and use to predict labels while label (y) is what we want to predict.

Evaluation Method –

For evaluating these models, we will compare their accuracy scores. Accuracy score represents the percent of correct prediction on test data when compared to the actual labels in test data. We will also look at their confusion matrix whenever possible.

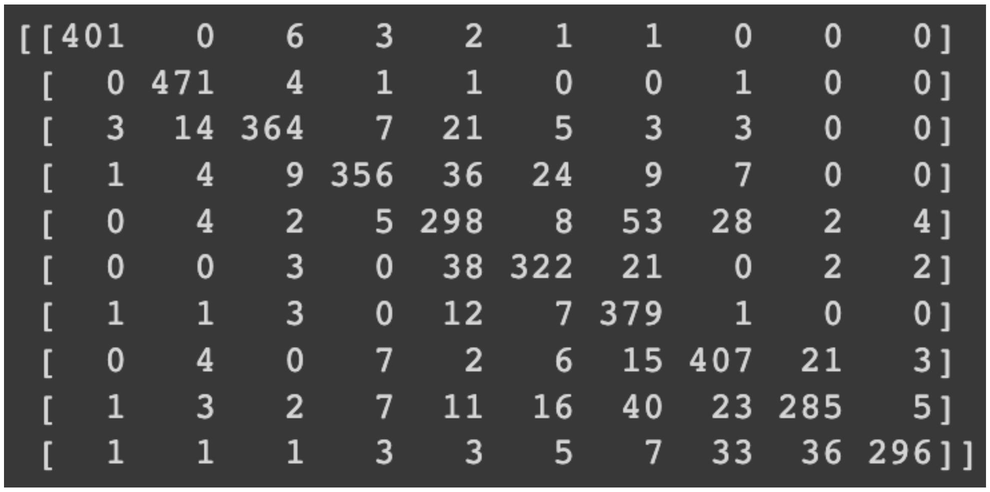
SECTION 2 - kNN

1. We create a base kNN model with default settings, using the training data. We use this model to make predictions on the test data set and calculate its accuracy.
2. After testing the base kNN model against the test data set, we receive an accuracy score of 88.59%. Below is the confusion matrix.



```
[[386  17   4   1   2   3   1   0   0   0]
 [  0 472   3   1   1   0   0   1   0   0]
 [  2  15 345  31  13   6   5   3   0   0]
 [  0   5   9 360  39  19  12   1   1   0]
 [  0   3   4   6 246  77  40  16   9   3]
 [  0   0   2   7  24 324  26   2   3   0]
 [  1   2   2   3   6  19 370   1   0   0]
 [  0   3   6   2   1   9  27 402  11   4]
 [  1   3   8   2   9  24  31  63 249   3]
 [  0   1   3   2   6   4   8  18  69 275]]
```

3. To tune the model, we repeat the process of creating model with different values of neighbors between 1 and 50 and test their accuracy. We find that n_neighbors = 3 should yield the highest performance.
4. We create a model with n_neighbors = 3 and test its accuracy. The accuracy of this model is 89.44% which is only slightly better than the base kNN model. Below is the confusion matrix.



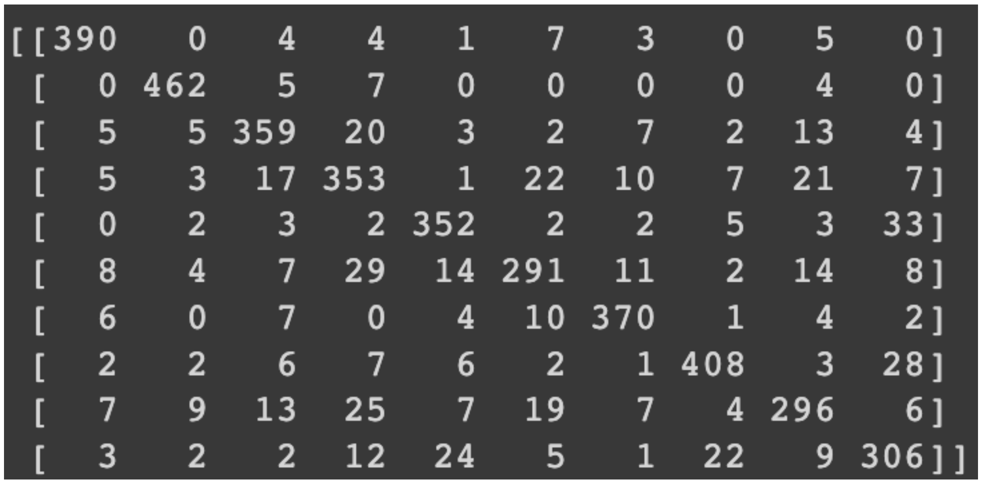
```
[[401   0   6   3   2   1   1   0   0   0]
 [  0 471   4   1   1   0   0   1   0   0]
 [  3  14 364   7  21   5   3   3   0   0]
 [  1   4   9 356  36  24   9   7   0   0]
 [  0   4   2   5 298   8  53  28   2   4]
 [  0   0   3   0  38 322  21   0   2   2]
 [  1   1   3   0  12   7 379   1   0   0]
 [  0   4   0   7   2   6  15 407  21   3]
 [  1   3   2   7  11  16  40  23 285   5]
 [  1   1   1   3   3   5   7  33  36 296]]
```

SECTION 3 – SVM

In SVM, we apply the linearSVM and SVM using linear, polynomial and rbf kernels to our data and create separate models.

LinearSVC –

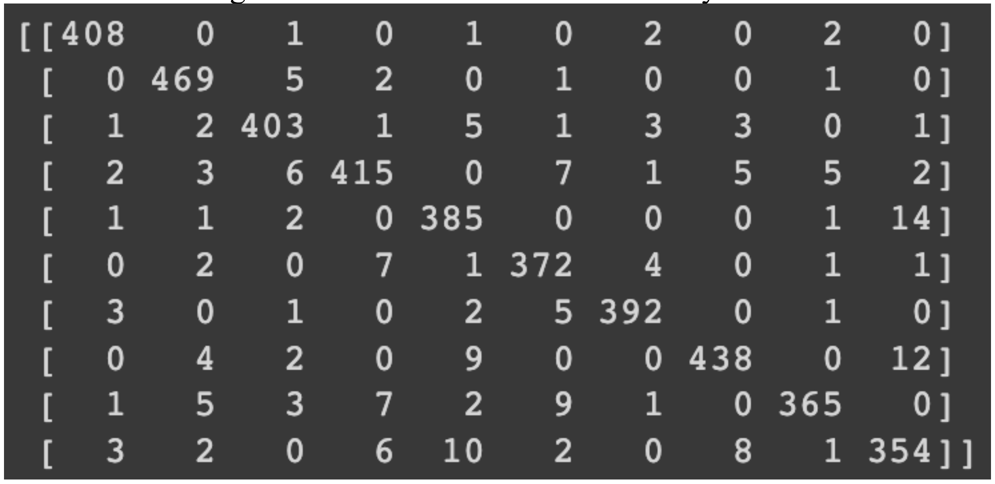
1. We create a base LinearSVC model with default settings, using the training data. We use this model to make predictions on the test data set and calculate its accuracy.
2. After testing the model against the test data set, we receive an accuracy score of 85.45%. Below is the confusion matrix.



```
[[390   0   4   4   1   7   3   0   5   0]
 [  0 462   5   7   0   0   0   0   4   0]
 [  5   5 359  20   3   2   7   2  13   4]
 [  5   3  17 353   1  22  10   7  21   7]
 [  0   2   3   2 352   2   2   5   3  33]
 [  8   4   7  29  14 291  11   2  14   8]
 [  6   0   7   0   4  10 370   1   4   2]
 [  2   2   6   7   6   2   1 408   3  28]
 [  7   9  13  25   7  19   7   4 296   6]
 [  3   2   2  12  24   5   1  22   9 306]]
```

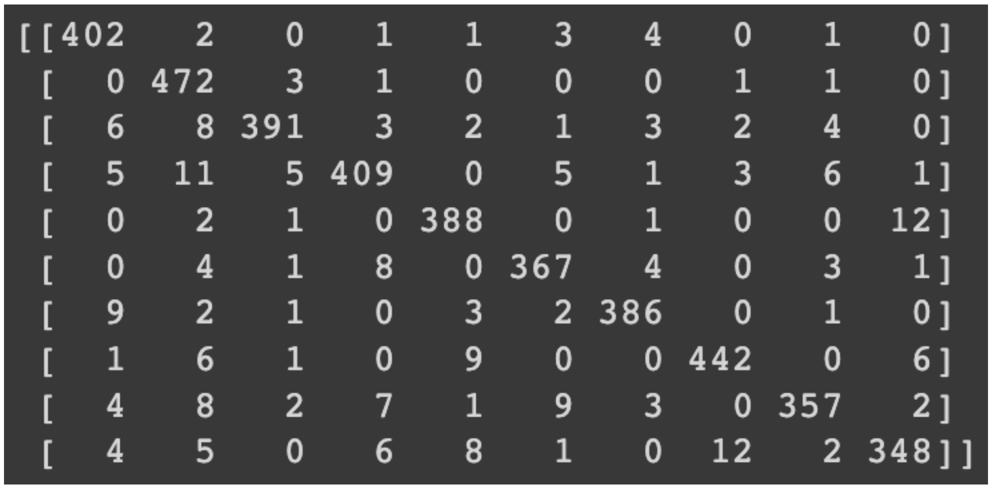
SVM –

1. We create a base SVM model with default settings, using the training data. We use this model to make predictions on the test data set and calculate its accuracy.
2. After testing the model against the test data set, we receive an accuracy score of 95.31% which is a lot higher than the LinearSVC's accuracy. Below is the confusion matrix.



```
[[408   0   1   0   1   0   2   0   2   0]
 [  0 469   5   2   0   1   0   0   1   0]
 [  1   2 403   1   5   1   3   3   0   1]
 [  2   3   6 415   0   7   1   5   5   2]
 [  1   1   2   0 385   0   0   0   1  14]
 [  0   2   0   7   1 372   4   0   1   1]
 [  3   0   1   0   2   5 392   0   1   0]
 [  0   4   2   0   9   0   0 438   0  12]
 [  1   5   3   7   2   9   1   0 365   0]
 [  3   2   0   6  10   2   0   8   1 354]]
```

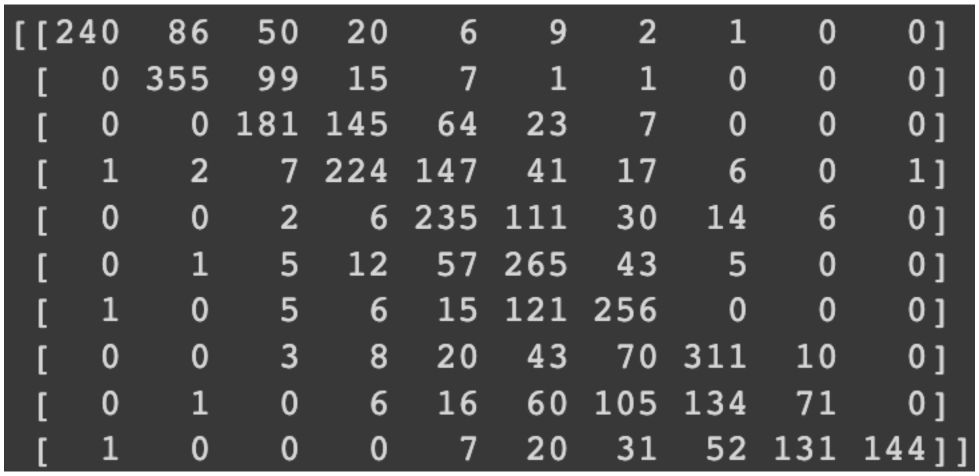
3. To tune the model, we create different SVM models with values of C in the range [0.001, 0.01, 0.1, 1, 10, 100], values of gamma in the range [0.0001, 0.001, 0.01, 0.1] and kernels in ['linear','rbf', 'poly']. We find that the best SVM model is created with C = 0.001, gamma = 0.0001 and kernel = 'poly'.
4. We use these settings to create a new SVM model and test its accuracy score against the test data. We receive an accuracy of 94.38% which shows that the accuracy of the base SVM model was better. Below is the confusion matrix.



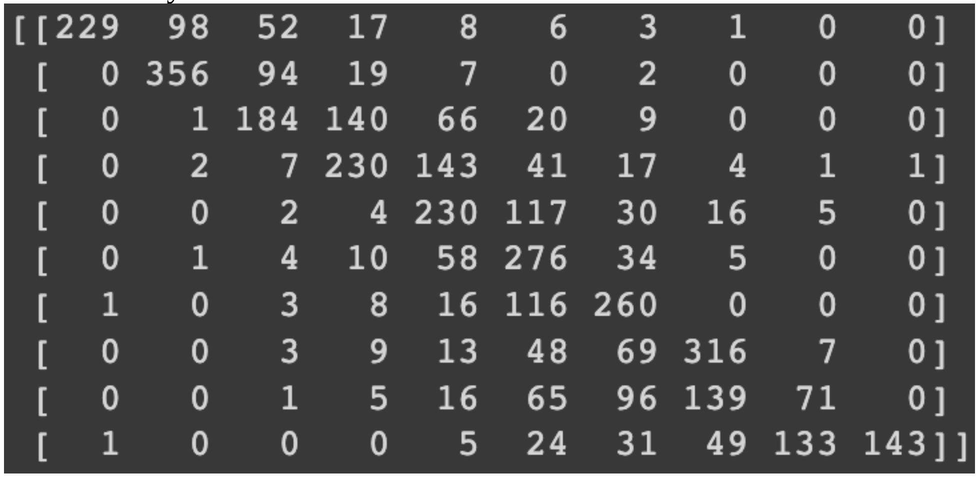
```
[[402   2   0   1   1   3   4   0   1   0]
 [  0 472   3   1   0   0   0   1   1   0]
 [  6   8 391   3   2   1   3   2   4   0]
 [  5  11   5 409   0   5   1   3   6   1]
 [  0   2   1   0 388   0   1   0   0  12]
 [  0   4   1   8   0 367   4   0   3   1]
 [  9   2   1   0   3   2 386   0   1   0]
 [  1   6   1   0   9   0   0 442   0   6]
 [  4   8   2   7   1   9   3   0 357   2]
 [  4   5   0   6   8   1   0  12   2 348]]
```

SECTION 4 – RANDOM FOREST

1. We create a base Random Forest model with default settings, using the training data. We use this model to make predictions on the test data set and calculate its accuracy.

2. After testing the model against the test data set, we receive an accuracy score of 80.56%. Below is the confusion matrix.


```
[[240  86  50  20   6   9   2   1   0   0]
 [  0 355  99  15   7   1   1   0   0   0]
 [  0   0 181 145  64  23   7   0   0   0]
 [  1   2   7 224 147  41  17   6   0   1]
 [  0   0   2   6 235 111  30  14   6   0]
 [  0   1   5  12  57 265  43   5   0   0]
 [  1   0   5   6  15 121 256   0   0   0]
 [  0   0   3   8  20  43  70 311  10   0]
 [  0   1   0   6  16  60 105 134  71   0]
 [  1   0   0   0   7  20  31  52 131 144]]
```

3. To tune the model, we create different Random Forest models with values of estimators in between 0 and 1000 with increments of 100. We find that the best Random Forest model exists when n_estimators = 800.
4. We use these settings to create a new Random Forest model and test its accuracy score against the test data. We receive an accuracy of 80.81% which is only slightly higher than the accuracy of base random forest model. Below is the confusion matrix.


```
[[229  98  52  17   8   6   3   1   0   0]
 [  0 356  94  19   7   0   2   0   0   0]
 [  0   1 184 140  66  20   9   0   0   0]
 [  0   2   7 230 143  41  17   4   1   1]
 [  0   0   2   4 230 117  30  16   5   0]
 [  0   1   4  10  58 276  34   5   0   0]
 [  1   0   3   8  16 116 260   0   0   0]
 [  0   0   3   9  13  48  69 316   7   0]
 [  0   0   1   5  16  65  96 139  71   0]
 [  1   0   0   0   5  24  31  49 133 143]]
```

SECTION 5 – ALGORITHM PERFORMANCE COMPARISON

Model	Base kNN	Tuned kNN	Base LinearSVC	Base SVM	Tuned SVM	Base RF	Tuned RF
Accuracy	88.59%	89.44%	85.45%	95.31%	94.38%	80.56%	80.81%

The above table shows the accuracies of each model that we have tested in this homework. Clearly, the SVM models outperform the other two models by a significant margin. Although the accuracy of tuned SVM model is less than the accuracy of the base SVM model, it is preferrable to use one of these two SVM models to predict labels. Of these 3, Random Forest algorithm generates models with the lowest accuracies. Parameter tuning only has a marginal impact on the accuracy of model in all 3 algorithms.

Shivangi Mundhra
SUID - 842548148

The accuracy of Random Forest suffers because Random Forest is derived from Decision Trees. The accuracy of Decision Tree model was, as seen in previous assignment, around 78%. Clearly using multiple decision trees does not increase the accuracy by a lot. Support Vector Machine can clearly create distinct hyperplanes between different labels which leads to such a high accuracy.