EE 511 Homework Assignment 3 Solutions

1. In this classification problem we have to predict the cause of power outage

Sr. no.	Cause of outage (Y)
1	Weather,
2	Bird/animal,
3	Vehicle accident (e.g. truck hits a telephone pole)
4	Equipment failure
5	Other reason

Input Vectors (X)	Attributes continuous or discrete	Number of Features
Measurement	Continuous: temperature, humidity, barometric pressure, wind speed	4
Wind direction	Discrete: North(N), North East(NE), East(E), South East(SE), South(S), South West(SW), West(W), North West(NW)	8
Rainfall	Continuous: In inches for past 2 days	1
Time	Discrete: Month, day of the week, time of the outage (assuming, we are representing time as hhmm)	3

(a) Let's assume that no one has ever worked on this problem before. What would you use as a baseline?

Solution: As a baseline we can use the K-Nearest neighbor model to predict the cause of power outage. The K-Nearest neighbor model is simple and easy to implement, in this model we do not need to make any assumptions.

(b) If you designed a classier based on a generative model, what distributions would you use? How many free parameters are in your model?

Solution: In this problem we can use *Gamma distribution*, which is a two-parameter family of continuous probability distributions. The probability density function for Gamma distribution is given by:

$$f(x) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}}$$

Here θ is the scale parameter and k is the shape parameter, two free parameters.

In this conditional distribution of power outage time x is a function of a set of variables for example in our case it will be time of power outage, wind speed, humidity etc. In many applications, wait time can be easily calculated using *Gamma distribution*.

(c) If you used logistic regression, what would the dimensionality of your feature vector be?
Solution: We will be using one hot encoder to convert the discrete value of the wind direction attributes(8) and adding to the rest of the features . In total the feature vector will be16.

(d) Explain how you might control the complexity of each model.
Solution: The complexity of the model can be controlled by using regularization hyperparameter, which is a technique to avoid overfitting by adding the penalty term in the error function. Two most popular regularisation methods are lasso and ridge regression which can be used. In both methods tuning parameters are added to the model, based on the size or effect on the model, these models impose the penalty on each term's coefficient size. We can also further use cross validation techniques for evaluating the one technique vs other and then selecting one of the models based on the smallest test error.

Solution 2
Given in the problem
$y_i = -1 : (-1, 5), (-1, 2), (2, 5)$ $y_i = 1 : (0, -2), (1, 0), (2, 1)$
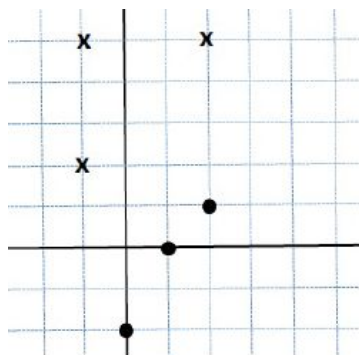
Solution 2 Given $y_i = -1 : (-1,5), (-1,2)(2,5)$
$y_i = 1 : (0,-2), (1,0), (2,1)$

2(a) By looking at the graph, we can see that the data is linearly seperable, therefore we can use the linear SVM to find the equation.

By inspecting the graph above we can see that there are 4 support vectors mentioned below.

$$S_1 = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 2 \\ 5 \end{bmatrix} \qquad S_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad S_4 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Vector augmented with 1 as a biqe,

$$S_1 = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} \qquad S_3 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \qquad S_4 = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$$

from the above vectors we will have 4 equations.

$$\alpha_1 \, \tilde{S}_1 \cdot \tilde{S}_1 + \alpha_2 \, \tilde{S}_2 \cdot \tilde{S}_1 + \alpha_3 \, \tilde{S}_3 \cdot \tilde{S}_1 + \alpha_4 \, \tilde{S}_4 \cdot \tilde{S}_1 = -1 \quad \text{——①}$$

$$\alpha_1 \, \tilde{S}_1 \cdot \tilde{S}_2 + \alpha_2 \, \tilde{S}_2 \cdot \tilde{S}_2 + \alpha_3 \, \tilde{S}_3 \cdot \tilde{S}_2 + \alpha_4 \, \tilde{S}_4 \cdot \tilde{S}_2 = -1 \quad \text{——②}$$

$$\alpha_1 \, \tilde{S}_1 \cdot \tilde{S}_3 + \alpha_2 \, \tilde{S}_2 \cdot \tilde{S}_3 + \alpha_3 \, \tilde{S}_3 \cdot \tilde{S}_3 + \alpha_4 \, \tilde{S}_4 \cdot \tilde{S}_3 = 1 \quad \text{——③}$$

$$\alpha_1 \, \tilde{S}_1 \cdot \tilde{S}_4 + \alpha_2 \, \tilde{S}_2 \cdot \tilde{S}_4 + \alpha_3 \, \tilde{S}_3 \cdot \tilde{S}_4 + \alpha_4 \, \tilde{S}_4 \cdot \tilde{S}_4 = 1 \quad \text{——④}$$

Putting value of vectors in equation ①

$$\alpha_1 \tilde{S}_1 \cdot \tilde{S}_1 + \alpha_2 \tilde{S}_2 \cdot \tilde{S}_1 + \alpha_3 \tilde{S}_3 \tilde{S}_1 + \alpha_4 \tilde{S}_4 \cdot \tilde{S}_1 = -1$$

$$\alpha_1 \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix} = -1$$

$$\alpha_1 [1+4+1] + \alpha_2 [-2+10+1] + \alpha_3 [-1+0+1] +$$
$$\alpha_4 [-2+2+1] = -1$$

$$\boxed{6\alpha_1 + 9\alpha_2 + \alpha_4 = -1}$$

Putting value of vectors in equation ②

$$\alpha_1 \tilde{S}_1 \tilde{S}_2 + \alpha_2 \tilde{S}_2 \tilde{S}_2 + \alpha_3 \tilde{S}_3 \tilde{S}_2 + \alpha_4 \tilde{S}_4 \tilde{S}_2 = -1$$

$$\alpha_1 \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} = -1$$

$$\alpha_1 [-2+10+1] + \alpha_2 [4+10+2] + \alpha_3 [2+0+2] +$$
$$\alpha_4 [4+5+1] = -1$$

$$\boxed{9\alpha_1 + 16\alpha_2 + 3\alpha_3 + 10\alpha_4 = -1}$$

Putting the vectors in equation ③

$$\alpha_1 \hat{s}_1 \tilde{s}_3 + \alpha_2 \check{s}_2 \tilde{s}_3 + \alpha_3 \check{s}_3 \tilde{s}_3 + \alpha_4 \hat{s}_4 \check{s}_3 = 1$$

$$\alpha_1 \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = 1$$

$$\alpha_1 \begin{bmatrix} -1 + 0 + 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 + 0 + 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 1 + 0 + 1 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 2 + 0 + 1 \end{bmatrix} = 1$$

$$\boxed{3\alpha_2 + 2\alpha_3 + 3\alpha_4 = 1}$$

Putting the vectors in equation ④

$$\alpha_1 \check{s}_1 \hat{s}_4 + \alpha_2 \check{s}_2 \tilde{s}_4 + \alpha_3 \check{s}_3 \hat{s}_4 + \alpha_4 \hat{s}_4 \check{s}_4 = 1$$

$$\alpha_1 \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} = 1$$

$$\alpha_1 \begin{bmatrix} -2 + 2 + 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 4 + 5 + 1 \end{bmatrix} + \alpha_3 \begin{bmatrix} 2 + 0 + 1 \end{bmatrix} +$$

$$\alpha_4 \begin{bmatrix} 4 + 1 + 1 \end{bmatrix} = 1$$

$$\boxed{\alpha_1 + 10\alpha_2 + 3\alpha_3 + 6\alpha_4 = 1}$$

We got 4 equations in terms of α

$$6\alpha_1 + 9\alpha_2 + \alpha_4 = -1$$

$$9\alpha_1 + 16\alpha_2 + 3\alpha_3 + 10\alpha_4 = -1$$

$$3\alpha_2 + 2\alpha_3 + 3\alpha_4 = 1$$

$$\alpha_1 + 10\alpha_2 + 3\alpha_3 + 6\alpha_4 = 1$$

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} \begin{bmatrix} 6 & 9 & 0 & 1 \\ 9 & 16 & 3 & 10 \\ 0 & 3 & 2 & 3 \\ 1 & 10 & 3 & 6 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

$$\alpha_1 = -\frac{1}{8} = -0.125$$

$$\alpha_2 = 0$$

$$\alpha_3 = \frac{7}{8} = 0.875$$

$$\alpha_4 = -\frac{1}{4} = 0.25$$

Now putting in $\hat{w} = \sum\limits_{9} \alpha_p \ddot{S}_i$

$$= \sum\limits_{9} (\alpha_1 \ddot{S}_1 + \alpha_2 \ddot{S}_2 + \alpha_3 \ddot{S}_3 + \alpha_4 \ddot{S}_4)$$
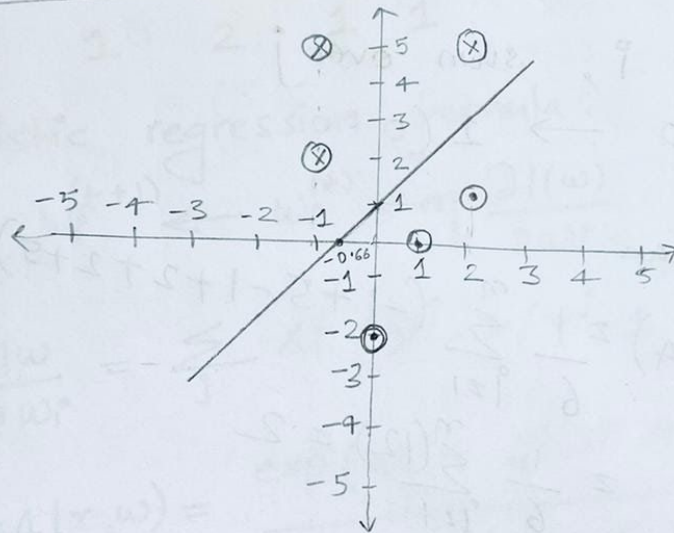
$$= -0.125 \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix} + 0 + 0.875 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 0.25 \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.125 \\ 0.25 \\ -0.125 \end{bmatrix} + \begin{bmatrix} 0.875 \\ 0 \\ 0.875 \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix} \Rightarrow \begin{bmatrix} 1.5 \\ 0.5 \\ 1 \end{bmatrix}$$

③

$$y = \omega x + b$$

$$\omega = \begin{bmatrix} 1.5 \\ 0.5 \end{bmatrix} \quad \text{and} \quad b = 1$$

$$\boxed{y = \begin{bmatrix} 1.5 \\ 0.5 \end{bmatrix} x + 1} \Rightarrow \begin{array}{l} y = 1.5x + 1 \\ y = 0.5x + 1 \end{array}$$

when $y = 0 \rightarrow x = -1/1.5 = -0.6$

$\quad\quad\quad y = 0 \rightarrow x = -1/0.5 = -2$

when $x = 0 \rightarrow y = 1$

2(b)

	X_0	X_1	X_2	Y
$j=0$	1	-1	5	0
$j=1$	1	-1	2	0
$j=2$	1	2	5	0
$j=3$	1	0	-2	1
$j=4$	1	1	0	1
$j=5$	1	2	1	1

Logistic regression formula:

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(w)}{\partial w_i}$$

$$\frac{\partial l w}{\partial w_i} = \sum_j x_i^j \left(y^j - P(Y^j = 1 \mid x^j, w) \right)$$

$$P(Y=1 \mid x, w) = \frac{\exp\left(w_0 + \sum_i w_i x_i \right)}{1 + \exp\left(w_0 + \sum_i w_i x_i \right)}$$

$w = (0, 0, 0)$

$x_0^j = 1$

$t = 0$

Calculating $\exp\left(w_0 + \sum_i w_i x_i \right)$ then $P(Y=1 \mid x, w)$

$\exp(0 + 0 \times (-1) + 0 \times 5) = 1 \rightarrow P(Y^0 = 1 \mid x^0, w) = 0.5$

$\exp(0 + 0 \times (-1) + 0 \times 2) = 1 \rightarrow P(Y^1 = 1 \mid x^1, w) = 0.5$

All probabilities will be 0.5

Loop over i, sum over j ($t = 1$)

$$i = 0, j = 0 \rightarrow \sum_j x_0^o (Y^o - P(Y = 1 | x^o, w)$$
$$= 1(0 - 0.5) = -0.5$$

Similarly,
$$i = 0, j = 1 \rightarrow 1(0 - 0.5) = -0.5$$
$$i = 0, j = 2 \rightarrow 1(0 - 0.5) = -0.5$$
$$i = 0, j = 3 \rightarrow 1(1 - 0.5) = 0.5$$
$$i = 0, j = 4 \rightarrow 1(1 - 0.5) = 0.5$$
$$i = 0, j = 5 \rightarrow 1(1 - 0.5) = 0.5$$

when $i = 1, j = 0 \rightarrow \sum_j x_1^o (Y^o - P(Y = 1 | x^o, w))$
$$= -1(0 - 0.5) = 0.5$$
$$i = 1, j = 1 \rightarrow -1(0 - 0.5) = 0.5$$
$$i = 1, j = 2 \rightarrow 2(0 - 0.5) = -1$$
$$i = 1, j = 3 \rightarrow 0(1 - 0.5) = 0$$
$$i = 1, j = 4 \rightarrow 1(1 - 0.5) = 0.5$$
$$i = 1, j = 5 \rightarrow 2(1 - 0.5) = 1$$

when $i = 2, j = 0 \rightarrow \sum_j x_2^o (Y^o - P(Y = 1 | x^o, w)$
$$\Rightarrow 5(0 - 0.5) = -2.5$$
$$i = 2, j = 1 \rightarrow 2(0 - 0.5) = -1$$
$$i = 2, j = 2 \rightarrow 5(0 - 0.5) = -2.5$$
$$i = 2, j = 3 \rightarrow -2(1 - 0.5) = -1$$

$$i = 2, j = 4 \rightarrow 0(1 - 0.5) = 0$$
$$i = 2, j = 5 \rightarrow 1(1 - 0.5) = 0.5$$
$$grad = [0, 1.5, -6.5]$$

given $\eta = 0.1$, putting into formula

$$w_i^{t+1} \leftarrow w_i^{(t)} + \eta \frac{\partial l(w)}{\partial w_i}$$

$$w_i^{t+1} \leftarrow [0,0,0] + 0.1 \times [0, 1.5, -6.5]$$

$$\boxed{w_i^{t+1} \leftarrow [0, 0.15, -0.65]}$$

For itteration 2 $(t = 1)$

calculating $\exp(w_0 + \sum_i w_i x_i)$ then $P(Y=1|x, w)$

$\rightarrow \exp(0 + 0.15 \times (-1) + (-0.65) 5) = \exp(3.4) = 0.033$

$$Probability = \frac{0.033}{1 + 0.033} = 0.0319$$

$\rightarrow \exp(0 + 0.15 \times (-1) + (-0.65) \times 2) = \exp(-1.45) = 0.234$

$$Probability = \frac{0.234}{1 + 0.234} = 0.186$$

$\rightarrow \exp(0 + 0.15 \times 2 + (-0.65) \times 5) = \exp(-2.95) = 0.052$

$$Probability = \frac{0.052}{1 + 0.052} = 0.047$$

$\longrightarrow \exp(0 + 0.15 \times 0 + (-0.5)(-2)) = \exp(-13) = 3.669$

$\qquad\qquad\qquad (-0.65) \qquad\qquad = 3.66$

$\qquad$ Probability $= 0.7858$

$\longrightarrow \exp(0 + 0.15 \times 1 + 0) = \exp(0.15) = 1.161$

$\qquad$ Probability $= 0.537$

$\longrightarrow \exp(0 + 0.15 \times 2 + (-0.65) \times 1) = \exp(-0.35)$

$\qquad\qquad\qquad\qquad\qquad\qquad = 0.704$

$\qquad$ Probability $= 0.413$

Loop over i, sum over j

$\quad i = 0, j = 0 \rightarrow 1(0 - 0.031) = -0.031$

$\quad i = 0, j = 1 \rightarrow 1(0 - 0.186) = -0.186$

$\quad i = 0, j = 2 \rightarrow 1(0 - 0.047) = -0.047$

$\quad i = 0, j = 3 \rightarrow 1(1 - 0.78) = 0.214$

$\quad i = 0, j = 4 \rightarrow 1(1 - 0.53) = 0.47$

$\quad i = 0, j = 5 \rightarrow 1(1 - 0.41) = 0.59$

Now, $i = 1, j = 0 \rightarrow -1(0 - 0.031) = 0.031$

$\quad i = 1, j = 1 \rightarrow -1(0 - 0.186) = 0.186$

$\quad i = 1, j = 2 \rightarrow 2(0 - 0.047) = -0.094$

$\quad i = 1, j = 3 \rightarrow 0(1 - 0.78) = 0$

$\quad i = 1, j = 4 \rightarrow 1(1 - 0.53) = 0.47$

$\quad i = 1, j = 5 \rightarrow 2(1 - 0.41) = 1.18$

Now, $i = 2, j = 0 \rightarrow 5(0 - 0.031) = -0.155$

$\quad i = 2, j = 1 \rightarrow 2(0 - 0.186) = -0.372$

$i = 2, j = 2 \rightarrow 5(0 - 0.047) = -0.235$

$i = 2, j = 3 \rightarrow -2(1 - 0.78) = -0.428$

$i = 2, j = 4 \rightarrow 0(1 - 0.53) = 0$

$i = 2, j = 5 \rightarrow 1(1 - 0.41) = 0.59$

$grad = (0.9913, 1.758, -0.631)$

$w^{t+1} = [0, 0.15, -0.65] + 0.1 \times [0.9913, 1.758, -0.631]$

$\boxed{w^{t+1} = [0.09913, 0.32586, -0.71319]}$

For itteration 3 : $t = 2$

$\rightarrow exp(0.099 + 0.325 \times (-1) + (-0.713) \times 5) = 0.0225$

$\quad exp~to$ Probability $= 0.02204$

$exp(0.099 + 0.325 \times (-1) + (-0.713) \times 2) = 0.191$

$\quad$ Probability $= 0.160$

$exp(0.099 + 0.325 \times (2) + (-0.713) \times 5) = 0.059$

$\quad$ Probability $= 0.056$

$exp(0.099 + 0.325 \times (0) + (-0.713) \times (-2) = 4.597$

$\quad$ Probability $= 0.82135$

$exp(0.099 + 0.325 \times 0 + (-0.713) \times 1) = 1.529$

$\quad$ Probability $= 0.6046$

$exp(0.099 + 0.325 \times 2 + (-0.86) \times 1) = 1.038$

$\quad$ Probability $= 0.50941$

For $i=0, j=0 \rightarrow$ $1(0 - 0.02204) = -0.02204$

$i=0, j=1 \rightarrow$ $1(0 - 0.16069) = -0.16069$

$i=0, j=2 \rightarrow$ $1(0 - 0.05651) = -0.5651$

$i=0, j=3 \rightarrow$ $1(1 - 0.82135) = 0.17865$

$i=0, j=4 \rightarrow$ $1(1 - 0.60468) = 0.39532$

$i=0, j=5 \rightarrow$ $1(1 - 0.50941) = 0.49059$

$i=1, j=0 \rightarrow$ $-1(0 - 0.02204) = 0.02204$

$i=1, j=1 \rightarrow$ $-1(0 - 0.16069) = 0.16069$

$i=1, j=2 \rightarrow$ $2(0 - 0.05651) = -0.11302$

$i=1, j=3 \rightarrow$ $0(1 - 0.82135) = 0$

$i=1, j=4 \rightarrow$ $1(1 - 0.60468) = 0.39532$

$i=1, j=5 \rightarrow$ $2(1 - 0.50941) = 0.98118$

$i=2, j=0 \rightarrow$ $5(0 - 0.02204) = -0.1102$

$i=2, j=1 \rightarrow$ $2(0 - 0.16069) = -0.32138$

$i=2, j=2 \rightarrow$ $5(0 - 0.05651) = -0.28255$

$i=2, j=3 \rightarrow$ $-2(1 - 0.82135) = -0.3573$

$i=2, j=4 \rightarrow$ $0(1 - 0.60468) = 0$

$i=2, j=5 \rightarrow$ $1(1 - 0.50941) = 0.49059$

η (learning rate) $= 0.1$

$w^{(t+1)} = [0.09913, 0.32586, -0.71519] + 0.1 \times [0.82532, 1.44621, -0.58084]$
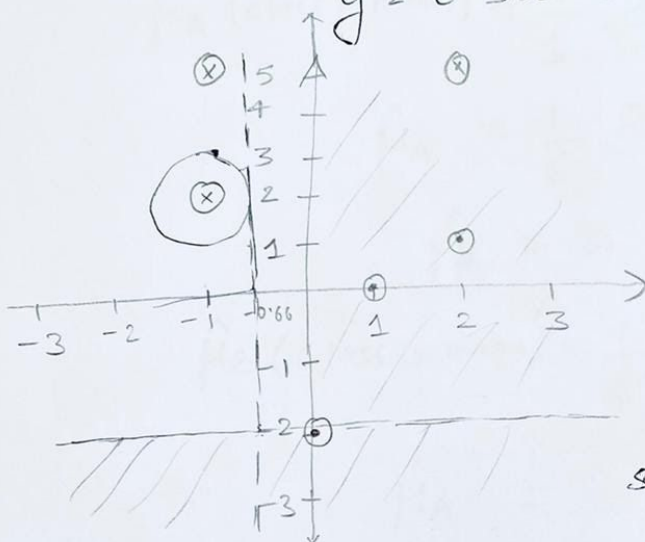
$$\boxed{w^{(t+1)} = [0.1866, 0.47048, -0.77127]}$$

2(c) 1- Nearest neighbor classifier

from part 2(a) we have

$$y = 1.5x + 1 \implies x = -0.66 \text{ when } y = 0$$

$$y = 0.5x + 1 \implies x = -2 \text{ when } y = 0$$

testing distance

$$d(x^a, x^b) = \sqrt{\sum_{j=1}^{n} (-0.66 - (-1))^2}$$

$$= \sqrt{(-0.66 + 1)^2}$$

$$= 0.344$$

so, the nearest neighbor
is at $(-1, 2)$

2 (d) To calculate the mean formula used

is $\hat{\mu} = \dfrac{1}{n} \sum\limits_{i=1}^{n} x_i$

$\hat{\mu}_A$ (class A mean) $= \dfrac{1}{6} \sum\limits_{i=1}^{n} (-1+5-1+2+2+5)$

$\hat{\mu}_A = \dfrac{1}{6} \sum\limits_{i=1}^{n} (12)$

$\hat{\mu}_A = 2$

$\hat{\mu}_B$ (Class B mean) $= \dfrac{1}{6} \sum\limits_{i=1}^{n} (0-2+1+0+2+1)$

$\hat{\mu}_B = \dfrac{1}{6} (2) = \dfrac{1}{3}$

$\hat{\mu}_B = 0.33$

Now, to find the decision function we
calculate posterior probability. Standard Variance = 1.

$\boxed{P(X_i \neq | Y = y_k) = \dfrac{1}{\sqrt{2\pi \sigma_{ik}^2}} \, e^{-\frac{1}{2}\left(\frac{x - \mu_{ik}}{\sigma_{ik}}\right)^2}}$

$P(\text{class A}) = \dfrac{1}{\sqrt{2\pi}} \, e^{-\frac{1}{2}\left(\frac{12-2}{1}\right)^2} = 0.76$

$P(\text{class B}) = \dfrac{1}{\sqrt{2\pi}} \, e^{-\frac{1}{2}\left(\frac{2-0.33}{1}\right)} = 0.18$

Part II: Computer Assignment Summary

I used CountVectorizer to represent the documents as a bag of words with occurrence counting. I found 73686 words in the training data and ruled out the ones which occured only a few times (<=10). I left with 15226 words for which I computed mutual information and selected the top 5000 words. I represented my dataset using log-normalized counts where each entry becomes log(td +1). I used LinearSVC classifier which handles multiclass support using one-vs-the-rest scheme. I did a cross-validation for 2 classifiers and found the one with L2 penalty performing better. I thenevaluate by best classifier using test data and found 74% accuracy. I checked five largest outliers in the confusion matrix and noticed that our classifier is predicting wrong classes where they are similar in nature (for ex. predicting talk.politics.guns instead of talk.politics.misc). I also checked top10 and bottom 10 features for each class, and cross-checked them with our mutual information based selected features.

The top 10 and bottom 10 features against each class are inlined with what we found using mutual information. All the top 10 features are in our list of top 5000 words as per mutual information**.**

Neha_HW 3_Part II: Computer Assignment Solution

I used CountVectorizer to represent the documents as bag of words with occurance counting. I found 73686 words in the training data and ruled out the ones which occured only a few times (<= 10). I left with 15226 words for which I computed mutual information and selected the top 5000 words. I represented my dataset using log-normalized counts where each entry becomes log(td + 1). I used LinearSVC classifier which handles multiclass support using one-vs-the-rest scheme. I did a cross-validation for 2 classifiers and found the one with L2 penality performing better. I then evaluate by best classifier using test data and found 74% accuracy. I checked five largest outliers in the confusion matrix and noticed that our classifier is predicting wrong class where they are similar in nature (for ex. predicting talk.politics.guns instead of talk.politics.misc). I also checked top 10 and bottom 10 features for each class, and cross-checked them with our mutual information based selected features.

```
In [1]: import numpy as np
        import pandas as pd
        import math
```

(a) Setup - loading training data

```
In [2]: df = pd.read_csv('20ng-train-all-terms.csv', names=['class_publication_name
        df.head()
```

Out[2]:

	class_publication_name	document
0	alt.atheism	alt atheism faq atheist resources archive name...
1	alt.atheism	alt atheism faq introduction to atheism archiv...
2	alt.atheism	re gospel dating in article mimsy umd edu mang...
3	alt.atheism	re university violating separation of church s...
4	alt.atheism	re soc motss et al princeton axes matching fun...

(b) Vocabulary Selection - coverting the given document text into features (word) using CountVectorizer

```
In [3]: rom sklearn.feature_extraction.text import CountVectorizer
        ectorizer = CountVectorizer()
         = vectorizer.fit_transform(df.document.to_numpy())
        ord_count_df = pd.DataFrame(X.toarray(), columns = vectorizer.get_feature_n
        ord_count_df.head()
```

Out[3]:

	aa	aaa	aaaa	aaaaaaaaaaaa	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaauuuuuuuuuuuuuuuuuuuuuuuu
0	0	0	0	0	
1	0	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	0	

5 rows × 73686 columns

(b) Vocabulary Selection - ruling out words that occur only a few times (i.e. word_count <= 10 in the entire corpus)

```
In [4]: uent_word_df = pd.DataFrame()
        i in range(0, word_count_df.shape[1]):
        if(word_count_df.iloc[:,i].sum() > 10):
            frequent_word_df[word_count_df.iloc[:,i].name] = word_count_df.iloc[:,i]
        uent_word_df.head()
```

Out[4]:

	aa	aaa	aamir	aardvark	aaron	aas	aau	ab	abandon	abandoned	...	zoology	zoom	zoroa
0	0	0	0	0	0	0	0	0	0	0	...	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	

5 rows × 15226 columns

(b) Vocabulary Selection - selecting top 5000 words by mutual information

```python
In [5]: from sklearn.feature_selection import mutual_info_classif

        X_train = frequent_word_df
        Y_train = df.class_publication_name
        top_features = []
        feature_to_mi = {}

        mutual_info = mutual_info_classif(X_train, Y_train, discrete_features=True)

        for i in range(0,len(mutual_info)):
            feature_to_mi[frequent_word_df.columns.values[i]] = mutual_info[i]
        i = 0;
        for k in sorted(feature_to_mi, key=feature_to_mi.get, reverse=True):
            if(i==5000):
                break
            else:
                top_features.append(k)
                i = i+1
```

(b) Vocabulary Selection - table of the top ten words by their mutual information

```python
In [6]: print("Top 10\t", "Mutual Information")
        for i in range (0, 10):
            print(top_features[i],"\t", feature_to_mi[top_features[i]])
```

```
Top 10    Mutual Information
the       0.17974621123708354
of        0.1701979105305262
that      0.14461294724277796
in        0.1302655176431355
to        0.12813250389059153
god       0.11832752440098114
windows          0.11666172137334153
and       0.10945370613424468
is        0.10852105536891804
he        0.10115197747927895
```

(c) Input Representation - log-normalized counts where each entry becomes log(td + 1).

```
In [7]: X_train_best = pd.DataFrame()
        for feature in top_features:
            X_train_best[feature] = X_train[feature]
        X_train_best = np.log(X_train_best + 1)
        X_train_best.head()
```

Out[7]:

	the	of	that	in	to	god	windows	and	is
0	4.510860	4.077537	2.197225	3.044522	3.332205	2.772589	0.0	4.007333	3.218876	2.197
1	5.170484	5.049856	4.875197	4.418841	5.187386	4.025352	0.0	4.304065	4.844187	2.564
2	3.713572	3.433987	2.944439	2.772589	2.890372	0.000000	0.0	3.091042	2.944439	1.098
3	2.484907	2.197225	0.000000	0.693147	1.945910	0.000000	0.0	2.197225	1.098612	0.693
4	1.098612	0.693147	0.693147	0.693147	1.098612	0.000000	0.0	1.098612	0.693147	0.000

5 rows × 5000 columns

(d) Classifier - using linear support vector machine classifiers and using 5-Fold cross validation

```
In [8]: from sklearn.svm import LinearSVC
        from sklearn.model_selection import cross_val_score

        #clf1 = LinearSVC(penalty="l1", dual=False, tol=1e-3)
        clf2 = LinearSVC(penalty="l2", dual=False, tol=1e-3)

        #print('clf1', cross_val_score(clf1, X_train_best, Y_train, n_jobs=-1))
        print('clf2', cross_val_score(clf2, X_train_best, Y_train, n_jobs=-1))
```

```
clf2 [0.77379371 0.82558654 0.84373617 0.82949513 0.81310895]
```

From the cross validation above, we can see classifier 2 (clf2) is better

```
In [9]: clf2.fit(X_train_best, Y_train)
```

Out[9]: LinearSVC(dual=False, tol=0.001)

(e) Evaluation - load and process the test data

```
In [15]: df_test = pd.read_csv('20ng-test-all-terms.csv', names=['class_publication_
         X_test_best = pd.DataFrame()
         X_test = vectorizer.fit_transform(df_test.document.to_numpy())
         word_count_df_test = pd.DataFrame(X_test.toarray(), columns = vectorizer.ge
         for feature in top_features:
             if feature in word_count_df_test:
                 X_test_best[feature] = word_count_df_test[feature]
             else:
                 X_test_best[feature] = 1
         X_test_best = np.log(X_test_best + 1)
         X_test_best.head()
```

Out[15]:

	the	of	that	in	to	god	windows	and	is	
0	2.708050	1.791759	0.693147	0.693147	1.386294	0.693147	0.0	1.098612	1.098612	0.693
1	2.944439	2.564949	2.708050	2.197225	2.708050	0.693147	0.0	2.079442	2.484907	1.386
2	2.197225	1.386294	2.197225	1.791759	2.708050	0.000000	0.0	1.386294	2.397895	0.693
3	1.945910	1.098612	1.386294	0.693147	2.564949	1.609438	0.0	0.693147	2.079442	1.098
4	3.332205	2.564949	2.772589	2.397895	1.791759	0.000000	0.0	1.945910	2.484907	0.000

5 rows × 5000 columns

(e) Evaluation - evaluate using test data and report the accuracy

```
In [16]: Y_test = df_test.class_publication_name
         Y_pred = clf2.predict(X_test_best)
         from sklearn.metrics import classification_report
         print(classification_report(Y_test, Y_pred))
```

	precision	recall	f1-score	support
alt.atheism	0.75	0.68	0.71	319
comp.graphics	0.67	0.60	0.63	389
comp.os.ms-windows.misc	0.71	0.66	0.68	394
comp.sys.ibm.pc.hardware	0.65	0.60	0.62	392
comp.sys.mac.hardware	0.70	0.76	0.73	385
comp.windows.x	0.77	0.68	0.72	392
misc.forsale	0.84	0.86	0.85	390
rec.autos	0.81	0.75	0.78	395
rec.motorcycles	0.78	0.90	0.83	398
rec.sport.baseball	0.91	0.86	0.89	397
rec.sport.hockey	0.94	0.93	0.94	399
sci.crypt	0.87	0.84	0.86	396
sci.electronics	0.63	0.63	0.63	393
sci.med	0.78	0.72	0.75	396
sci.space	0.87	0.83	0.85	394
soc.religion.christian	0.74	0.85	0.79	398
talk.politics.guns	0.68	0.82	0.74	364

(e) Evaluation - five largest off-diagonal entries

In [25]:
```python
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(Y_test, Y_pred)
top_5 = [0,0,0,0,0]
for i in range(confusion_matrix.shape[0]):
    for j in range(confusion_matrix.shape[1]):
        if((i != j) & (confusion_matrix[i][j] > top_5[0])):
            top_5.pop(0)
            top_5.append(confusion_matrix[i][j])
            top_5.sort()
print(confusion_matrix,'\n', top_5)
```

```
[[217   1   0   1   0   1   0   0   6   0   0   1   1  11   6  21   2   8
   12  31]
 [  1 233  19  12  11  31  10   2   8   2   0   5  19   0   4   4   2   8
   15   3]
 [  1  19 262  36  12  15   2   1   3   1   1   2   5   2   1   5   2   3
   20   1]
 [  1  14  29 236  35   9  11   3   2   0   0   4  29   3   1   3   2   2
    8   0]
 [  0   4   9  21 293   2   8   2   4   2   1   7  19   1   2   0   1   0
    9   0]
 [  0  37  28   8  11 268   4   1   5   0   0   0   9   5   3   1   0   0
   12   0]
 [  0   3   1  10  10   0 335   5   4   0   1   0   7   2   1   0   1   1
    9   0]
 [  1   0   2   2   4   1  12 295  22   2   0   1  15   7   2   2   5   7
   14   1]
 [  0   1   1   1   4   0   3  13 357   1   0   0   2   1   1   1   2   3
    7   0]
 [  1   1   1   3   2   2   1   3   2 343  12   2   1   3   0   1   0   1
   17   1]
 [  2   1   0   1   1   1   0   1   1  12 371   1   1   0   1   0   2   0
    3   0]
 [  0   2   1   1   7   1   2   1   4   3   1 333  11   1   0   0   3   5
   19   1]
 [  3  10  12  22  12   4   8   8   6   1   1  11 249  10   8   7   1   7
   13   0]
 [  4   6   2   4   6   5   0  13   7   5   1   2  13 284   5  10   5   2
   20   2]
 [  3   9   2   1   3   2   1   2   4   1   0   1   7  12 326   6   0   2
   11   1]
 [  8   1   0   1   0   2   0   4   3   0   0   2   2   7   2 340   2   3
   12   9]
 [  3   0   2   1   1   3   1   3   3   0   1   3   1   4   0   3 298   1
   30   6]
 [ 15   0   0   0   2   1   1   3   4   2   1   2   4   0   4  10   6 284
   34   3]
 [  1   1   0   2   1   0   0   3   4   2   0   4   0   7   5   3  87  10
  171   9]
 [ 30   4   0   2   2   0   0   2  11   0   2   0   3   6   3  42  16   3
   16 109]]
 [35, 36, 37, 42, 87]
```

Five largest off-diagonal entries are:

1) 87 times talk.politics.misc got predicted as talk.politics.guns 2) 42 times talk.religion.misc got predicted as soc.religion.christians 3) 37 times comp.windows.x got predicted as comp.graphics 4) 36 times comp.os.ms-windows.misc got predicted as comp.graphics 5) 35 times comp.sys.ibm.pc.hardware got predicted as comp.sys.mac.hardware

All of the above examples shows that our model sometimes predicts similar clases from the to same domain (for ex. predicting talk.politics.guns instead of talk.politics.misc).

(f) Model Inspection - table of top 10 and bottom 10 features for each class

In [18]:
```
frequent_word_df_expanded = frequent_word_df.copy()
frequent_word_df_expanded.insert(0, 'class_publication_name', df.class_publ
frequent_word_df_weighted = frequent_word_df_expanded.groupby('class_public
frequent_word_df_weighted.head()
```

Out[18]:

class_publication_name	aa	aaa	aamir	aardvark	aaron	aas	aau	ab	abandon	abandoned	.
alt.atheism	2	2	0	0	0	0	0	0	0	1	.
comp.graphics	1	0	0	0	0	0	2	18	1	0	.
comp.os.ms-windows.misc	2	0	0	0	1	0	0	2	3	0	.
comp.sys.ibm.pc.hardware	15	0	0	0	0	0	6	5	0	0	.
comp.sys.mac.hardware	3	1	0	1	5	0	4	1	0	0	.

5 rows × 15226 columns

```
In [30]: 0,20):
         ,0,0,0,0,0,0,0,0,0]
         ures = []
          [100,100,100,100,100,100,100,100,100,100]
         eatures = []
         nge (0, frequent_word_df_weighted.shape[1]):
         uent_word_df_weighted.iloc[i][j] > top_10[0]):
         _10.pop(0)
         _10.append(frequent_word_df_weighted.iloc[i][j])
         _10_features.append(frequent_word_df_weighted.columns.values[j])
         _5.sort()
         uent_word_df_weighted.iloc[i][j] < bottom_10[9]):
         tom_10.insert(0,frequent_word_df_weighted.iloc[i][j])
         tom_10_features.insert(0, frequent_word_df_weighted.columns.values[j])
         tom_10.sort()
         ent_word_df_weighted.index.values[i], "top_10", top_10_features[-10:])
         1d(top_10_features[-10:], top_features))
         ent_word_df_weighted.index.values[i], "bottom_10", bottom_10_features[:10])
         1d(bottom_10_features[:10], top_features))
```

```
[False False False False  True False False  True False False]
sci.crypt top_10 ['on', 'one', 'only', 'or', 'order', 'other', 'that', 't
he', 'their', 'to']
[ True  True  True  True  True  True  True  True  True  True]
sci.crypt bottom_10 ['abdullah', 'abbreviation', 'abbott', 'abate', 'aban
doned', 'abandon', 'ab', 'aau', 'aas', 'aaron']
[False False False False False False  True False False  True]
sci.electronics top_10 ['that', 'the', 'to', 'two', 'type', 'uk', 'uky',
'university', 'up', 'you']
[ True  True  True  True  True  True False  True  True  True]
sci.electronics bottom_10 ['aboard', 'abo', 'abilities', 'abiding', 'abdu
llah', 'abc', 'abbreviation', 'abbott', 'abate', 'abandoned']
[False False False  True False  True False False False False]
sci.med top_10 ['that', 'the', 'their', 'them', 'then', 'theory', 'therap
ies', 'this', 'to', 'tobacco']
[ True  True  True  True  True  True False  True  True False]
sci.med bottom_10 ['aber', 'abdullah', 'abc', 'abbreviation', 'abbott',
'abate', 'abandoned', 'abandon', 'ab', 'aau']
[False False  True False False False False False  True False]
sci.space top_10 ['space', 'the', 'their', 'them', 'then', 'there', 'to',
```

The top 10 and bottom 10 features against each class are inlined with what we found using mutual information. All the top 10 features are in our list of top 5000 words as per mutual information.