Cara Evangeline Chandra Mohan
190539421

Data Mining - Lab Assignment 2

1.What are the new correctly classified instances, and the new confusion matrix? Briefly explain the reason behind your observation.

When k=4

	Actual Class = A	**Actual Class = B**
Predicted Class = A	661	216
Predicted Class = B	39	84

When k=999

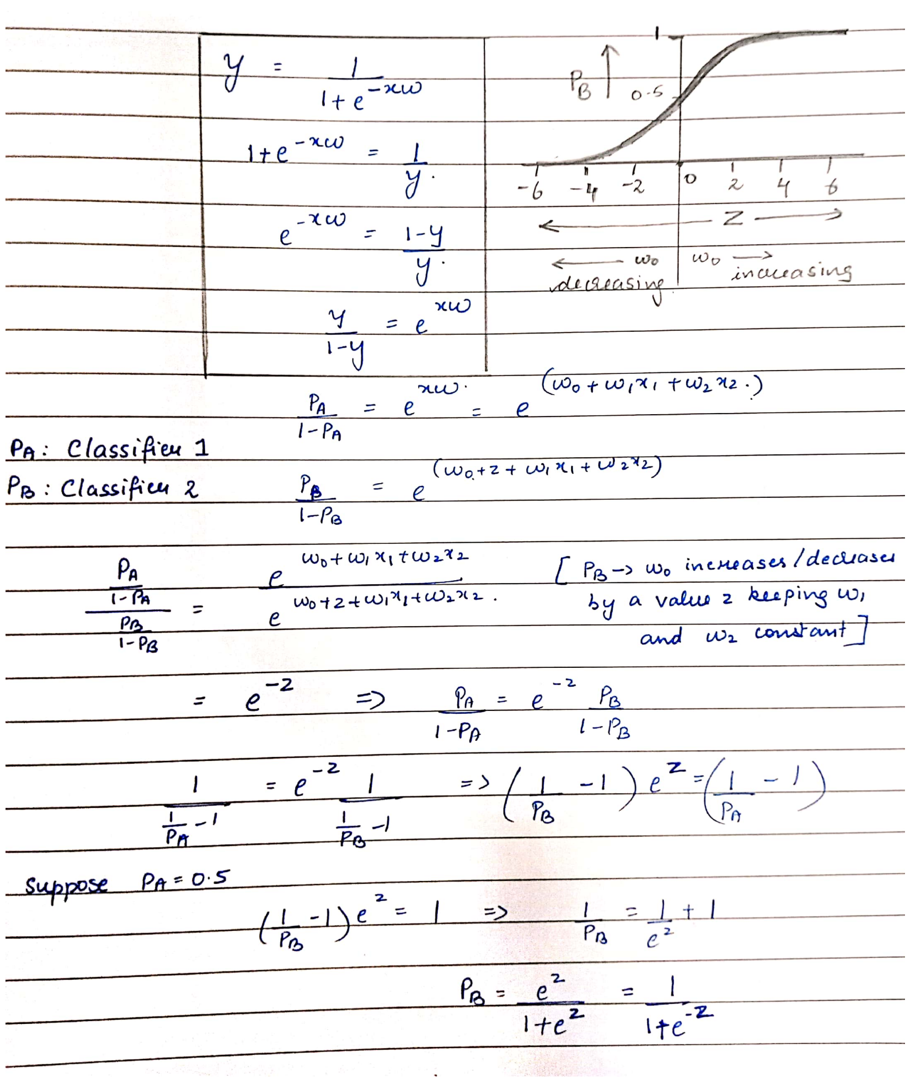
	Actual Class = A	**Actual Class = B**
Predicted Class = A	700	300
Predicted Class = B	0	0

From the above table we can understand that 39 new instances are classified correctly as Class A (ie.700-661 = 39). The reason behind this is: as seen in confusion matrix 1 and 2 we can clearly say that 700 tuples belong to class A which is the majority class and hence when k = 999 and for cross validation of 10-folds where in one iteration 900 tuples are training data and 100 tuples are test data, when every test tuple is applied the knn algorithm of k=999, it looks for 999 nearest neighbors and finds the majority of the classes to be class A and hence allocates every tuple Class A though it belongs to Class B.

2.Try (by varying the values of weights in the code), and describe the effect of the following actions on the classifier (with a brief explanation):

- **changing the first weight dimension, i.e., w_0?**
- **changing the second two parameters, w_1, w_2?**
- **negating all the weights?**
- **scaling the second two weights up or down (e.g. dividing or multiplying both w_1 and w_2 by 10)?**

Cara Evangeline Chandra Mohan
190539421


1
P_B
0.5
-6
-4
-2
0
2
4
6
Z
w_0 decreasing
w_0 increasing

$$y = \frac{1}{1+e^{-xw}}$$

$$1+e^{-xw} = \frac{1}{y}$$

$$e^{-xw} = \frac{1-y}{y}$$

$$\frac{y}{1-y} = e^{xw}$$

$$\frac{P_A}{1-P_A} = e^{xw} = e^{(w_0+w_1x_1+w_2x_2)}$$

P_A : Classifier 1
P_B : Classifier 2

$$\frac{P_B}{1-P_B} = e^{(w_0+z+w_1x_1+w_2x_2)}$$

$$\frac{\frac{P_A}{1-P_A}}{\frac{P_B}{1-P_B}} = \frac{e^{w_0+w_1x_1+w_2x_2}}{e^{w_0+z+w_1x_1+w_2x_2}}$$

[$P_B \rightarrow w_0$ increases/decreases by a value z keeping w_1 and w_2 constant]

$$= e^{-z} \Rightarrow \frac{P_A}{1-P_A} = e^{-z}\frac{P_B}{1-P_B}$$

$$\frac{1}{\frac{1}{P_A}-1} = e^{-z}\frac{1}{\frac{1}{P_B}-1} \Rightarrow \left(\frac{1}{P_B}-1\right)e^{z} = \left(\frac{1}{P_A}-1\right)$$

Suppose $P_A = 0.5$

$$\left(\frac{1}{P_B}-1\right)e^{z} = 1 \Rightarrow \frac{1}{P_B} = \frac{1}{e^{z}}+1$$

$$P_B = \frac{e^{z}}{1+e^{z}} = \frac{1}{1+e^{-z}}$$

Fig:- source: self

a) When w_0 is increased keeping w_1 and w_2 constant, the probability of class=1 (ie. P_B as shown in the above equation) for every combination of x_1 and x_2 increases. P_A be the classifier 1 for certain values of w_0, w_1 and w_2 and P_B be the classifier 2 for w_1, w_2 and w_0 increased/ decreased by z, and consider P_A to be 0.5 then from the graph given in figure we can see that P_B is increasing for increasing w_0 (ie. z is positive) and P_B is decreasing for decreasing w_0 (ie. z is negative). In short we can say that increasing/decreasing w_0 increases/decreases the probability of a tuple being classified as Label 1.

b) When w_1 and w_2 increases/decreases, then as shown in the equation in above figure, z is replaced with $f(x_1,x_2) = z_1x_1 + z_2x_2$ (where z_1 is the increase/decrease in w_1 and z_2 is the increase/decrease in w_2 and x_1, x_2 are the attribute values). Here the equation $f(x_1,x_2)$ is a linear equation in x_1 and x_2 and hence the slope separating the classes varies as w_1 and w_2 varies

c) Negating the values of w_0, w_1, w_2 changes the class labels vice versa since the probability of getting a class label is a Log function and hence gives a lesser accuracy.

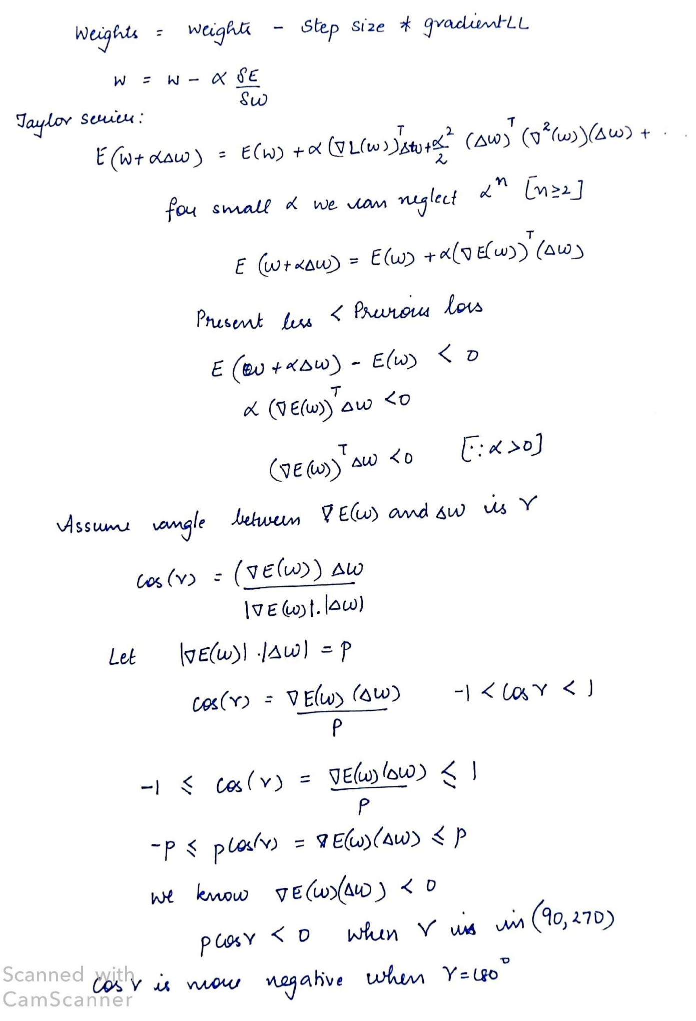
Cara Evangeline Chandra Mohan
190539421

d) Dividing/multiplying w_1 and w_2 by 10 differentiates clearly the boundary between the two classes

3.(a) How many times did your loop execute? (b) Report your classifier weights that first get you at least 75% train accuracy.

The loop executes for 8000 iterations
The classifier weights that first give at least 75% train accuracy is: [-0.579 1.000 0.789]

4.(a) Notice that we used the negative of the gradient to update the weights. Explain why. (b) Given the gradient at the new point, determine (in terms of "increase" or "decrease", no need to give values), what we should do to each of w_0, w_1 and w_2 to further improve the loss function.



Weights = weights − step size * gradientLL

$$W = W - \alpha \frac{\delta E}{\delta w}$$

Taylor series:

$$E(W+\alpha\Delta w) = E(W) + \alpha(\nabla L(w))^T\Delta w + \frac{\alpha^2}{2}(\Delta w)^T(\nabla^2(w))(\Delta w) + \cdots$$

for small α we can neglect α^n $[n \geq 2]$

$$E(w+\alpha\Delta w) = E(w) + \alpha(\nabla E(w))^T(\Delta w)$$

Present loss < Previous loss

$$E(w+\alpha\Delta w) - E(w) < 0$$

$$\alpha(\nabla E(w))^T\Delta w < 0$$

$$(\nabla E(w))^T\Delta w < 0 \quad [\because \alpha > 0]$$

Assume angle between $\nabla E(w)$ and Δw is γ

$$\cos(\gamma) = \frac{(\nabla E(w))\Delta w}{|\nabla E(w)|.|\Delta w|}$$

Let $|\nabla E(w)| \cdot |\Delta w| = p$

$$\cos(\gamma) = \frac{\nabla E(w)(\Delta w)}{p} \qquad -1 < \cos\gamma < 1$$

$$-1 \leq \cos(\gamma) = \frac{\nabla E(w)(\Delta w)}{p} \leq 1$$

$$-p \leq p\cos(\gamma) = \nabla E(w)(\Delta w) \leq p$$

we know $\nabla E(w)(\Delta w) < 0$

$p\cos\gamma < 0$ when γ is in $(90, 270)$

$\cos\gamma$ is more negative when $\gamma = 180^\circ$

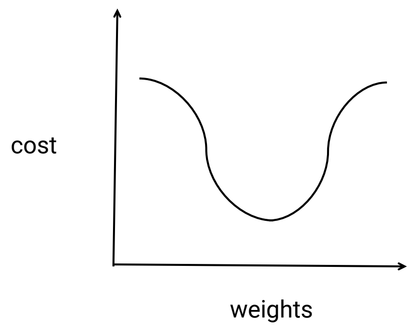
Fig:- source: https://medium.com/datadriveninvestor/why-we-move-our-weights-in-opposite-direction-of-gradients-565679ff9320

Cara Evangeline Chandra Mohan
190539421

a) As the equations in the above figure describe, we should move weight in the opposite direction of gradient descent and hence the negative of the gradient is used to update weight.

b) Given gradient descent at a point, we can't conclude any decision about w_0, w_1, w_2 because it depends upon the number of tuples belonging to a class, the loss function used and the number of data points.

5.Suppose we wanted to turn this step into an algorithm that sequentially takes such steps to get to the optimal solution. What is the effect of "step-size"? Your answer should be in terms of the trade-offs, i.e., pros and cos, in choosing a big value for step-size versus a small value for step-size.



To find the optimal weights we move towards the minimum cost, so here we use the learning rate (step size) to update the values of weight. How we choose the learning rate matters because convergence to minima is important here.

<u>Smaller step size:</u>

- Pros: Very less probability of skipping any local minima
- Cons: The process is too slow

<u>Larger step size:</u>

- Pros: Requires less training iterations
- Cons: Might fail to converge at local minima

Cara Evangeline Chandra Mohan
190539421

6.Answer this question either by doing a bit of research, or looking at some source codes, or thinking about them yourself: (a) Argue why choosing a constant value for step_size, whether big or small, is not a good idea. A better idea should be to change the value of step-size along the way. Explain whether the general rule should be to decrease the value of step-size or increase it. (b) Come up with the stopping rule of the gradient descent algorithm which was described from a high level at the start of this section. That is, come up with (at least two) rules that if either one of them is reached, the algorithm should stop and report the results.

a) Choosing a constant step-size can either slows the process or misses the convergence point, so it is better to keep changing the step-size as we proceed with further iterations. Depending upon the current point's distance with convergence (ie. minima) we need to choose the value of step-size. For every iteration we need to calculate the error, if increasing step-size increases the error than the previous iteration's error then we need to decrease the step-size and move backwards. When we are near to the minima, it is better to choose a smaller step-size and when we are farther, bigger value of step-size is preferred.

b) **Rule 1**: Say suppose convergence occurs at iteration 7, then the value of error at iteration 6 and 8 will be greater than at 7 and hence we need to stop the process at iteration 8
Note: If the loss function is Mean Square Error
Rule 2: Stop when gradient descent is approaching zero

7.Provide one advantage and one disadvantage of using exhaustive search for optimisation, especially in the context of training a model in machine learning.

Advantage:

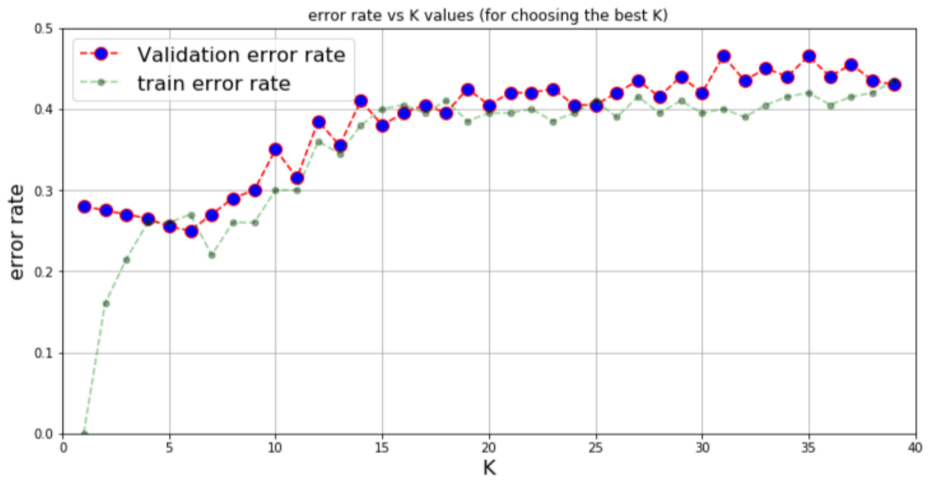
Every possible combination of weights is tried and this method is very accurate to find the model which best fits and gives lesser cost function

Disadvantage:

It's an inefficient approach to try all combinations particularly it takes a lot of time to compute all models.

Cara Evangeline Chandra Mohan
190539421

8.(a) What is the best value for K? Explain how you got to this solution. (b) Interpret the main trends in the plot. In partiular, specify which parts correspond to under-fitting and which part with over-fitting.



According to general thumb rule k=sqrt(N)/2 where N is the total number of training samples. In our example we have N=200, so k is approx 7.
As shown in the above graph, we have minimum validation error at k=6 which is close to k=7 and hence k=6 is the best value
When k=1, overfitting occurs because the train error rate is 0 and the model is overestimated.
When k=40, underfitting occurs because the model considers 40 neighbourhoods and not estimated correctly.

9.Given what you learned from this entire lab(!), provide one advantage and one disadvantage of using MaxEnt vs KNN.

MaxEnt:

Advantage:
- Gives the probability of an instance belonging to a particular class
- Faster than KNN

Disadvantage:
- Linear classification problem and cannot be applied on non-linear dataset

KNN:

Advantage:
- Can handle Non-linear problems

Disadvantage:
- Slower than MaxEnt
- Choice of K must be made properly