CS5785: Homework 1

Team: Kulvinder Lotay (ksl76) Balaji Kamakoti (bk498)

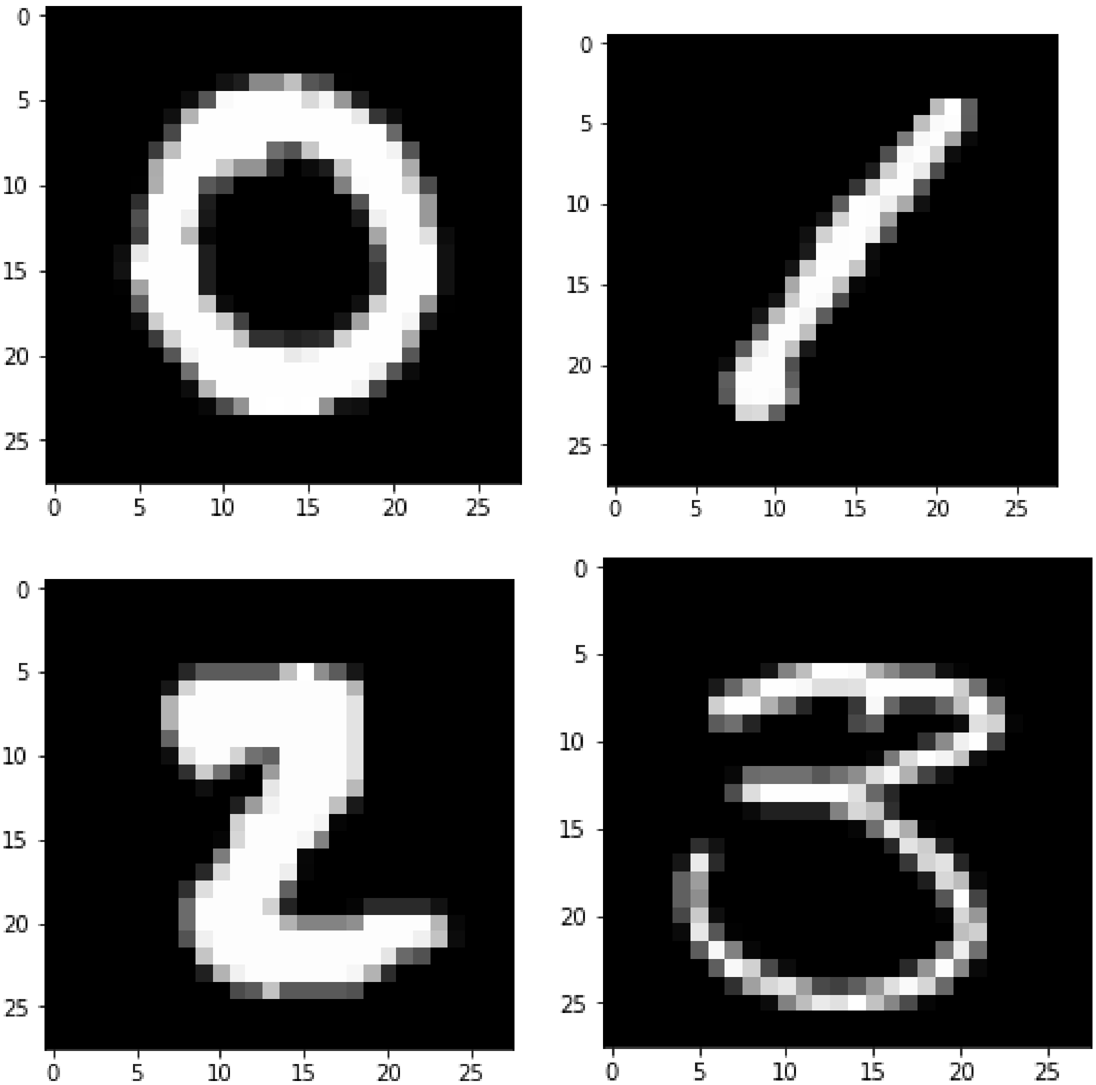
9-13-2018

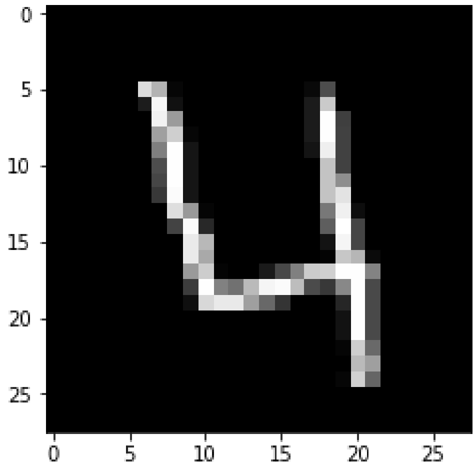
PROGRAMMING EXERCISES

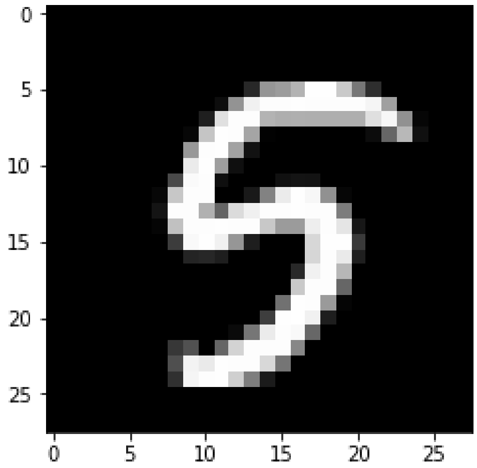
Please check attached .ipynb file for all related code

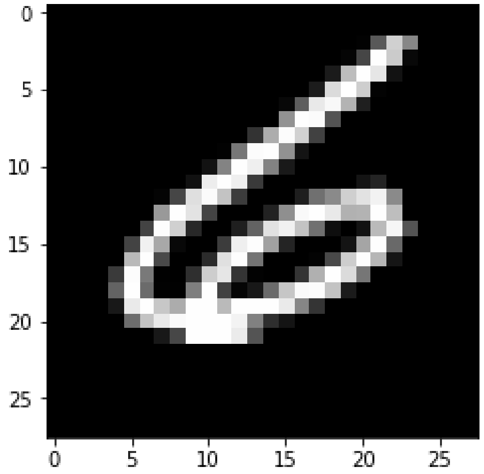
DIGIT RECOGNIZER:

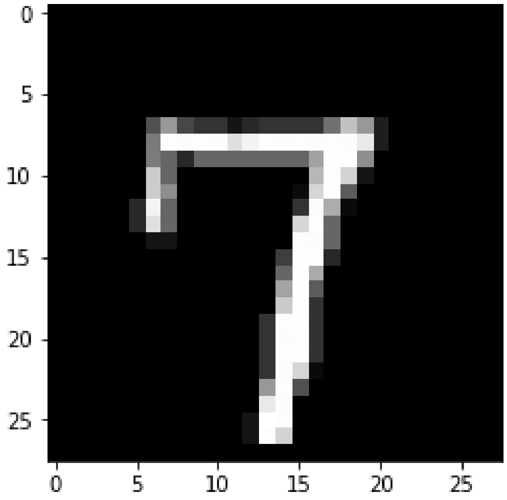
1. Digit Recognizer
    b. Write a function to display anMNIST digit. Display one of each digit.

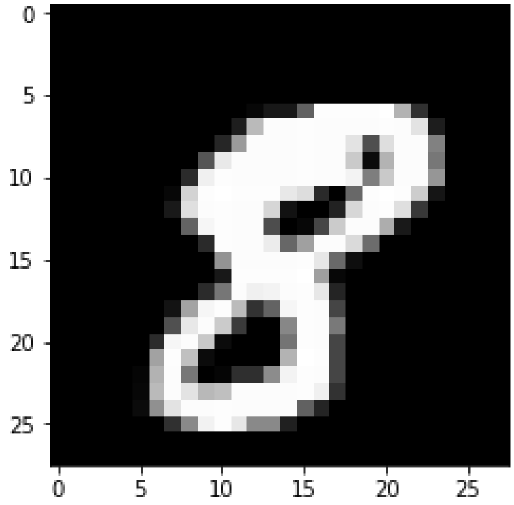


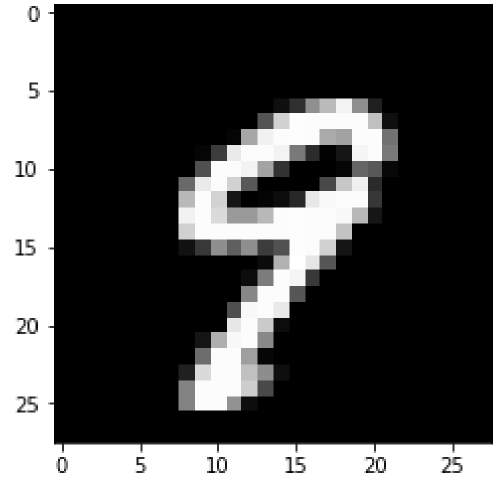

0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25

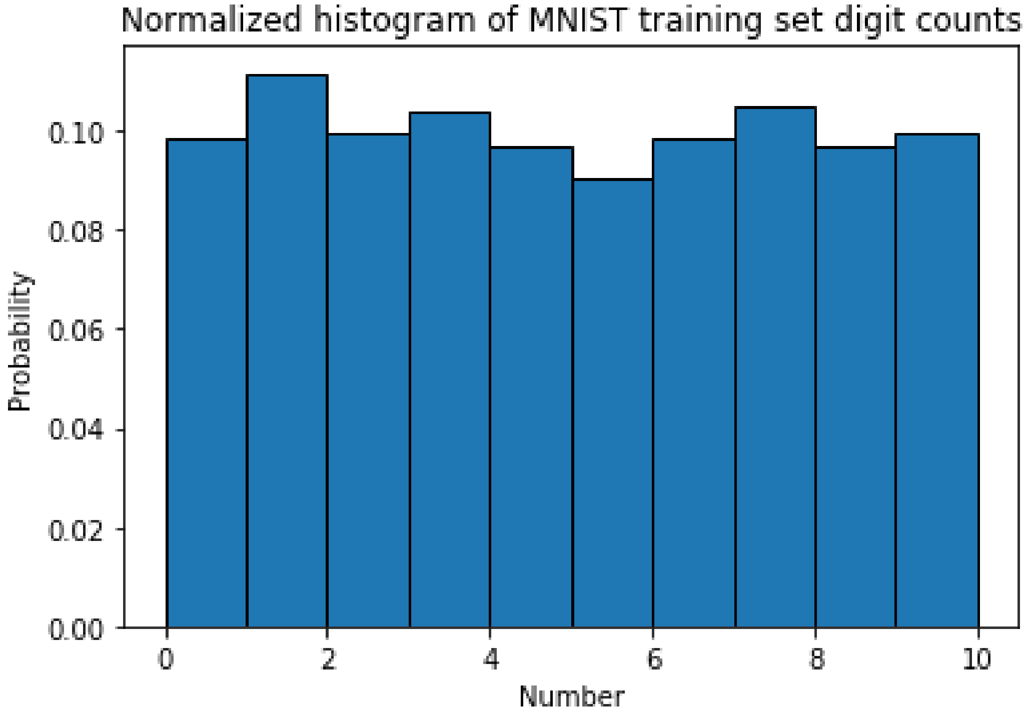

0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25

c. Examine the prior probability of the classes in the training data. Is it uniform across the digits? Display a normalized histogram of digit counts. Is it even?



ANSWER: The histogram is not uniform across the digits, and is not even.

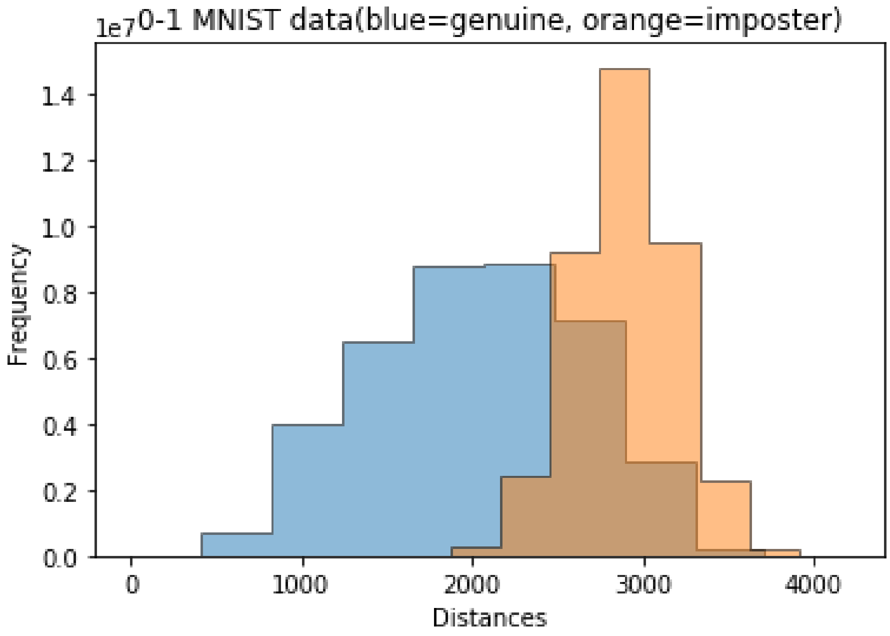
d. Pick one example of each digit from your training data. Then, for each sample digit, compute and show the best match (nearest neighbor) between your chosen sample and the rest of the training data. Use L2 distance between the two images' pixel values as the metric. This probably won't be perfect, so add an asterisk next to the erroneous examples (if any).

ANSWER: Taking the first 10 samples, these are the matches:

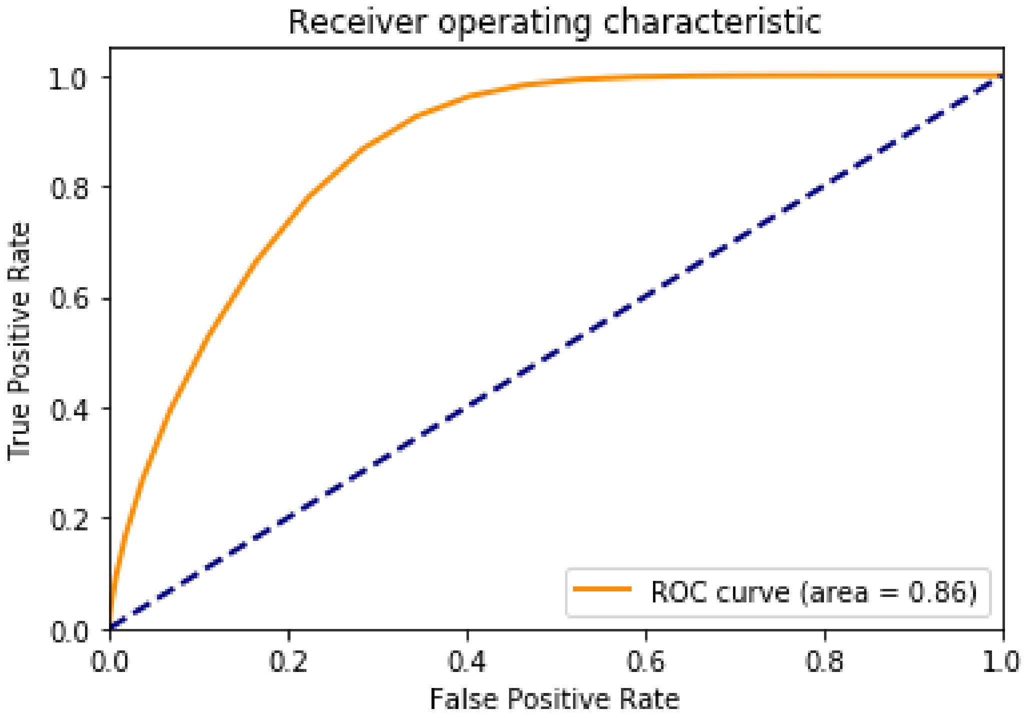
```
SAMPLE: 0
MATCH: 0
True/Genuine

SAMPLE: 1
MATCH: 1
True/Genuine

SAMPLE: 2
MATCH: 2
True/Genuine

SAMPLE: 3
MATCH: 5*
False/Imposter
```

```
SAMPLE: 4
MATCH: 4
True/Genuine

SAMPLE: 5
MATCH: 5
True/Genuine

SAMPLE: 6
MATCH: 6
True/Genuine

SAMPLE: 7
MATCH: 7
True/Genuine

SAMPLE: 8
MATCH: 8
True/Genuine

SAMPLE: 9
MATCH: 9
True/Genuine
```

e. Consider the case of binary comparison between the digits 0 and 1. Ignoring all the other digits, compute the pairwise distances for all genuine matches and all impostor matches, again using the L2 norm. Plot histograms of the genuine and impostor distances on the same set of axes.



f. Generate an ROC curve from the above sets of distances. What is the equal error rate? What is the error rate of a classifier that simply guesses randomly?



EER calculated via method/code provided at https://yangcha.github.io/EER-ROC/

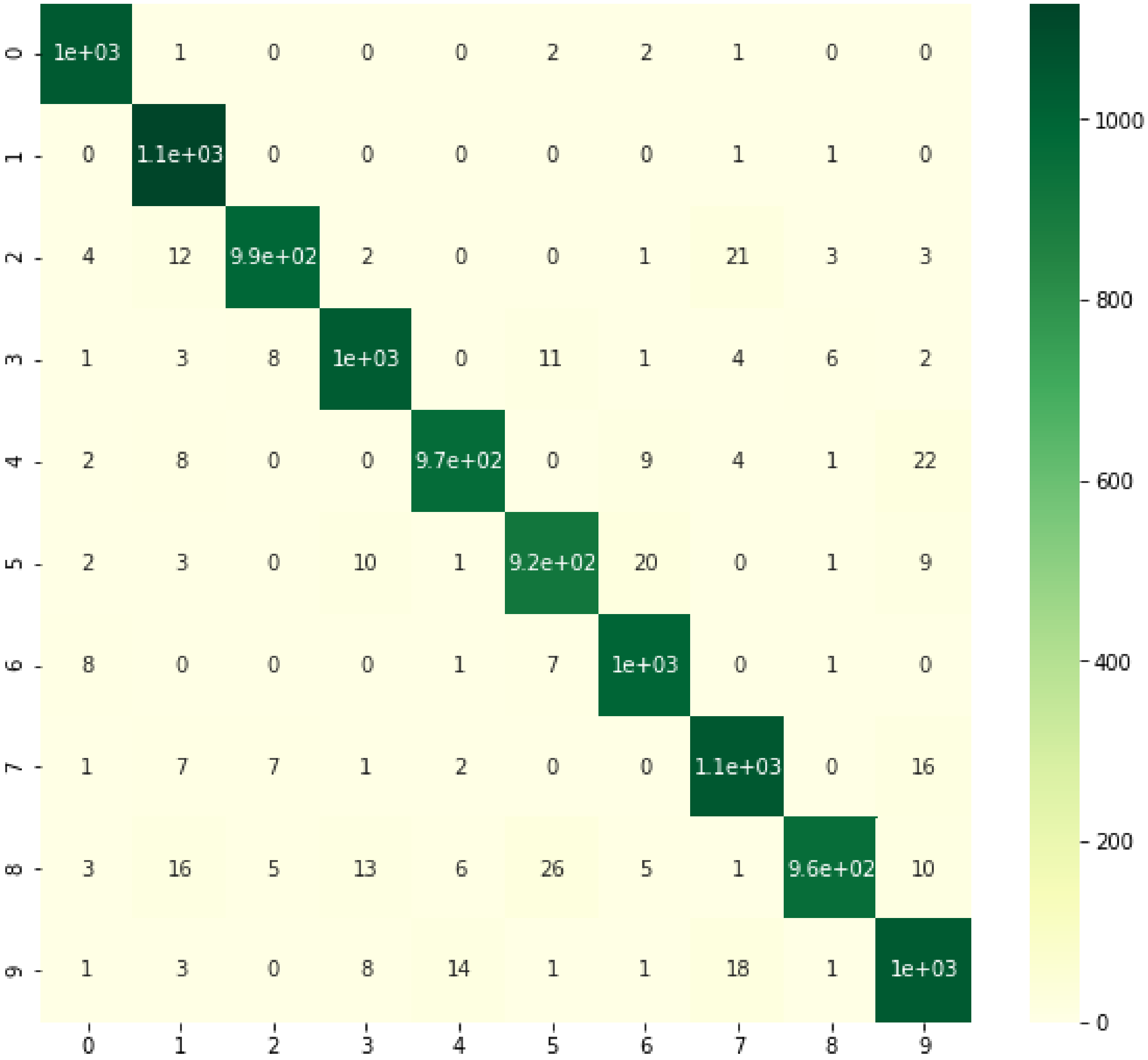
EER = 0.22219236320236982

The error rate of a classifier that simply guesses randomly is represented by the dotted blue line above, and would typically be **0.5** or **50%**.

g. Randomly split the training data into two halves. Train your k-NN classifier on the first half of the data, and test it on the second half, reporting your average accuracy.

Following implementation in attached file, got:
ACCURACY: 96.52380952380952% with *k=5*

h. Generate a confusion matrix (of size 10 x 10) from your results. Which digits are particularly tricky to classify?



	0	1	2	3	4	5	6	7	8	9
0	1e+03	1	0	0	0	2	2	1	0	0
1	0	1.1e+03	0	0	0	0	0	1	1	0
2	4	12	9.9e+02	2	0	0	1	21	3	3
3	1	3	8	1e+03	0	11	1	4	6	2
4	2	8	0	0	9.7e+02	0	9	4	1	22
5	2	3	0	10	1	9.2e+02	20	0	1	9
6	8	0	0	0	1	7	1e+03	0	1	0
7	1	7	7	1	2	0	0	1.1e+03	0	16
8	3	16	5	13	6	26	5	1	9.6e+02	10
9	1	3	0	8	14	1	1	18	1	1e+03

ANSWER: The digits 8 and 9 are trickier to classify.

j. Train your classifier with all of the training data, and test your classifier with the test data. Submit your results to Kaggle.

Submitted results to Kaggle at: https://kaggle.com/c/digit-recognizer
Profile: https://www.kaggle.com/kslotay76

Received a score of 0.969

THE TITANIC DISASTER:

2. The Titanic Disaster
   b. Using logistic regression, try to predict whether a passenger survived the disaster. You can choose the features (or combinations of features) you would like to use or ignore, provided you justify your reasoning.

ANSWER:

Used the following features: Age, Embarked, Sex, PClass, Fare

In cases where there were null values, some columns were dropped where the null values were a majority, while in the following cases:

Age: the median value was filled in
Embarked: the modal value was filled in (as there were very few missing values and the occurrence of the modal value was popular by a significant amount)
Fare: the median value was filled in

The following features were dropped as they were unique per passenger:

PassengerId, Cabin, Name, Ticket

The following features were dropped as they seemed to skew the prediction for survival, following several tests:

SibSp, Parch

PClass and Embarked were broken into columns of binary values, Pclass_1, Pclass_2, Pclass_3 and Embarked_Q, Embarked_C, Embarked_S.

The sex column was split into Sex_male, and Sex_female with binary 1 or 0 representations, the Sex_female column was then dropped.

   c. Train your classifier using all of the training data, and test it using the testing data. Submit your results to Kaggle.

Submitted results to Kaggle at: https://www.kaggle.com/c/titanic
Profile: https://www.kaggle.com/kslotay76

Received a score of 0.74641

WRITTEN EXERCISES

1. Variance of a sum. Show that the variance of a sum is:

   var[X−Y] = *var*[X] + *var*[Y] − 2*cov*[X, Y], where *cov*[X,Y] is the covariance between random variables X and Y .

$$Var(X-Y) = E[\big((X-u_X) - (Y-u_Y)\big)^2$$

$$Var(X-Y) = E[((X-u_X)^2 + (Y-u_Y)^2 - 2(X-u_X)(Y-u_Y)]$$

$$Var(X-Y) = E[(X-u_X)^2] + E[(Y-u_Y)^2] - 2E[(X-u_X)(Y-u_Y)]$$

$$Cov(X,Y) = E[(X-u_X)(Y-u_Y)]$$

$$Var(X,Y) = Var(X) + Var(Y) - 2Cov(X,Y)$$

2. Bayes rule for quality control. You're the foreman at a factory making ten million widgets per year. As a quality control step before shipment, you create a detector that tests for defective widgets before sending them to customers. The test is uniformly 95% accurate, meaning that the probability of testing positive given that the widget is defective is 0.95, as is the probability of testing negative given that the widget is not defective. Further, only one in 100,000 widgets is actually defective.

    a. Suppose the test shows that a widget is defective. What are the chances that it's actually defective given the test result?
    b. If we throw out all widgets that are defective, how many good widgets are thrown away per year? How many bad widgets are still shipped to customers each year?

Random variables:

D: Product is defective

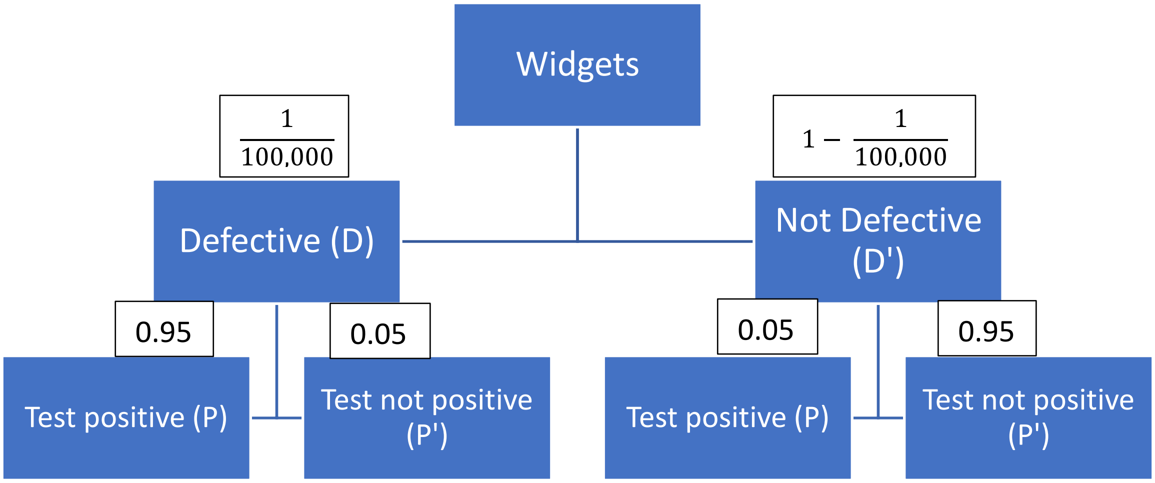
D': Product not defective

P: Quality control test is positive, in other words, test detects defective products

P': Quality control test is negative

Bayes rule:

$$P(A|B) = \frac{P(A)P(B|A)}{P(A)P(B|A) + P(A')P(B|A')}$$



Given:

P(P|D) = 0.95

Hence, P(P’|D) = 0.05

P(P’|D’) = 0.95

P(P|D’) = 0.05

P(D) = 1/100k

P(D’) = 1 – 1/100k

10M units shipped every year

(a) **To find P(D|P)**

$$P(D|P) = \frac{P(D)P(P|D)}{P(D)P(P|D) + P(D')P(P|D')}$$

$$P(D|P) = \frac{(\frac{1}{100{,}000})\,(0.95)}{\left(\frac{1}{100{,}000}\right)(0.95) + \left(1 - \frac{1}{100{,}000}\right)(0.05)}$$

$$P(D|P) = 0.0001899658$$

(b) **To find: Good widgets thrown away per year**. In other words, the positive quality control misclassified a good widget as a defective and as a result had to be thrown away = P(D’|P).

P(D’|P) = $\left(1-\frac{1}{100{,}00}\right)(0.05)\ 10M\ units$

P(D|P) = 499,995 units

To find: Bad widgets shipped to customers. In other words, the quality control test was negative, so defective parts got incorrectly screened as good and shipped to customers = P(D|P’) x 10M units.

$$P(D|P') = \left(1-\frac{1}{100{,}000}\right)(0.05)\ x\ 10M\ units$$

P(D|P) = 5 widgets

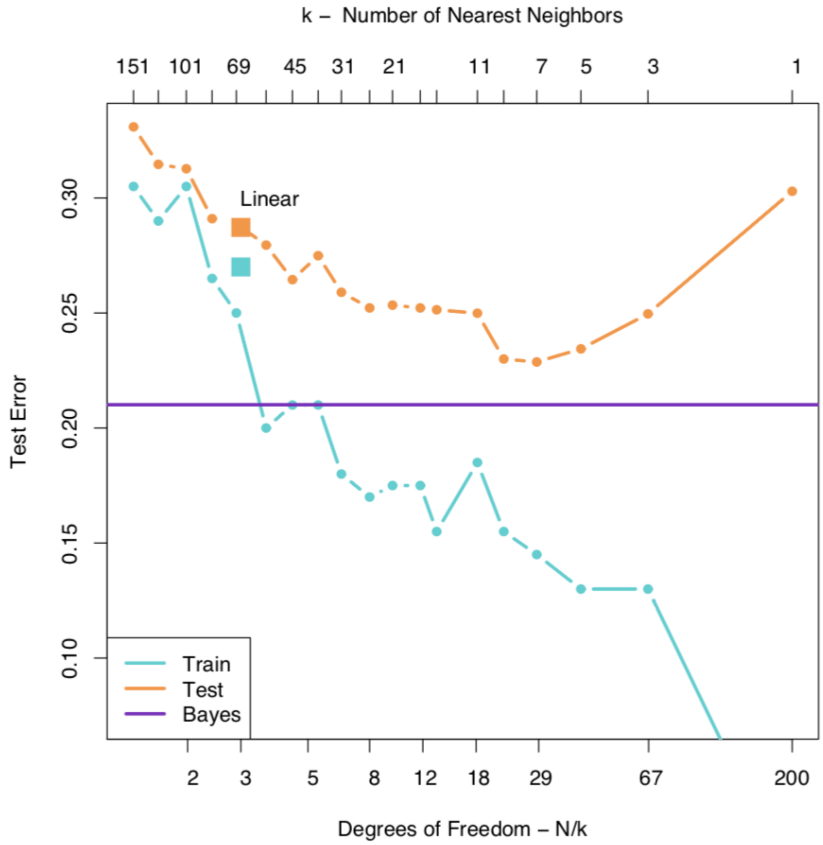


Figure from Elements of Statistical Learning: Section 2.3, Pg 17

3. In *k*-nearest neighbors, the classification is achieved by plurality vote in the vicinity of data. Suppose our training data comprises *n* data points with two classes, each comprising exactly half of the training data, with some overlap between the two classes.

   a. Describe what happens to the average 0-1 prediction error on the training data when the neighbor count *k* varies from *n* to 1. (In this case, the prediction for training data point x_i includes (x_i , y_i) as part of the example training data used by *k*NN).

   ANSWER: As neighbor count *k* varies from *n* to 1, the prediction error will decrease, approaching 0 as *k* approaches 1.

   b. We randomly choose half of the data to be removed from the training data, train on the remaining half, and test on the held-out half. Predict and explain with a sketch how the average 0-1 prediction error on the held-out validation set might change when *k* varies? Explain your reasoning.

ANSWER: As the figure above illustrates, as *k* varies from high values to 1, the prediction error on the training data will decrease. However, for the test data, the prediction will reduce until an optimal point and then increase.

c. In *k*NN, once *k* is determined, all of the *k*-nearest neighbors are weighted equally in deciding the class label. This may be inappropriate when *k* is large. Suggest a modification to the algorithm that avoids this caveat.

ANSWER: In a *kNN* algorithm with a large *k* value, it may be more appropriate to use a weighting function with an output determined by the proximity of nearest neighbors. Put another way, neighbors with lower Euclidean distances get weighted more highly, while neighbors with higher Euclidean distances get weighted less (distance inversely proportional to weight). One way to determine the weights is to analyze the maximum and minimum values of range of Euclidean distances for the neighbors and then selectively apply weights in a range, such as between 2-1 (2 for minimum, 1 for maximum and scaling for the values in between.

d. Give two reasons why *kNN* may be undesirable when the input dimension is high.

ANSWER: Firstly, in data sets with high input dimensions, the distance to all the different features (the different neighbors) becomes less clear due to high variance, blurring the notion of neighbors that are near or far.

Secondly, classifying data sets with high input dimensions requires a much larger training set (potentially exponentially larger) for the model to fit to, which may or not always be available.

e. The plot below shows a binary classification task with two classes: *red* and *blue*. Training data points are shown as triangles, and test data points are shown as squares, and all data points are colored according to their class. The line in the plot is the decision rule from training a logistic regression classifier on the training data, with points above the line classified as *red* and points below the line classified as *blue*.

i. Calculate confusion matrices for both the training and test data for the logistic regression classifier.

Training Set

		Predicted		
Actual		Red	Blue	
	Red	6	3	9
	Blue	3	8	11
		9	11	20

Test Data

		Predicted		
Actual		Red	Blue	
	Red	3	1	4
	Blue	2	4	6
		5	5	10

ii. Now, calculate confusion matrices for both the training and test data for the *k*NN algorithm with *k* = 1 on the same data, using the same train/test split (you should be able to do this by hand).

Training Set

		Predicted		
Actual		Red	Blue	
	Red	5	4	9
	Blue	5	6	11
		10	11	20

Test Data

		Predicted		
Actual		Red	Blue	
	Red	1	4	5
	Blue	3	2	5
		4	6	10

iii. Discuss the difference in accuracy on both the training and test data between the two algorithms. Explain why this difference occurs.

With logistic regression, the accuracy for both the training and test set is 70%. With kNN, the accuracy for the training set is 55% and the test set is 30% set. This difference occurs because with k=1, the kNN has a higher prediction bias, resulting in poor prediction compared to logistic regression.