Assignment #1

Machine Learning with Networks

Recognition of digits using Naïve Bayes Classifier (NBC) and k-nearest-neighbors (KNN)

Garuda Suma Pranavi – UIN: 926009146

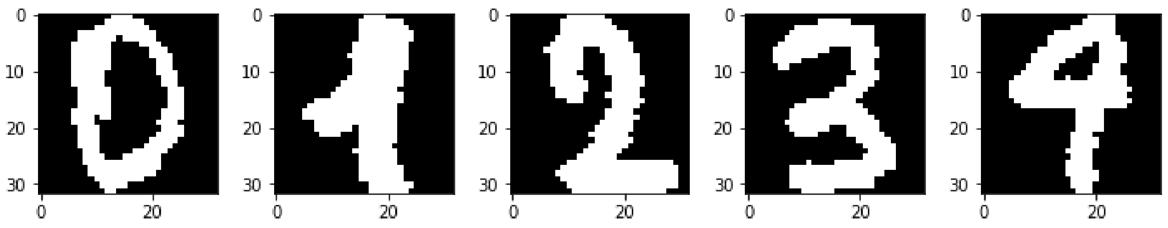
Code for all parts is included in Appendix. I had written the code in Jupyter notebooks and ran it as one unit as some parts are interdependent on each other.

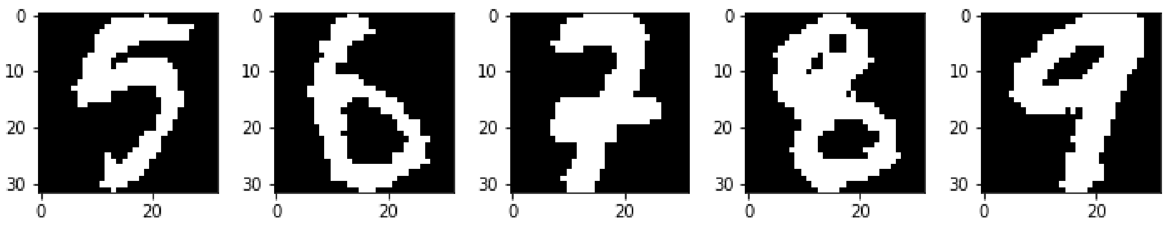
1. Details about the data:

   The training data consists of 1934 text files and testing data consists of 946 text files, each file with 32 rows and 32 columns – each element being a '1' or a '0'. We have converted this 32*32 image to an array with a length of 1024 which are then reshaped back to display the original image.

   The files are named in such a way that they indicate the digit represented by the '1's and '0's, so the label is extracted from the name itself.

   The code has been written to display the training vector as a binary digit.





2. Implementing Naïve Bayes Classifier for digit recognition:

   Used the inbuilt SciKit learn model – Naïve Bayes classifiers. This classifier uses a simple frequentist approach by comparing predicted data to the label to compute the training and testing error rates (i.e. calculating conditional probability). We can make different assumptions about this conditional probability, to see which one performs the best.

$$\hat{y} = \arg\max_{y} P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

Distributions	Bernoulli	Gaussian	Multinomial
Training Error Rate	0.063082	0.208376	0.072389
Testing Error Rate	0.068710	0.266385	0.076110

We can see that the Bernoulli Naïve Bayes classifier model is most suitable for this problem.

3. Implementing k-nearest-neighbor Classifier for digit recognition:

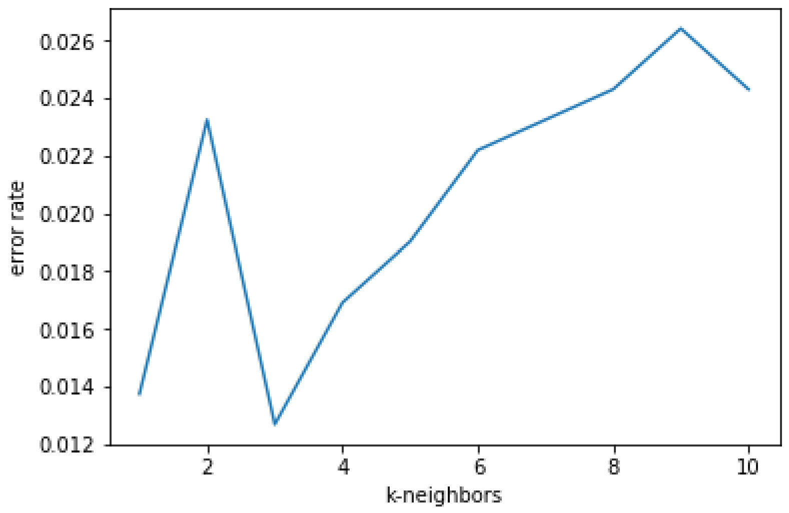
Being one of the easiest supervised learning algorithms , it takes the labels of the k nearest neighbors and classifies the new vector according to the most commonly occurring labels.
The testing error rates have been shown using a frequentist approach i.e. how many testing labels are actually equal to the predicted labels on the testing dataset – using 1-10 neighbors:

k	0	1	2	3	4	5	6	7	8	9
Testing Error Rate	0.0137	0.0232	0.0126	0.0169	0.0190	0.0221	0.0232	0.0243	0.0264	0.0243

We notice that k=3 gives us the minimum error rate for our system and hence classification with the help of 3 neighbors would yield the best results possible.

We can see 2 regions in our plot : k < 3 and k > 3

When k < 3 i.e. high model complexity, the error rate is high because the model overfits the data and negatively impacts the performance of the model on new data. When k > 3, we would expect that larger k's would reduce the noise in classification but this would be completely dependent on our data (it's size).



Comparing the Naïve Bayes classifier and K-nearest-neighbor models, we can see that the NB Classifier has an error rate approx. 4 times that of the KNN model (0.068 and 0.016 respectively)

Naïve Bayes Classifier

Pros: It's very easy to understand and build, can be easily trained even with a small dataset (using smoothening approaches) and is comparatively fast. It also performs well in the multiclass problems.

Cons: The assumption that every feature is independent might not always be the case. (But it doesn't give the worst performance even the assumption doesn't hold).

Possible ways to improve the performance of NBC:

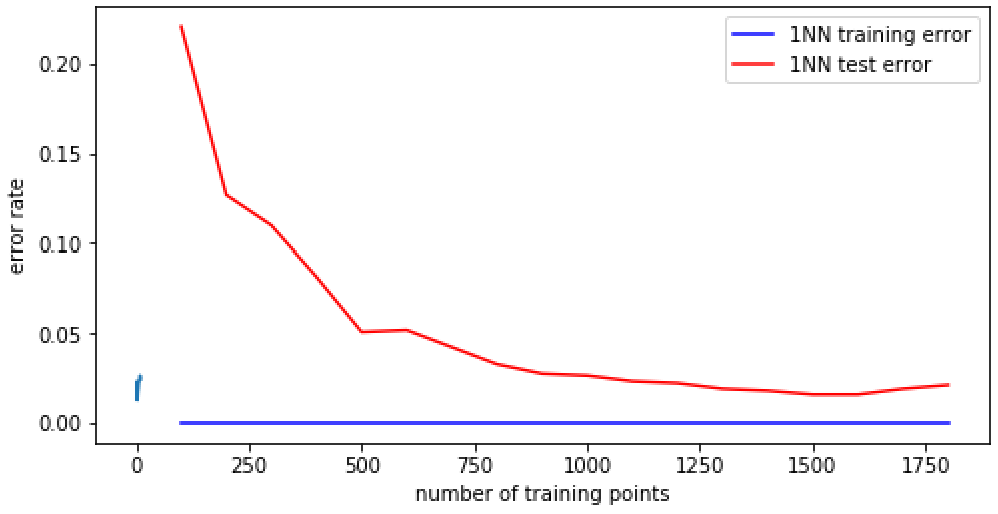
1. As we're calculating joint probabilities and multiplying a number of small values together, it is better to work in the log probability space.
2. We should keep trying distributions that best characterize the data and prediction problem: Gaussian, Exponential, Bernoulli etc.
3. Using probabilities for feature selection: We can use a search algorithm to find the best combination of attributes and evaluate their performance at predicting the output variable.
   We could also remove the redundant features using dimensionality reduction such as PCA (Principal Component Analysis)

K Nearest Neighbor Classifier

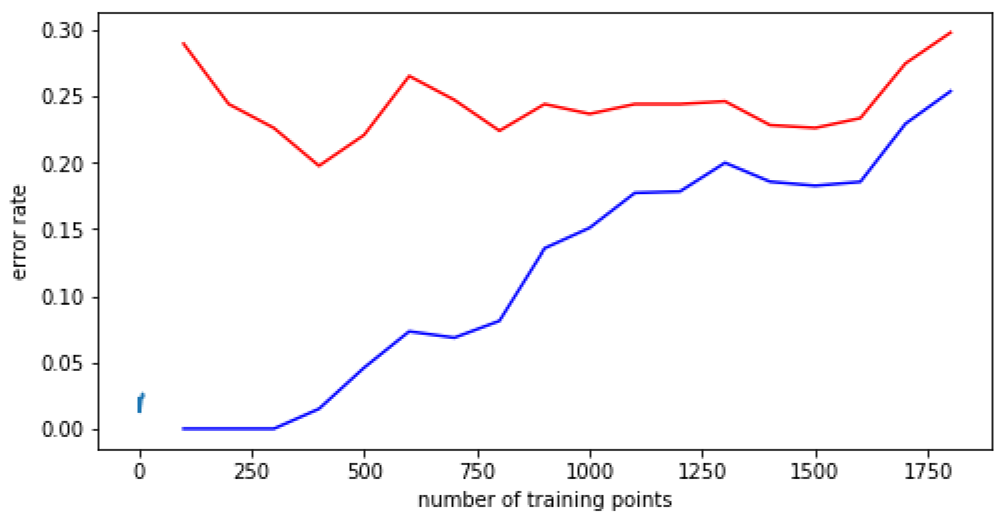
Pros: It is easy to implement, is flexible to feature/distance choices, naturally handles multi class problems and can do well in practice with enough representative data.

Cons: Due to the calculation of nearest neighbor distances every time a prediction is made, it is computationally expensive and can take a lot of time for large training datasets. It might throw errors in some datasets due to it's sensitivity to local data (does not take a generalized view into account).

4. We evaluate the training and testing error rates with varying number of training samples :100,200,300 ……………………………., 1700,1800.



We can see that there is 0 error rate for the training data with 1 neighbor classifier (obviously). However, the error rate gradually (asymptotically) approaches 0 as the number of training data keep increasing when we evaluate the test vector (since we will have more samples of similar data, less distances and hence classification with more probability)



The training error rates increase with large datasets, however the test error rate remains nearly the same whatever the size of the dataset might be.

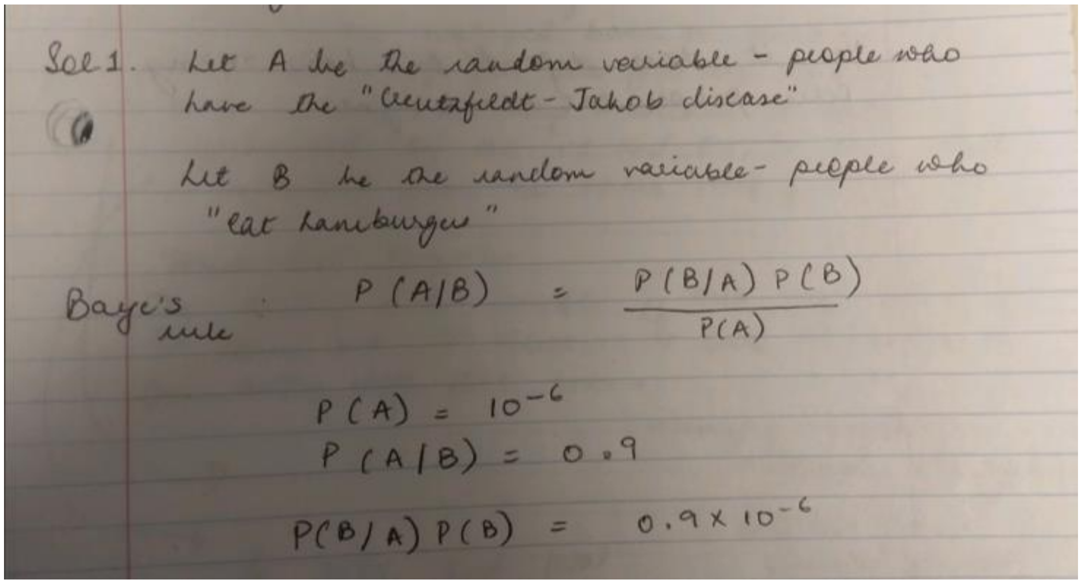
5. **Bonus Question**: Would MNIST dataset help improve the performance of the digits testing_data?

   We can use the MNIST dataset, however we'll have to reshape a few things as the digits dataset is comprised of 32*32 features and the MNIST one comprises of 28*28.

   I used the K neighbors classifier approach and uses the expansive amount of training data in the MNIST dataset, however we see that this leads to no improvement in the prediction performance. (It is even worse than Bayes classifier – **error rate of 0.8974**)

2. Math Assignment

1.



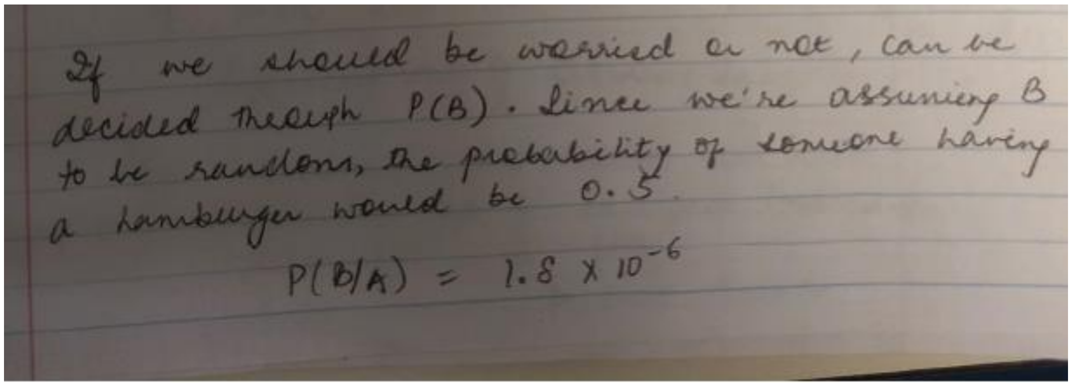
Sol 1. Let A be the random variable - people who have the "Creutzfeldt - Jakob disease"

Let B be the random variable - people who "eat hamburgers"

Baye's rule $$P(A/B) = \frac{P(B/A)\,P(B)}{P(A)}$$

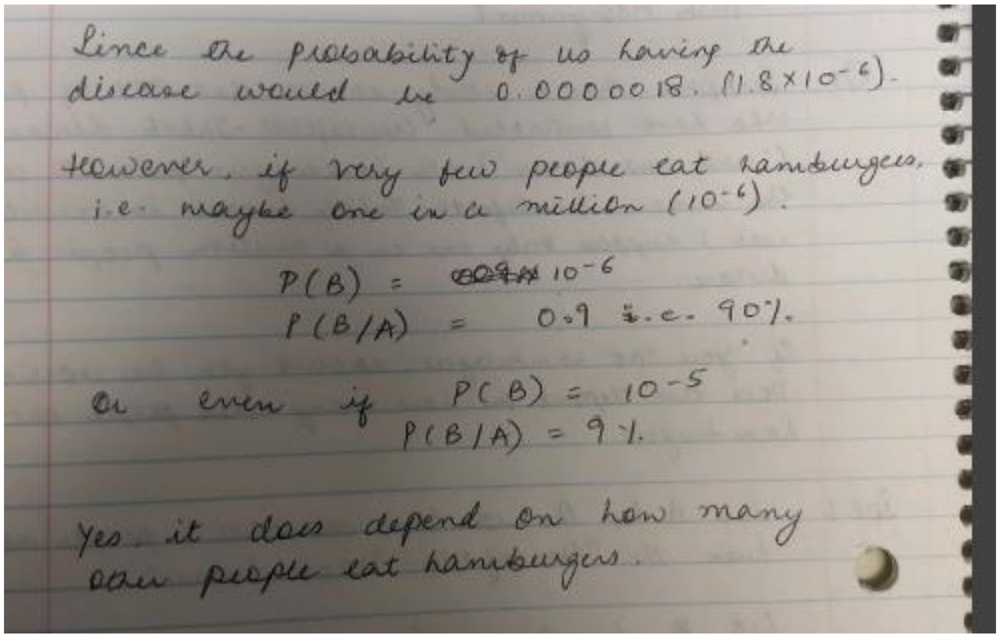
$$P(A) = 10^{-6}$$
$$P(A/B) = 0.9$$

$$P(B/A)\,P(B) = 0.9 \times 10^{-6}$$



If we should be worried or not, can be decided through P(B). Since we're assuming B to be random, the probability of someone having a hamburger would be 0.5.

$$P(B/A) = 1.8 \times 10^{-6}$$



Since the probability of us having the disease would be 0.0000018. (1.8×10^{-6}).

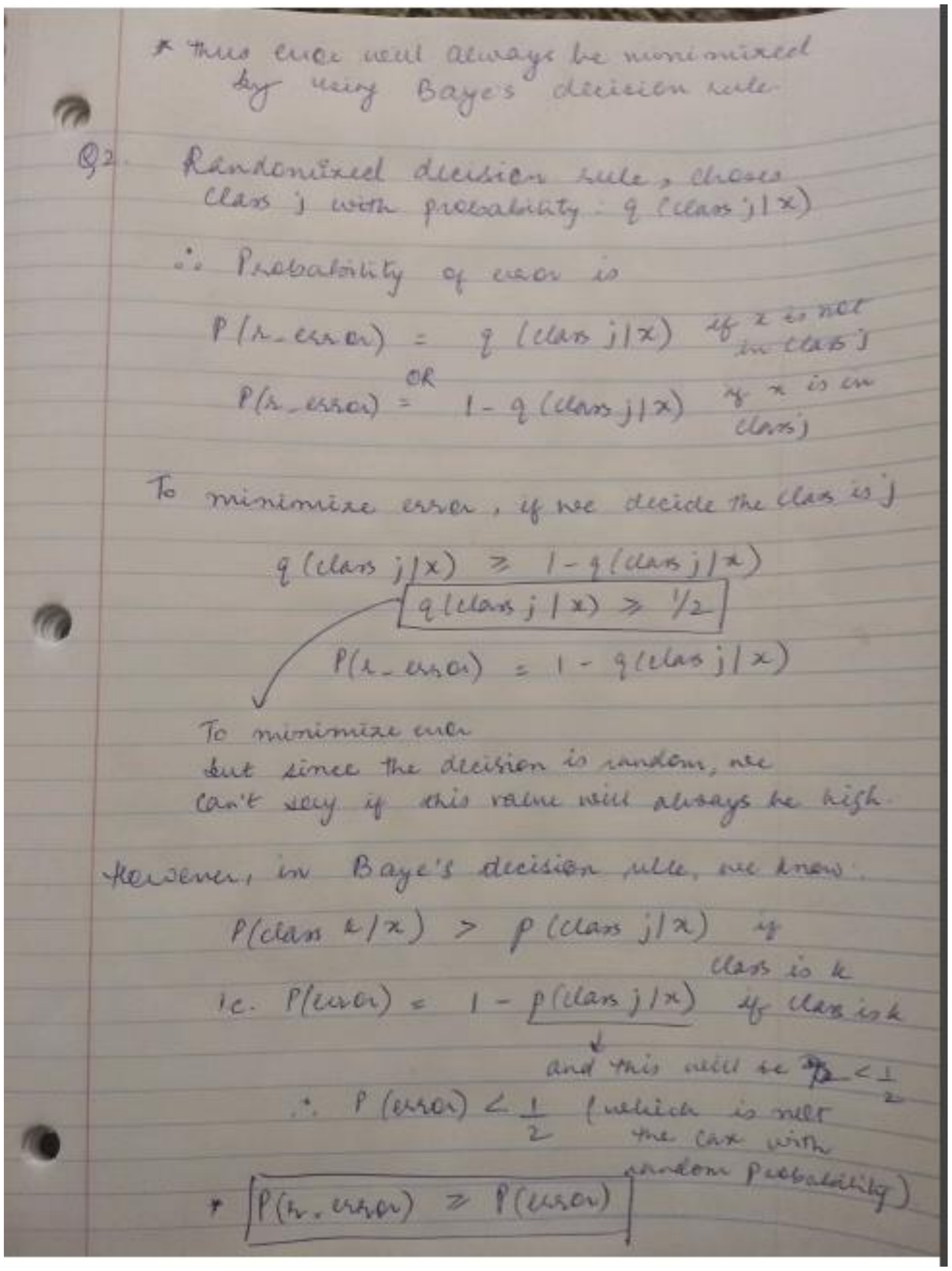
However, if very few people eat hamburgers, i.e. maybe one in a million (10^{-6}).

$$P(B) = ~~0.9\times~~ 10^{-6}$$
$$P(B/A) = 0.9 \text{ i.e. } 90\%.$$

Or even if $P(B) = 10^{-5}$
$P(B/A) = 9\%$.

Yes it does depend on how many other people eat hamburgers.

2.



* thus error will always be minimized by using Baye's decision rule

Q2. Randomized decision rule, choses class j with probability : $q(\text{class } j \mid x)$

∴ Probability of error is

$P(r\_error) = q(\text{class } j \mid x)$ if x is not in class j

OR

$P(r\_error) = 1 - q(\text{class } j \mid x)$ if x is in class j

To minimize error, if we decide the class is j

$$q(\text{class } j \mid x) \geq 1 - q(\text{class } j \mid x)$$

$$\boxed{q(\text{class } j \mid x) \geq 1/2}$$

$$P(r\_error) = 1 - q(\text{class } j \mid x)$$

To minimize error
but since the decision is random, we can't say if this value will always be high.

However, in Baye's decision rule, we know :

$P(\text{class } k / x) > p(\text{class } j \mid x)$ if class is k

i.e. $P(\text{error}) = 1 - \underline{p(\text{class } j \mid x)}$ if class is k

and this will be ~~[illegible]~~ $< \frac{1}{2}$

∴ $P(\text{error}) < \frac{1}{2}$ (which is not the case with random probability)

* $\boxed{P(r\_error) \geq P(\text{error})}$