Machine Learning Assignment 3

Tingyu Shi(400253854)

June 24, 2023

Contents

1 Task 1

1.1 A.1 (MLE Derivation)

From the questions given, we know the following facts:

1. Since all the features follow Bernoulli distribution, we have write the following PMF for features:

$$P(x_i|y=0,\theta) = \alpha_i^{x_i} * (1-\alpha_i)^{1-x_i}$$
$$P(x_i|y=1,\theta) = \beta_i^{x_i} * (1-\beta_i)^{1-x_i}$$

In order to do MLE, we need to make the following assumptions:

1. Assume that there are n_0 data points from the training set that have label 0.
2. Assume that there are n_1 data points from the training set that have label 1.

$$\begin{aligned}
\theta &= \underset{\theta}{argmax}\ P(Z|\theta) \\
&= \underset{\theta}{argmax}\ P(X,Y|\theta) \\
&= \underset{\theta}{argmax}\ \prod_{i=1}^{n} P(x^i, y^i|\theta) \\
&= \underset{\theta}{argmax}\ \prod_{i=1}^{n} P(x^i|y^i,\theta) * P(y^i|\theta) \\
&= \underset{\theta}{argmax}\ \left(\prod_{i:y^i=0} P(x^i|y^i=0,\theta) * P(y^i=0|\theta)\right) * \left(\prod_{i:y^i=1} P(x^i|y^i=1,\theta) * P(y^i=1|\theta)\right) \\
&= \underset{\theta}{argmax}\ \left(\prod_{i:y^i=0} P(x^i|y^i=0,\theta)\right) * \left(\prod_{i:y^i=1} P(x^i|y^i=1,\theta)\right) * \left(\prod_{i:y^i=0} P(y^i=0|\theta) * \prod_{i:y^i=1} P(y^i=1|\theta)\right)
\end{aligned}$$

We can clearly see that right now we have the following three parts from the above equation

1. $$\prod_{i:y^i=0} P(x^i|y^i=0,\theta)$$
2. $$\prod_{i:y^i=1} P(x^i|y^i=1,\theta)$$
3. $$\prod_{i:y^i=0} P(y^i=0|\theta) * \prod_{i:y^i=1} P(y^i=1|\theta)$$

In order to maximize the whole equation, we can maximize each part individually. Also, the first part only relates to α. The second part only relates β. The third part only relates to γ. Therefore, we can rewrite the above three parts as the following:

1.
$$\prod_{i:y^i=0} P(x^i|y^i = 0, \alpha)$$

2.
$$\prod_{i:y^i=1} P(x^i|y^i = 1, \beta)$$

3.
$$\prod_{i:y^i=0} P(y^i = 0|\gamma) * \prod_{i:y^i=1} P(y^i = 1|\gamma)$$

1.1.1 Find α

$$
\begin{aligned}
\alpha &= \underset{\alpha}{argmax} \prod_{i:y^i=0} P(x^i|y^i=0,\alpha) \\
&= \underset{\alpha}{argmax} \prod_{i:y^i=0} P(x_1^i \dots x_d^i|y^i=0,\alpha) \\
&= \underset{\alpha}{argmax} \prod_{i:y^i=0} \left(\prod_{j=1}^{d} P(x_j^i|y^i=0,\alpha_j) \right) \\
&= \underset{\alpha}{argmax} \prod_{i:y^i=0} \left(\prod_{j=1}^{d} \alpha_j^{x_j^i} * (1-\alpha_j)^{(1-x_j^i)} \right)
\end{aligned}
$$

Since $\alpha_j (j \in [1,d])$ are independent of each other, we can choose to optimize a single α_j, then we know how to optimize all α_js.

$$
\begin{aligned}
\alpha_j &= \underset{\alpha_j}{argmax} \prod_{i:y^i=0} \alpha_j^{x_j^i} * (1-\alpha_j)^{(1-x_j^i)} \\
&= \underset{\alpha_j}{argmax} \sum_{i:y^i=0} x_j^i * log(\alpha_j) + (1-x_j^i) * log(1-\alpha_j) \\
&= \underset{\alpha_j}{argmax} \sum_{i:y^i=0} x_j^i * log(\alpha_j) + \sum_{i:y^i=0} (1-x_j^i) * log(1-\alpha_j) \\
&= \underset{\alpha_j}{argmax}\ log(\alpha_j) * \sum_{i:y^i=0} x_j^i + log(1-\alpha_j) * \sum_{i:y^i=0} (1-x_j^i) \\
&= \underset{\alpha_j}{argmax}\ log(\alpha_j) * \sum_{i:y^i=0} x_j^i + log(1-\alpha_j) * (n_0 - \sum_{i:y^i=0} x_j^i)
\end{aligned}
$$

Let

$$f(\alpha_j) = log(\alpha_j) * \sum_{i:y^i=0} x_j^i + log(1-\alpha_j) * (n_0 - \sum_{i:y^i=0} x_j^i)$$

$$\frac{\partial f(\alpha_j)}{\partial \alpha_j} = \frac{\sum\limits_{i:y^i=0} x_j^i}{\alpha_j} - \frac{n_0 - \sum\limits_{i:y^i=0} x_j^i}{1-\alpha_j}$$

Then, let

$$\frac{\partial f(\alpha_j)}{\partial \alpha_j} = 0$$

$$\alpha_j = \frac{\sum\limits_{i:y^i=0} x_j^i}{n_0}$$

1.1.2 Find β

$$
\begin{aligned}
\beta &= \underset{\beta}{argmax} \prod_{i:y^i=1} P(x^i|y^i=1,\beta) \\
&= \underset{\beta}{argmax} \prod_{i:y^i=1} P(x_1^i \dots x_d^i|y^i=1,\beta) \\
&= \underset{\beta}{argmax} \prod_{i:y^i=1} \left(\prod_{j=1}^{d} P(x_j^i|y^i=1,\beta_j) \right) \\
&= \underset{\beta}{argmax} \prod_{i:y^i=1} \left(\prod_{j=1}^{d} \beta_j^{x_j^i} * (1-\beta_j)^{(1-x_j^i)} \right)
\end{aligned}
$$

Since $\beta_j (j \in [1,d])$ are independent of each other, we can choose to optimize a single β_j, then we know how to optimize all β_js.

$$
\begin{aligned}
\beta_j &= \underset{\beta_j}{argmax} \prod_{i:y^i=1} \beta_j^{x_j^i} * (1-\beta_j)^{(1-x_j^i)} \\
&= \underset{\beta_j}{argmax} \sum_{i:y^i=1} x_j^i * log(\beta_j) + (1-x_j^i) * log(1-\beta_j) \\
&= \underset{\beta_j}{argmax} \sum_{i:y^i=1} x_j^i * log(\beta_j) + \sum_{i:y^i=1} (1-x_j^i) * log(1-\beta_j) \\
&= \underset{\beta_j}{argmax}\ log(\beta_j) * \sum_{i:y^i=1} x_j^i + log(1-\beta_j) * \sum_{i:y^i=1} (1-x_j^i) \\
&= \underset{\beta_j}{argmax}\ log(\beta_j) * \sum_{i:y^i=1} x_j^i + log(1-\beta_j) * (n_1 - \sum_{i:y^i=1} x_j^i)
\end{aligned}
$$

Let

$$f(\beta_j) = log(\beta_j) * \sum_{i:y^i=1} x_j^i + log(1-\beta_j) * (n_1 - \sum_{i:y^i=1} x_j^i)$$

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = \frac{\sum_{i:y^i=1} x_j^i}{\beta_j} - \frac{n_1 - \sum_{i:y^i=1} x_j^i}{1-\beta_j}$$

Then, let

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = 0$$

$$\beta_j = \frac{\sum_{i:y^i=1} x_j^i}{n_1}$$

1.1.3 Find γ

$$\begin{aligned}\gamma &= \underset{\gamma}{argmax} \prod_{i:y^i=0} P(y^i=0|\gamma) * \prod_{i:y^i=1} P(y^i=1|\gamma) \\ &= \underset{\gamma}{argmax} \prod_{i:y^i=0} \gamma * \prod_{i:y^i=1} (1-\gamma) \\ &= \underset{\gamma}{argmax}\ \gamma^{n_0}(1-\gamma)^{n_1} \\ &= \underset{\gamma}{argmax}\ log(\gamma^{n_0}(1-\gamma)^{n_1}) \\ &= \underset{\gamma}{argmax}\ n_0 log(\gamma) + n_1 log(1-\gamma)\end{aligned}$$

Let

$$f(\gamma) = n_0 log(\gamma) + n_1 log(1-\gamma)$$

$$\frac{\partial f(\gamma)}{\partial \gamma} = \frac{n_0}{\gamma} - \frac{n_1}{1-\gamma}$$

Let

$$\frac{\partial f(\gamma)}{\partial \gamma} = 0$$

$$\gamma = \frac{n_0}{n_0+n_1}$$

1.2 A.2 (Parameters with MLE)

The following are printed outputs:

gamma is 0.49575551782682514

beta is [0.03198653198653199, 0.03198653198653199, 0.0016835016835016834,
0.003367003367003367, 0.0, 0.0,
0.0016835016835016834, 0.0, 0.0,
0.0016835016835016834]

alpha is [0.03424657534246575, 0.018835616438356163, 0.0,
0.0, 0.0017123287671232876, 0.0017123287671232876,
0.0, 0.0017123287671232876, 0.0017123287671232876,
0.0]

1.3 B (Naive bayes classifier)

No outputs were generated in this section, please check the code in the code file.

1.4 C (Prediction Accuracy With MLE)

Accuracy on Training Set(using MLE): 96.01018675721562 %
Accuracy on Testing Set(using MLE): 62.54777070063694 %

1.5 D.1 (MAP Derivation)

For the P.M.F of Bernoulli distribution, it was shown in section A.1. In order to do MAP, we need to make the following assumptions:

1. Assume that there are n_0 data points from the training set that have label 0.
2. Assume that there are n_1 data points from the training set that have label 1.

$$
\begin{aligned}
\theta &= \underset{\theta}{argmax}\ P(\theta|Z) \\
&= \underset{\theta}{argmax}\ \frac{P(Z|\theta) * P(\theta)}{P(Z)} \\
&= \underset{\theta}{argmax}\ P(Z|\theta) * P(\theta) \\
&= \underset{\theta}{argmax}\ P(X, Y|\theta) * P(\theta) \\
&= \underset{\theta}{argmax}\ \left(\prod_{i=1}^{n} P(x^i, y^i|\theta)\right) * \left(P(\gamma) * \prod_{i=1}^{d} P(\alpha_i)P(\beta_i)\right) \\
&= \underset{\theta}{argmax}\ \left(\prod_{i=1}^{n} P(x^i|y^i, \theta) * P(y^i|\theta)\right) * \left(P(\gamma) * \prod_{i=1}^{d} P(\alpha_i)P(\beta_i)\right) \\
&= \underset{\theta}{argmax}\ \left(\prod_{i:y^i=0} P(x^i|y^i = 0, \theta) * P(y^i = 0|\theta)\right) * \left(\prod_{i:y^i=1} P(x^i|y^i = 1, \theta) * P(y^i = 1|\theta)\right) * \\
&\qquad \left(P(\gamma) * \prod_{i=1}^{d} P(\alpha_i)P(\beta_i)\right) \\
&= \underset{\theta}{argmax}\ \left[\left(\prod_{i:y^i=0} P(x^i|y^i = 0, \theta)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right] * \\
&\qquad \left[\left(\prod_{i:y^i=1} P(x^i|y^i = 1, \theta)\right) * \left(\prod_{i=1}^{d} P(\beta_i)\right)\right] * \\
&\qquad \left[P(\gamma) * \left(\prod_{i:y^i=0} P(y^i = 0|\theta)\right) * \left(\prod_{i:y^i=1} P(y^i = 1|\theta)\right)\right]
\end{aligned}
$$

We can clearly see that right now we have the following three parts from the above equation

1. $$\left[\left(\prod_{i:y^i=0} P(x^i|y^i=0,\theta)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right]$$

2. $$\left[\left(\prod_{i:y^i=1} P(x^i|y^i=1,\theta)\right) * \left(\prod_{i=1}^{d} P(\beta_i)\right)\right]$$

3. $$\left[P(\gamma) * \left(\prod_{i:y^i=0} P(y^i=0|\theta)\right) * \left(\prod_{i:y^i=1} P(y^i=1|\theta)\right)\right]$$

In order to maximize the whole equation, we can maximize each part individually. Also, the first part only relates to α. The second part only relates β. The third part only relates to γ. Therefore, we can rewrite the above three parts as the following:

1. $$\left[\left(\prod_{i:y^i=0} P(x^i|y^i=0,\alpha)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right]$$

2. $$\left[\left(\prod_{i:y^i=1} P(x^i|y^i=1,\beta)\right) * \left(\prod_{i=1}^{d} P(\beta_i)\right)\right]$$

3. $$\left[P(\gamma) * \left(\prod_{i:y^i=0} P(y^i=0|\gamma)\right) * \left(\prod_{i:y^i=1} P(y^i=1|\gamma)\right)\right]$$

1.5.1 Find α

$$
\begin{aligned}
\alpha &= \underset{\alpha}{argmax} \left[\left(\prod_{i:y^i=0} P(x^i|y^i=0,\alpha)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right] \\
&= \underset{\alpha}{argmax} \left[\left(\prod_{i:y^i=0} P(x_1^i \ldots x_d^i|y^i=0,\alpha)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right] \\
&= \underset{\alpha}{argmax} \left[\left(\prod_{i:y^i=0} \left(\prod_{j=1}^{d} P(x_j^i|y^i=0,\alpha_j)\right)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right] \\
&= \underset{\alpha}{argmax} \left[\left(\prod_{i:y^i=0} \left(\prod_{j=1}^{d} \alpha_j^{x_j^i} * (1-\alpha_j)^{1-x_j^i}\right)\right) * \left(\prod_{i=1}^{d} P(\alpha_i)\right)\right]
\end{aligned}
$$

Since $\alpha_j (j \in [1,d])$ are independent of each other, we can choose to optimize a single α_j, then we know how to optimize all α_js.

$$
\begin{aligned}
\alpha_j &= \underset{\alpha_j}{argmax} \left(\prod_{i:y^i=0} \alpha_j^{x_j^i} * (1-\alpha_j)^{(1-x_j^i)}\right) * P(\alpha_j) \\
&= \underset{\alpha_j}{argmax} \left(\sum_{i:y^i=0} x_j^i * log(\alpha_j)\right) + \left(\sum_{i:y^i=0} (1-x_j^i) * log(1-\alpha_j)\right) + log(P(\alpha_j)) \\
&= \underset{\alpha_j}{argmax} \left(log(\alpha_j) \sum_{i:y^i=0} x_j^i\right) + \left(log(1-\alpha_j) \sum_{i:y^i=0} (1-x_j^i)\right) + log(P(\alpha_j)) \\
&= \underset{\alpha_j}{argmax} \left(log(\alpha_j) \sum_{i:y^i=0} x_j^i\right) + \left(log(1-\alpha_j) \left(n_0 - \sum_{i:y^i=0} x_j^i\right)\right) + log(P(\alpha_j))
\end{aligned}
$$

Let

$$
f(\alpha_j) = log(\alpha_j) \sum_{i:y^i=0} x_j^i + log(1-\alpha_j) \left(n_0 - \sum_{i:y^i=0} x_j^i\right) + log(P(\alpha_j))
$$

When $\alpha_j \leq 0.5$

$$
f(\alpha_j) = log(\alpha_j) \sum_{i:y^i=0} x_j^i + log(1-\alpha_j) \left(n_0 - \sum_{i:y^i=0} x_j^i\right) + log(4\alpha_j)
$$

$$
\frac{\partial f(\alpha_j)}{\partial \alpha_j} = \frac{(n_0+1)\alpha_j - \left(\sum_{i:y^i=0} x_j^i\right) - 1}{(\alpha_j - 1)\alpha_j}
$$

Then, let

$$
\frac{\partial f(\alpha_j)}{\partial \alpha_j} = 0
$$

$$\alpha_j = \frac{\left(\sum_{i:y^i=0} x_j^i\right) + 1}{n_0 + 1}$$

When $\alpha_j > 0.5$

$$f(\alpha_j) = log(\alpha_j) \sum_{i:y^i=0} x_j^i + log(1-\alpha_j)\left(n_0 - \sum_{i:y^i=0} x_j^i\right) + log(4 - 4\alpha_j)$$

$$\frac{\partial f(\alpha_j)}{\partial \alpha_j} = \frac{(n_0+1)\alpha_j - \left(\sum_{i:y^i=0} x_j^i\right)}{(\alpha_j - 1)\alpha_j}$$

Then, let

$$\frac{\partial f(\alpha_j)}{\partial \alpha_j} = 0$$

$$\alpha_j = \frac{\left(\sum_{i:y^i=0} x_j^i\right)}{n_0 + 1}$$

1.5.2 Find β

$$\begin{aligned}
\beta &= \underset{\beta}{argmax} \left[\left(\prod_{i:y^i=1} P(x^i|y^i=1,\beta) \right) * \left(\prod_{i=1}^{d} P(\beta_i) \right) \right] \\
&= \underset{\beta}{argmax} \left[\left(\prod_{i:y^i=1} P(x^i_1 \dots x^i_d|y^i=1,\beta) \right) * \left(\prod_{i=1}^{d} P(\beta_i) \right) \right] \\
&= \underset{\beta}{argmax} \left[\left(\prod_{i:y^i=1} \left(\prod_{j=1}^{d} P(x^i_j|y^i=1,\beta_j) \right) \right) * \left(\prod_{i=1}^{d} P(\beta_i) \right) \right] \\
&= \underset{\beta}{argmax} \left[\left(\prod_{i:y^i=1} \left(\prod_{j=1}^{d} \beta_j^{x^i_j} * (1-\beta_j)^{1-x^i_j} \right) \right) * \left(\prod_{i=1}^{d} P(\beta_i) \right) \right]
\end{aligned}$$

Since $\beta_j (j \in [1,d])$ are independent of each other, we can choose to optimize a single β_j, then we know how to optimize all β_js.

$$\begin{aligned}
\beta_j &= \underset{\beta_j}{argmax} \left(\prod_{i:y^i=1} \beta_j^{x^i_j} * (1-\beta_j)^{(1-x^i_j)} \right) * P(\beta_j) \\
&= \underset{\beta_j}{argmax} \left(\sum_{i:y^i=1} x^i_j * log(\beta_j) \right) + \left(\sum_{i:y^i=1} (1-x^i_j) * log(1-\beta_j) \right) + log(P(\beta_j)) \\
&= \underset{\beta_j}{argmax} \left(log(\beta_j) \sum_{i:y^i=1} x^i_j \right) + \left(log(1-\beta_j) \sum_{i:y^i=1} (1-x^i_j) \right) + log(P(\beta_j)) \\
&= \underset{\beta_j}{argmax} \left(log(\beta_j) \sum_{i:y^i=1} x^i_j \right) + \left(log(1-\beta_j) \left(n_1 - \sum_{i:y^i=1} x^i_j \right) \right) + log(P(\beta_j))
\end{aligned}$$

Let

$$f(\beta_j) = log(\beta_j) \sum_{i:y^i=1} x^i_j + log(1-\beta_j) \left(n_1 - \sum_{i:y^i=1} x^i_j \right) + log(P(\beta_j))$$

When $\beta_j \leq 0.5$

$$f(\beta_j) = log(\beta_j) \sum_{i:y^i=1} x^i_j + log(1-\beta_j) \left(n_1 - \sum_{i:y^i=1} x^i_j \right) + log(4\beta_j)$$

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = \frac{(n_1+1)\beta_j - \left(\sum_{i:y^i=1} x^i_j \right) - 1}{(\beta_j - 1)\beta_j}$$

Then, let

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = 0$$

$$\beta_j = \frac{\left(\sum_{i:y^i=1} x_j^i \right) + 1}{n_1 + 1}$$

When $\beta_j > 0.5$

$$f(\beta_j) = log(\beta_j) \sum_{i:y^i=1} x_j^i + log(1 - \beta_j) \left(n_1 - \sum_{i:y^i=1} x_j^i \right) + log(4 - 4\beta_j)$$

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = \frac{(n_1 + 1)\beta_j - \left(\sum_{i:y^i=1} x_j^i \right)}{(\beta_j - 1)\beta_j}$$

Then, let

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = 0$$

$$\beta_j = \frac{\left(\sum_{i:y^i=1} x_j^i \right)}{n_1 + 1}$$

1.5.3 Find γ

$$
\begin{aligned}
\gamma &= \underset{\gamma}{argmax} \left[P(\gamma) * \left(\prod_{i:y^i=0} P(y^i=0|\theta) \right) * \left(\prod_{i:y^i=1} P(y^i=1|\theta) \right) \right] \\
&= \underset{\gamma}{argmax}\ P(\gamma)\gamma^{n_0}(1-\gamma)^{n_1} \\
&= \underset{\gamma}{argmax}\ log(P(\gamma)\gamma^{n_0}(1-\gamma)^{n_1}) \\
&= \underset{\gamma}{argmax}\ n_0 log(\gamma) + n_1 log(1-\gamma) + log(P(\gamma))
\end{aligned}
$$

Let

$$f(\gamma) = n_0 log(\gamma) + n_1 log(1-\gamma) + log(P(\gamma))$$

When $\gamma \leq 0.5$

$$f(\gamma) = n_0 log(\gamma) + n_1 log(1-\gamma) + log(4\gamma)$$

$$\frac{\partial f(\gamma)}{\partial \gamma} = \frac{n_0}{\gamma} + \frac{1}{\gamma} - \frac{n_1}{1-\gamma}$$

Let

$$\frac{\partial f(\gamma)}{\partial \gamma} = 0$$

$$\gamma = \frac{n_0 + 1}{n_0 + n_1 + 1}$$

When $\gamma > 0.5$

$$f(\gamma) = n_0 log(\gamma) + n_1 log(1-\gamma) + log(4-4\gamma)$$

$$\frac{\partial f(\gamma)}{\partial \gamma} = \frac{(n_0 + n_1 + 1)\gamma - n_0}{(\gamma - 1)\gamma}$$

Let

$$\frac{\partial f(\gamma)}{\partial \gamma} = 0$$

$$\gamma = \frac{n_0}{n_0 + n_1 + 1}$$

1.6 D.2 (Parameters With MAP)

The following are printed outputs:

gamma is 0.4961832061068702

beta is [0.03361344537815126, 0.03361344537815126, 0.0033613445378151263, 0.005042016806722689, 0.0016806722689075631, 0.0016806722689075631, 0.0033613445378151263, 0.0016806722689075631, 0.0016806722689075631, 0.0033613445378151263]

alpha is [0.035897435897435895, 0.020512820512820513, 0.0017094017094017094, 0.0017094017094017094, 0.003418803418803419, 0.003418803418803419, 0.0017094017094017094, 0.003418803418803419, 0.003418803418803419, 0.0017094017094017094]

1.7 D.3 (Prediction Accuracy With MAP)

Accuracy on Training Set(using MAP): 88.96434634974533 %
Accuracy on Testing Set(using MAP): 75.54140127388536 %

1.8 D.4 (MLE VS. MAP)

For accuracy on training set, MLE performs better than MAP. MLE has accuracy of 96.01% on training set and MAP has accuracy of 88.96% on training set. However, when it comes to testing dataset, MAP(with accuracy of 75.54%) performed better than MLE(with accuracy of 62.55%). Clearly, MLE method caused overfitting problem.

Justification:

1. For MLE, parameters estimations only base on the dataset we have. It has no knowledge about the distribution of the parameters. As a result, MLE cannot generalize the model we are developing so that MLE caused overfitting problem.

2. By using MAP, we can avoid 0 values in α_i and β_i . If α_i and β_i calculated from MLE are 0, then 1 will be added to the numerator when calculating MAP α_i and β_i , therefore, 0 values are avoided in α_i and β_i for MAP. As a result, we can avoid 0 probability values when we do the predictions, which can make predictions more meaningful.

2 Task 2 (SVM Classifier)

2.1 A (Linear Classifier)

Accuracy on Training Set(using SVM linear kernel): 98.49357554275588 %
Accuracy on Testing Set(using SVM linaer kernel): 88.08255659121171 %

2.2 B (RBF kernel effect)

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.70): 98.93664155959237 %
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.70): 86.21837549933421 %

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.65): 98.84802835622509 %
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.65): 86.08521970705726 %

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.60): 98.58218874612317 %
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.60): 85.9520639147803 %

Discussion:
RBF kernel is used to measure the similarity between two points. Smaller gamma value means further influence. As a result, two points can be considered similar even if they are quite far from each other. This is not desirable for classification problem. As a consequence, smaller gamma value produces worse accuracy for classification problem.

2.3 C (IDF Importance)

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.70 idf=True): 99.9113867966327%
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.70 idf=True): 90.0133155792277%

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.65 idf=True): 99.9113867966327%
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.65 idf=True): 90.14647137150466%

Accuracy on Training Set(using SVM RBF kernel with gamma = 0.60 idf=True): 99.9113867966327%
Accuracy on Testing Set(using SVM RBF kernel with gamma = 0.60 idf=True): 90.21304926764314%

Discussion:
If we turn on use_idf, the accuracies have increased on both training set and testing for all three gamma values. Turning on use_idf can decrease the impact of frequent words such as "is", "the", etc. This can increase the impact of other more important features in disguise. As a result, accuracies on both training set and testing set have increased.