# *CS 6375*

# *Machine Learning*

## *Assignment 3.1*

## *Perceptron and K means*

*The University of Texas at Dallas*


***Submitted by,***

***Parth Mehta***

***NetID: pjm150030***

# Perceptron Algorithm Implementation

I have implemented the perceptron algorithm. I have taken the Spam and Ham dataset and I am checking the accuracy by varying Learning rate, number of iteration and stop words.

The perceptron accuracy with stop words is given in the table below.

***With Stop Words:***

| Number of Iteration | Learning Rate | Naïve Bayes | | Logistic Regression (With Lambda = 0) | | Perceptron | |
|---|---|---|---|---|---|---|---|
| | | Spam Accuracy | Ham Accuracy | Spam Accuracy | Ham Accuracy | Spam Accuracy | Ham Accuracy |
| 100 | 0.03 | 84.61 | 97.98 | 83.07 | 94.54 | 83.84 | 93.67 |
| 80 | 0.03 | 84.61 | 97.98 | 83.07 | 94.82 | 83.84 | 93.67 |
| 60 | 0.03 | 84.61 | 97.98 | 83.07 | 95.11 | 83.84 | 93.67 |
| 40 | 0.03 | 84.61 | 97.98 | 83.07 | 94.54 | 83.84 | 93.67 |
| 20 | 0.03 | 84.61 | 97.98 | 94.61 | 77.58 | 70.0 | 95.68 |
| 100 | 0.09 | 84.61 | 97.98 | 83.07 | 94.82 | 83.84 | 91.66 |
| 80 | 0.09 | 84.61 | 97.98 | 83.07 | 95.11 | 83.84 | 91.66 |
| 60 | 0.09 | 84.61 | 97.98 | 83.84 | 94.82 | 83.84 | 91.66 |
| 40 | 0.09 | 84.61 | 97.98 | 85.38 | 93.96 | 83.84 | 91.66 |
| 20 | 0.09 | 84.61 | 97.98 | 94.61 | 76.72 | 70.0 | 97.12 |
| 100 | 0.27 | 84.61 | 97.98 | 83.07 | 94.82 | 86.15 | 92.52 |
| 80 | 0.27 | 84.61 | 97.98 | 83.07 | 95.11 | 86.15 | 92.52 |
| 60 | 0.27 | 84.61 | 97.98 | 83.07 | 95.40 | 86.15 | 92.52 |
| 40 | 0.27 | 84.61 | 97.98 | 83.07 | 95.11 | 86.15 | 92.52 |
| 20 | 0.27 | 84.61 | 97.98 | 94.61 | 77.3 | 70 | 96.55 |
| 100 | 0.81 | 84.61 | 97.98 | 82.30 | 94.82 | 89.23 | 91.09 |
| 80 | 0.81 | 84.61 | 97.98 | 82.30 | 95.11 | 89.23 | 91.09 |
| 60 | 0.81 | 84.61 | 97.98 | 83.07 | 94.82 | 89.23 | 91.09 |
| 40 | 0.81 | 84.61 | 97.98 | 82.30 | 95.11 | 89.23 | 91.09 |
| 20 | 0.81 | 84.61 | 97.98 | 94.61 | 76.72 | 72.3 | 95.68 |

As we can see from the above table, the results from perceptrons and Logistic Regression are almost similar for our dataset.

***Without Stop Words***

| Number of Iteration | Learning Rate | Naïve Bayes | | Logistic Regression (With Lambda = 0) | | Perceptron | |
|---|---|---|---|---|---|---|---|
| | | Spam Accuracy | Ham Accuracy | Spam Accuracy | Ham Accuracy | Spam Accuracy | Ham Accuracy |
| 100 | 0.03 | 86.92 | 96.55 | 84.61 | 97.70 | 77.69 | 93.96 |
| 80 | 0.03 | 86.92 | 96.55 | 84.61 | 97.70 | 80.0 | 93.91 |
| 60 | 0.03 | 86.92 | 96.55 | 84.61 | 97.70 | 77.69 | 92.81 |
| 40 | 0.03 | 86.92 | 96.55 | 85.38 | 97.70 | 53.07 | 95.11 |
| 20 | 0.03 | 86.92 | 96.55 | 95.38 | 60.05 | 30.0 | 95.68 |
| 100 | 0.09 | 86.92 | 96.55 | 84.61 | 97.70 | 81.53 | 92.52 |
| 80 | 0.09 | 86.92 | 96.55 | 84.61 | 97.70 | 80.76 | 92.81 |
| 60 | 0.09 | 86.92 | 96.55 | 84.61 | 97.70 | 80.76 | 92.24 |
| 40 | 0.09 | 86.92 | 96.55 | 86.15 | 97.70 | 70.76 | 91.95 |
| 20 | 0.09 | 86.92 | 96.55 | 95.38 | 65.51 | 31.53 | 96.55 |
| 100 | 0.27 | 86.92 | 96.55 | 84.61 | 98.27 | 83.07 | 93.56 |
| 80 | 0.27 | 86.92 | 96.55 | 84.61 | 98.27 | 83.07 | 92.81 |
| 60 | 0.27 | 86.92 | 96.55 | 84.61 | 98.27 | 83.07 | 92.24 |
| 40 | 0.27 | 86.92 | 96.55 | 86.92 | 98.27 | 74.76 | 92.52 |
| 20 | 0.27 | 86.92 | 96.55 | 95.38 | 65.51 | 40 | 95.61 |
| 100 | 0.81 | 86.92 | 96.55 | 84.61 | 98.27 | 62.30 | 95.97 |
| 80 | 0.81 | 86.92 | 96.55 | 84.61 | 98.27 | 82.30 | 93.10 |
| 60 | 0.81 | 86.92 | 96.55 | 84.61 | 98.27 | 80.76 | 93.39 |
| 40 | 0.81 | 86.92 | 96.55 | 86.92 | 98.27 | 76.92 | 92.52 |
| 20 | 0.81 | 86.92 | 96.55 | 95.38 | 63.50 | 37.69 | 95.97 |

When we remove the stop words, the number of features are also reduced. So, perceptron accuracy is reduced. Also, the logistic regression is calculated with lambda value to ‘0’. So, we do not have regularization in this case.

***Neural Networks:***

*I have downloaded weka jar file and executed the code with following parameters for out data sets.*

| ***Learning Rate*** | ***Momentum*** | ***Iteration*** | ***Hidden Layers*** | ***Hidden Units*** | ***Accuracy*** |
|---|---|---|---|---|---|
| *0.03* | *0.1* | *2* | *1* | *2* | *72.80* |
| *0.03* | *0.2* | *20* | *3* | *12* | *72.80* |
| *0.1* | *0.4* | *2* | *1* | *5* | *83.26* |
| *0.1* | *0.1* | *20* | *1* | *10* | *93.72* |
| *0.1* | *0.2* | *50* | *3* | *30* | *95.39* |
| *0.1* | *0.4* | *100* | *3* | *17* | *95.18* |
| *0.3* | *0.1* | *2* | *1* | *4* | *91.42* |
| *0.3* | *0.1* | *20* | *1* | *10* | *95.39* |
| *0.3* | *0.2* | *50* | *3* | *30* | *96.23* |
| *0.3* | *0.4* | *100* | *2* | *20* | *95.18* |
| *0.9* | *0.1* | *2* | *1* | *4* | *93.93* |
| *0.9* | *0.2* | *10* | *1* | *10* | *94.56* |
| *0.9* | *0.2* | *20* | *2* | *20* | *94.97* |
| *0.9* | *0.4* | *50* | *3* | *30* | *89.74* |
| *0.9* | *0.6* | *50* | *3* | *17* | *81.59* |
| | | | | | |

*As we can see from the above table, we get good accuracy for learning rate 0.1. Also, Momentum of 0.2 is also good for the dataset.*

# ***K means***

## *KOALA image:*

The original file is of ***size 762 kb*** for koala image. I have run the program multiple times and calculated the compression ratio with the help of formula

***Compression ratio = Input Image Size/Output Image Size***

| K value | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th | 10th | Average | Variance |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 6 | 6 | 5.95 | 6 | 6 | 6 | 5.95 | 5.95 | 6 | 6 | 5.985 | 0.000525 |
| 5 | 4.40 | 4.43 | 4.46 | 4.46 | 4.46 | 4.46 | 4.46 | 4.46 | 4.43 | 4.43 | 4.445 | 0.000405 |
| 10 | 4.64 | 4.70 | 4.67 | 4.67 | 4.64 | 4.70 | 4.67 | 4.64 | 4.70 | 4.67 | 4.67 | 0.00054 |
| 15 | 4.79 | 4.76 | 4.79 | 4.76 | 4.76 | 4.76 | 4.76 | 4.73 | 4.79 | 4.82 | 4.772 | 0.000576 |
| 20 | 4.88 | 4.79 | 4.85 | 4.79 | 4.85 | 4.85 | 4.79 | 4.88 | 4.79 | 4.85 | 4.832 | 0.001296 |

## *Penguins Image*

*The original image is of size 759kb.*

| K value | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th | 10th | Average | Variance |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 9.1 | 9.07 | 9.1 | 9.1 | 9.1 | 9.1 | 9.1 | 9.07 | 9.1 | 9.07 | 9.091 | 0.000189 |
| 5 | 7.16 | 7.51 | 7.51 | 7.16 | 7.44 | 7.51 | 7.51 | 7.51 | 7.51 | 7.51 | 7.433 | 0.019061 |
| 10 | 6.71 | 6.71 | 6.66 | 6.66 | 6.71 | 6.66 | 6.66 | 7.09 | 6.78 | 6.66 | 6.73 | 0.01578 |
| 15 | 6.78 | 6.54 | 6.6 | 6.66 | 6.96 | 6.49 | 6.54 | 6.54 | 6.66 | 6.54 | 6.631 | 0.018609 |
| 20 | 6.78 | 6.66 | 6.6 | 6.78 | 6.6 | 6.9 | 6.96 | 6.72 | 6.9 | 6.78 | 6.768 | 0.014256 |

***Q.*** *Display the Images after data compression using K-means clustering for different values of K(2,5,10,15,20).*

### *Original KOALA.JPG image*

### *Compressed images Images for KOALA.jpg*

***K = 2***

***K = 5***

***K=10***

***K = 15***

***K = 20***

***Original Penguines.jpg image***

***Compressed images for penguins.jpg***

***K = 2***

***K = 5***

***K = 10***

***K=15***

***K = 20***

*Q.* Is there a tradeoff between image quality and degree of compression? What would be a good value of K for each of the two images?

Ans: As K value increases, the image quality also increases and image compression reduces. Out of all the values, the K=20 value is good as the image quality is good for both koala and penguin images.