Question 1

a) $W^{(1)}$ is matrix of size $d \times d$ because it operates on a vector x with d features including the bias term and it returns the vector in $\mathbb{R}^d$. $z_1 \in \mathbb{R}^d$ because h, the return of the activation function, is in $\mathbb{R}^d$, which implies that $z_1 \in \mathbb{R}^d$. $z_2 \in \mathbb{R}^d$, because the sum of two vectors is well defined only if they are of the same dimension and we know that $h \in \mathbb{R}^d$. $W^{(2)}$ is a matrix of size $1 \times d$; this is because it operates on a vector $z_2 \in \mathbb{R}^d$ and returns a scalar or a vector in $\mathbb{R}$.

b) The total number of parameters as a function of d is given by $p(d) = d^2 + d$, i.e. the number of scalars in the $W^{(1)}$ and $W^{(2)}$ matrices.

c)

$$\begin{aligned}
\overline{y} &= \frac{\partial \mathcal{L}}{\partial y} = y - t \\
\overline{W}^{(2)} &= \frac{\partial \mathcal{L}}{\partial W^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \frac{\partial y}{\partial W^{(2)}} = \overline{y} x^T \\
\overline{z}_2 &= \frac{\partial \mathcal{L}}{\partial z_2} = \frac{\partial \mathcal{L}}{\partial y} \frac{\partial y}{\partial z_2} = \overline{y} W^{(2)} \\
\overline{h} &= \frac{\partial \mathcal{L}}{\partial h} = \frac{\partial \mathcal{L}}{\partial z_2} \frac{\partial z_2}{\partial h} = \overline{z}_2 I_d = \overline{z}_2 \\
\overline{z}_1 &= \frac{\partial \mathcal{L}}{\partial \overline{z}_1} = \frac{\partial \mathcal{L}}{\partial h} \frac{\partial h}{\partial \overline{z}_1} = \overline{h} \sigma'(z) \\
\overline{W}^{(1)} &= \frac{\partial \mathcal{L}}{\partial W^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_1} \frac{z_1}{\partial W^{(1)}} = \overline{z} W^{(1)} \\
\overline{x} &= \frac{\partial \mathcal{L}}{\partial x} = \overline{z}_1 \frac{\partial z_1}{\partial x} + \overline{z}_2 \frac{\partial z_2}{\partial x} = \overline{z}_1 W^{(1)} + \overline{z}_2 I_d = \overline{z}_1 W^{(1)} + \overline{z}_2
\end{aligned}$$

Question 2

a) We consider the two cases. When $k = k'$, we get that:

$$\begin{aligned}
\frac{\partial y_k}{\partial z_k} &= \frac{\partial}{\partial z_k} \frac{\exp(z_k)}{\sum_{k'=1}^{K} \exp(z_{k'})} \\
&= \frac{e^{z_k} \sum_{k'=1}^{K} \exp(x_{k'}) - e^{2z_k}}{(\sum_{k'=1}^{K} \exp(z_{k'}))^2} \\
&= \frac{e^{z_k}}{\sum_{k'=1}^{K} \exp(z_{k'})} - \Big(\frac{e^{z_k}}{\sum_{k'=1}^{K} \exp(z_{k'})}\Big)^2 \\
&= y_k - y_k^2
\end{aligned}$$

Now we consider the case $k \neq k'$:

$$\begin{aligned}
\frac{\partial y_k}{\partial z_{k'}} &= \frac{\partial}{\partial z_{k'}} \frac{\exp(z_k)}{\sum_{k''=1}^{K} \exp(z_{k''})} \\
&= -\frac{0 - \exp(z_k)\exp(z_{k'})}{(\sum_{k'=1}^{K} \exp(z_{k'}))^2} \\
&= -y_k y_{k'}
\end{aligned}$$

b) First we note that:

$$\begin{aligned}
\frac{\partial \mathcal{L}(t, y)}{\partial w_k} &= \frac{\partial \mathcal{L}}{\partial t}\frac{\partial t}{\partial w_k} + \frac{\partial \mathcal{L}}{\partial y}\frac{\partial y}{\partial w_k} \\
&= \frac{\partial \mathcal{L}}{\partial y}\frac{\partial y}{\partial w_k} && \text{since } \frac{\partial t}{\partial w_k} = 0
\end{aligned}$$

Now we compute the separate derivatives:

$$\frac{\partial \mathcal{L}}{\partial y} = \begin{pmatrix} \frac{\partial \mathcal{L}}{\partial y_1} & \dots & \frac{\partial \mathcal{L}}{\partial y_k} \end{pmatrix} = \begin{pmatrix} -\frac{t_1}{y_1} & \dots & -\frac{t_k}{y_k} \end{pmatrix}$$

and

$$\begin{aligned}
\frac{\partial y}{\partial w_k} &= \frac{\partial y}{\partial z_k}\frac{\partial z_k}{\partial w_k} \\
&= \begin{pmatrix} \frac{\partial y_1}{\partial z_k} \\ \vdots \\ \frac{\partial y_k}{\partial z_k} \\ \vdots \\ \frac{\partial y_K}{\partial z_k} \end{pmatrix} \cdot \frac{\partial}{\partial w_k}(w_k \cdot x) = \begin{pmatrix} -y_1 y_k \\ \vdots \\ y_k(1 - y_k) \\ \vdots \\ -y_K y_k \end{pmatrix} \cdot x
\end{aligned}$$

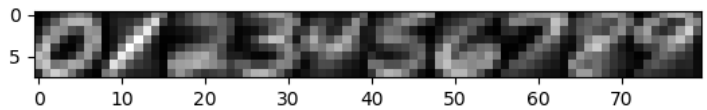
and so we get that:

$$
\begin{aligned}
\frac{\partial \mathcal{L}(t,y)}{\partial w_k} &= \frac{\partial \mathcal{L}}{\partial y}\frac{\partial y}{\partial w_k} \\
&= \begin{pmatrix} -\frac{t_1}{y_1} & \cdots & -\frac{t_k}{y_k} \end{pmatrix} \cdot \begin{pmatrix} -y_1 y_k \\ \vdots \\ y_k(1-y_k) \\ \vdots \\ -y_K y_k \end{pmatrix} \cdot x \\
&= (t_1 y_k + \cdots + t_k(y_k - 1) + \cdots + t_K y_k) \cdot x \\
&= (y_k(\sum_{i=1}^{K} t_i) - t_k) \cdot x \\
&= (y_k - t_k) \cdot x && \text{since } \sum_{i=1}^{K} t_i = 1
\end{aligned}
$$

as wanted.

Question 3

3.0



3.1.1.

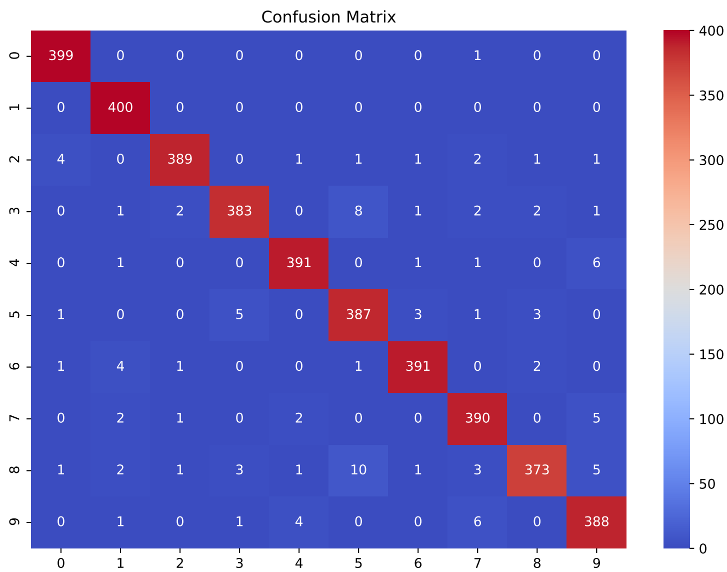
Test accuracy of K = 1: 0.969
Train accuracy for K = 1: 1.0
Test accuracy for K = 15: 0.961
Train accuracy for K = 15: 0.964

3.1.2. The method I decided on for breaking ties is choosing THE nearest label amongst the tied labels. I think this is a reasonable way to break ties because our classifier believes that each of the tied labels could be the digit and so choosing the nearest training point to the test point in question, should in more cases than not, give us the greatest likelihood of being that digit. Empirically, this was justified because I wrote code that randomly picked and the classifier that picked the closest training point to the test point performed with much better accuracy.

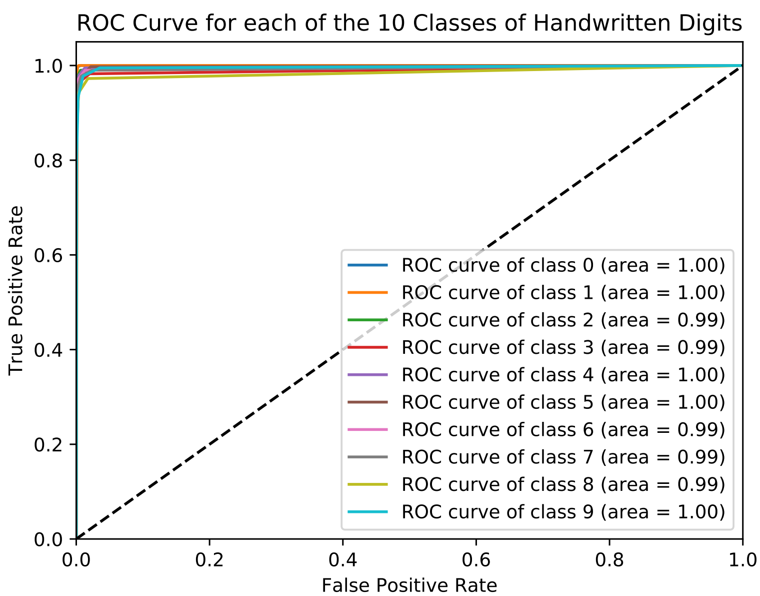
3.1.3.

KNN Parameters and Metrics:

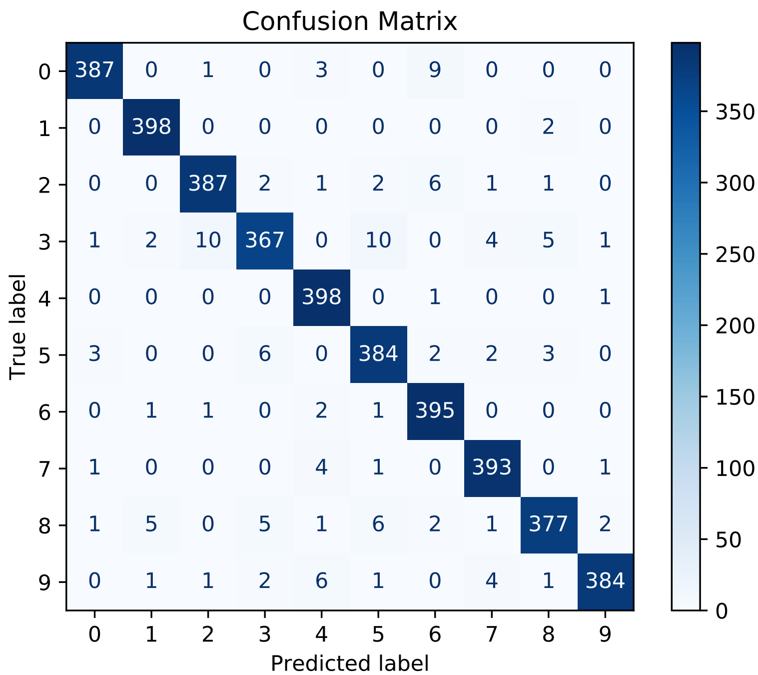
The average accuracy across folds for K = 1 : 0.965
The average accuracy across folds for K = 2 : 0.965
The average accuracy across folds for K = 3 : 0.966
The average accuracy across folds for K = 4 : 0.967
The average accuracy across folds for K = 5 : 0.964
The average accuracy across folds for K = 6 : 0.966
The average accuracy across folds for K = 7 : 0.961
The average accuracy across folds for K = 8 : 0.963
The average accuracy across folds for K = 9 : 0.957
The average accuracy across folds for K = 10 : 0.960
The average accuracy across folds for K = 11 : 0.958
The average accuracy across folds for K = 12 : 0.957
The average accuracy across folds for K = 13 : 0.955
The average accuracy across folds for K = 14 : 0.955
The average accuracy across folds for K = 15 : 0.953
The value of optimal K: 4
Test accuracy: 0.973
Train accuracy: 0.986
Error Rate: 0.027
Precision Score for digit 0 : 0.983
Precision Score for digit 1 : 0.973
Precision Score for digit 2 : 0.987
Precision Score for digit 3 : 0.977
Precision Score for digit 4 : 0.980
Precision Score for digit 5 : 0.951
Precision Score for digit 6 : 0.982
Precision Score for digit 7 : 0.961
Precision Score for digit 8 : 0.979
Precision Score for digit 9 : 0.956
Recall Score for digit 0 : 0.998
Recall Score for digit 1 : 1.000
Recall Score for digit 2 : 0.973
Recall Score for digit 3 : 0.958
Recall Score for digit 4 : 0.978
Recall Score for digit 5 : 0.968
Recall Score for digit 6 : 0.978
Recall Score for digit 7 : 0.975
Recall Score for digit 8 : 0.932
Recall Score for digit 9 : 0.970

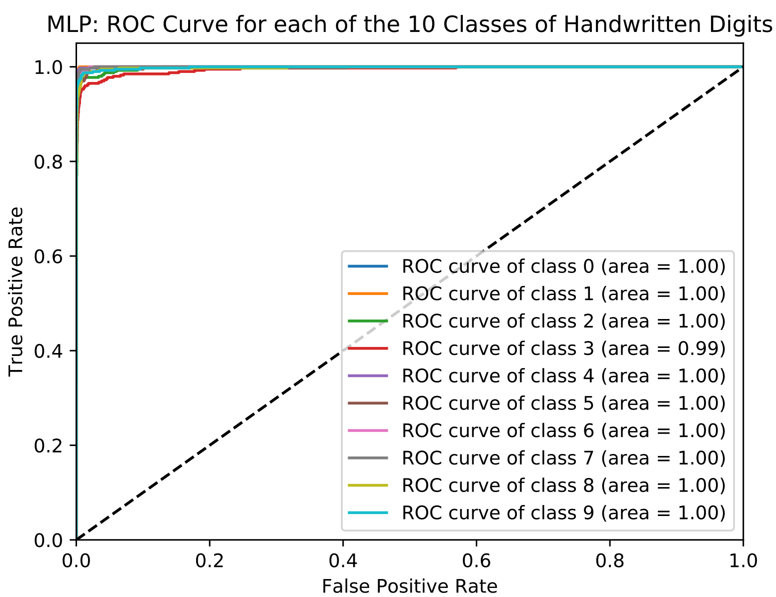


Note that the x-axis of the matrix represents the predicted labels and the y-axis of the matrix represents the true labels.

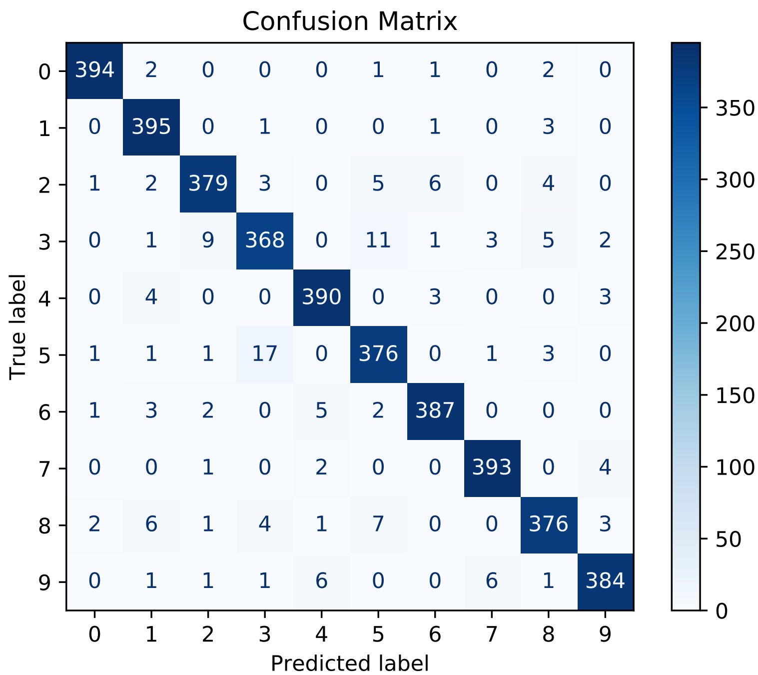


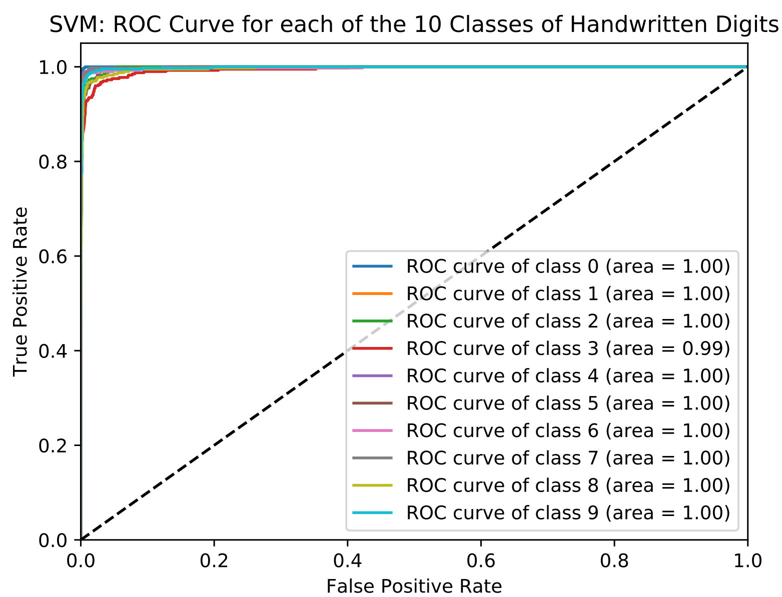
MLP Parameters and Metrics:
Best Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (50, 50), 'learning_rate_init': 0.01, 'solver': 'adam'}
Test accuracy: 0.968
Train accuracy: 0.999
Error Rate: 0.032
Precision Score for digit 0 : 0.985
Precision Score for digit 1 : 0.978
Precision Score for digit 2 : 0.968
Precision Score for digit 3 : 0.961
Precision Score for digit 4 : 0.959
Precision Score for digit 5 : 0.948
Precision Score for digit 6 : 0.952
Precision Score for digit 7 : 0.970
Precision Score for digit 8 : 0.969
Precision Score for digit 9 : 0.987
Recall Score for digit 0 : 0.968
Recall Score for digit 1 : 0.995
Recall Score for digit 2 : 0.968
Recall Score for digit 3 : 0.917
Recall Score for digit 4 : 0.995
Recall Score for digit 5 : 0.960
Recall Score for digit 6 : 0.988
Recall Score for digit 7 : 0.983
Recall Score for digit 8 : 0.943
Recall Score for digit 9 : 0.960


Confusion Matrix
True label
Predicted label
387 0 1 0 3 0 9 0 0 0
0 398 0 0 0 0 0 0 2 0
0 0 387 2 1 2 6 1 1 0
1 2 10 367 0 10 0 4 5 1
0 0 0 0 398 0 1 0 0 1
3 0 0 6 0 384 2 2 3 0
0 1 1 0 2 1 395 0 0 0
1 0 0 0 4 1 0 393 0 1
1 5 0 5 1 6 2 1 377 2
0 1 1 2 6 1 0 4 1 384

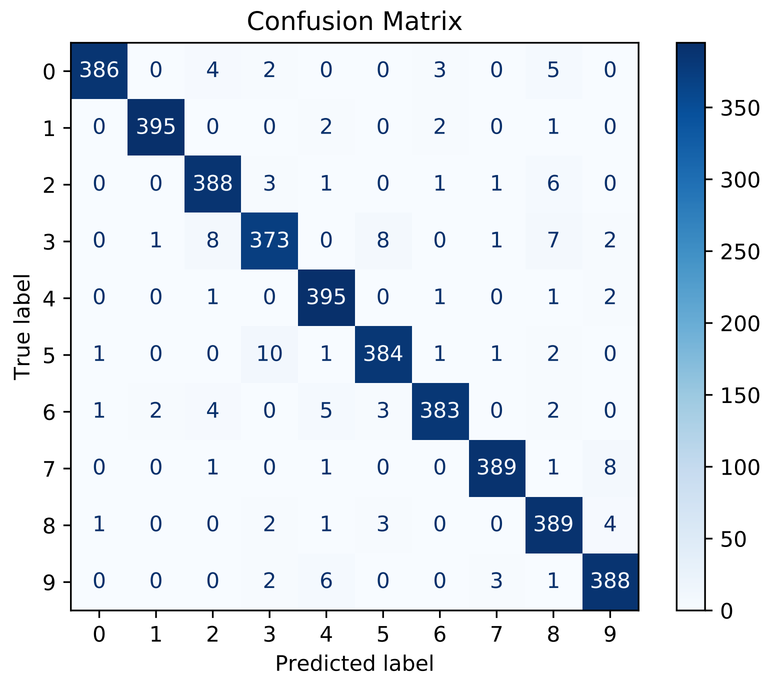

MLP: ROC Curve for each of the 10 Classes of Handwritten Digits
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = 1.00)
ROC curve of class 1 (area = 1.00)
ROC curve of class 2 (area = 1.00)
ROC curve of class 3 (area = 0.99)
ROC curve of class 4 (area = 1.00)
ROC curve of class 5 (area = 1.00)
ROC curve of class 6 (area = 1.00)
ROC curve of class 7 (area = 1.00)
ROC curve of class 8 (area = 1.00)
ROC curve of class 9 (area = 1.00)

SVM Parameters and Metrics:
Best Parameters: {'C': 100, 'gamma': 0.001, 'kernel': 'rbf', 'probability': True}
Test accuracy: 0.9605
Train accuracy: 0.969
Error Rate: 0.039
Precision Score for digit 0 : 0.987
Precision Score for digit 1 : 0.952
Precision Score for digit 2 : 0.962
Precision Score for digit 3 : 0.934
Precision Score for digit 4 : 0.965
Precision Score for digit 5 : 0.935
Precision Score for digit 6 : 0.970
Precision Score for digit 7 : 0.975
Precision Score for digit 8 : 0.954
Precision Score for digit 9 : 0.970
Recall Score for digit 0 : 0.985
Recall Score for digit 1 : 0.988
Recall Score for digit 2 : 0.948
Recall Score for digit 3 : 0.920
Recall Score for digit 4 : 0.975
Recall Score for digit 5 : 0.940
Recall Score for digit 6 : 0.968
Recall Score for digit 7 : 0.983
Recall Score for digit 8 : 0.940
Recall Score for digit 9 : 0.960

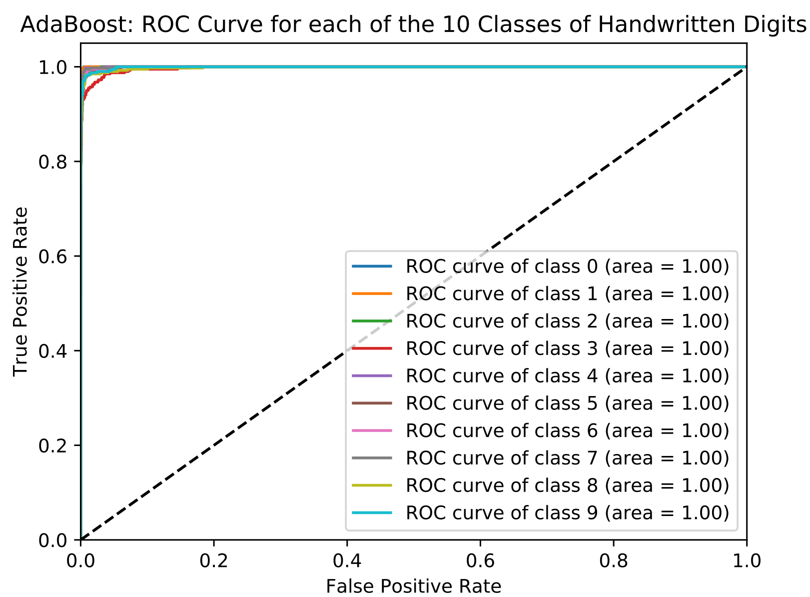

Confusion Matrix
True label
Predicted label
0
1
2
3
4
5
6
7
8
9
394 2 0 0 0 1 1 0 2 0
0 395 0 1 0 0 1 0 3 0
1 2 379 3 0 5 6 0 4 0
0 1 9 368 0 11 1 3 5 2
0 4 0 0 390 0 3 0 0 3
1 1 1 17 0 376 0 1 3 0
1 3 2 0 5 2 387 0 0 0
0 0 1 0 2 0 0 393 0 4
2 6 1 4 1 7 0 0 376 3
0 1 1 1 6 0 0 6 1 384
350
300
250
200
150
100
50
0


SVM: ROC Curve for each of the 10 Classes of Handwritten Digits
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve of class 0 (area = 1.00)
ROC curve of class 1 (area = 1.00)
ROC curve of class 2 (area = 1.00)
ROC curve of class 3 (area = 0.99)
ROC curve of class 4 (area = 1.00)
ROC curve of class 5 (area = 1.00)
ROC curve of class 6 (area = 1.00)
ROC curve of class 7 (area = 1.00)
ROC curve of class 8 (area = 1.00)
ROC curve of class 9 (area = 1.00)

AdaBoost Parameters and Metrics:
Best Parameters: {'base_estimator': DecisionTreeClassifier(max_depth = 8), 'learning_rate': 0.75, 'n_estimators': 150}
Test accuracy: 0.9675
Train accuracy: 1.0
Error Rate: 0.032
Precision Score for digit 0 : 0.992
Precision Score for digit 1 : 0.992
Precision Score for digit 2 : 0.956
Precision Score for digit 3 : 0.952
Precision Score for digit 4 : 0.959
Precision Score for digit 5 : 0.965
Precision Score for digit 6 : 0.980
Precision Score for digit 7 : 0.985
Precision Score for digit 8 : 0.937
Precision Score for digit 9 : 0.960
Recall Score for digit 0 : 0.965
Recall Score for digit 1 : 0.988
Recall Score for digit 2 : 0.970
Recall Score for digit 3 : 0.932
Recall Score for digit 4 : 0.988
Recall Score for digit 5 : 0.960
Recall Score for digit 6 : 0.958
Recall Score for digit 7 : 0.973
Recall Score for digit 8 : 0.973
Recall Score for digit 9 : 0.970



Confusion Matrix
True label
Predicted label
0
1
2
3
4
5
6
7
8
9
386 0 4 2 0 0 3 0 5 0
0 395 0 0 2 0 2 0 1 0
0 0 388 3 1 0 1 1 6 0
0 1 8 373 0 8 0 1 7 2
0 0 1 0 395 0 1 0 1 2
1 0 0 10 1 384 1 1 2 0
1 2 4 0 5 3 383 0 2 0
0 0 1 0 1 0 0 389 1 8
1 0 0 2 1 3 0 0 389 4
0 0 0 2 6 0 0 3 1 388
350
300
250
200
150
100
50
0



AdaBoost: ROC Curve for each of the 10 Classes of Handwritten Digits
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve of class 0 (area = 1.00)
ROC curve of class 1 (area = 1.00)
ROC curve of class 2 (area = 1.00)
ROC curve of class 3 (area = 1.00)
ROC curve of class 4 (area = 1.00)
ROC curve of class 5 (area = 1.00)
ROC curve of class 6 (area = 1.00)
ROC curve of class 7 (area = 1.00)
ROC curve of class 8 (area = 1.00)
ROC curve of class 9 (area = 1.00)

Summary

All of the four models performed very well after selecting the optimal hyper-parameters. While the KNN classifier performed slightly better under the various measures, the differences are so small that marginally better performance could be due to slight idiosyncrasies in the data. Therefore, it is difficult to generalize that the KNN classifier performs better than the other model in the task of classifying handwritten digits. For the four selected models, in general, the precision and recall scores were very close to 1. A precision score close to one means that the almost all of the test examples that are classified as positive are actually positive (where a positive example is that a test point that is the specified digit, and a negative example is that a test point that is not the specified digit). On the other hand, a recall score of close to one means that almost all of the positive test examples are classified as positive. Therefore, we can have lots of confidence in our classifiers. Next, we discuss ROC curves. The ROC curve plots the specificity versus the recall for various threshold values to model. A specificity close to one means that almost all of the negative examples are correctly classified as negative. This means that for the ideal model will have both of these axis values close to one, and thus producing an area under the ROC curve of 1. As seen from the above graphs, all of the classifiers have area close to 1, if not, equal to 1, for each of the hand written digit considered. Hence, we can be reasonable confident that each classifier is a very good classifier in the task for recognizing handwritten digits. I did not find the results of this report surprising, because all of the classifiers were thoroughly tuned for the best parameters and are all well-known classifiers, and so it is not surprising that they are so accurate. The classifier I am most impressed with, however, is the AdaBoost classifier. It is very intriguing to me how combining results of various weak classifiers can produce such a strong and accurate classifier. Although we talked about in class why this happens, seeing it practice is rather astonishing. I think the least surprising result was the KNN classifier, just because the idea of it is so simple. The digits are distinct enough that the pixelated form of the digits can easily be distinguished by considering similar examples near it and then taking the most common of them. Another observation that can be seen is that, the test and train accuracies of all the model are fairly close to each other. This implies that the classifiers are actually generalizing very well rather than simply memorizing the data. However, with the training accuracy so high, with some equalling 1, there could be a cause for concern that the classifiers could be over-fitting the data. This is highly unlikely simply because the testing accuracies are so high and the recall, precision, and area under the ROC curve are also near one. Overall, I am really impressed with how accurate the models we have learned so far can be in classifying something so "abstract" as recognizing a hand written digit.