Date: 02/24/19

CSE6363 - Machine Learning

Project 1 - Spring 2019

SUNDESH RAJ
1001633297

Linear Regression :-

1)

(a) The implementation for the polynomial fit solver is done in Python and can handle polynomial fits of different order (1 to 4).

(b) After applying the regression learner from 1 a) we obtain the following equations for different orders.

(i) 1st Order :-

$$\hat{y} = 1.74277 + 3.12417x_1 + 3.0584x_2$$

(ii) 2nd Order :-

$$\hat{y} = 4.41437 + 1.97318x_1 + 2.95898x_2 + 0.097x_1^2 + 0.0010x_2^2 + 0.0185x_1x_2$$

(iii) 3rd Order :-

$$\hat{y} = 4.071 + 2.23299x_1 + 2.95019x_2 + 0.0479x_1^2 + 0.0088x_2^2 + 0.00783x_1x_2 + 0.00072x_1^2x_2 + 0.00025x_1x_2^2 + 0.0027x_1^3 - 0.000609x_2^3$$

(iv) 4th Order :-

$$\hat{y} = 4.10528 + 2.422x_1 + 2.761x_2 - 0.0686x_1^2 + 0.0414x_2^2 + 0.0839x_1x_2 - 0.00688x_1^2x_2 - 0.0074x_1x_2^2 + 0.022x_1^3 - 0.0032x_2^3 - 0.0010x_1^4 + 0.00012x_2^4 + 0.00026x_1^2x_2^2 + 0.00026x_1^3x_2 + 0.00026x_1x_2^3$$

c) By evaluating our polynomial regression function there are minor changes in the actual value of Y and the predicted value $\hat{Y}$

Below we consider some of the data points and determine if they are overfitting and decide which order of polynomial function is more suitable.

→ consider the input dataset $X_1 = 8.0101$, $X_2 = 8.8651$
$Y = 54.1331$

1st order prediction = 53.8806, error = −0.2525
2nd order prediction = 54.0735, error = −0.0596
3rd order prediction = 54.0148, error = −0.1183
4th order prediction = 54.1642, error = 0.0311

From the above values we can see that the 4th order polynomial has the lowest error range. There seems to be no overfitting, if we would consider a higher order polynomial overfitting might occur.

→ consider input dataset $X_1 = 0.2922$, $X_2 = 0.2867$
$Y = 5.7326$

1st order prediction = 3.5324, error = −2.2001
2nd order prediction = 5.8492, error = 0.1166
3rd order prediction = 5.5757, error = −0.1568
4th order prediction = 5.6093, error = −0.1232

From the above mentioned error values we can see that the 2nd order polynomial function provides the lowest error. In this case there is no overfitting, although if we were to consider a higher order polynomial function overfitting might occur.

→ Consider the input data $x_1 = 9.2885$, $x_2 = 4.8990$

$Y = 46.3750$

1st order prediction = 45.7447, error = −0.6302

2nd order prediction = 46.4787, error = 0.1037

3rd order prediction = 46.4802, error = 0.1052

4th order prediction = 46.5942, error = 0.2192

From the above values we can see that the 2nd order polynomial function has the lowest error range. In this case there is no significant overfitting.

② Logistic Regression

a) The logistic regression classifier is written in Python

b) The classifier is then applied to the given test data to make the prediction.

c) (i) Differences between Naive Bayes and Logistic Regression

→ Basically Naive Bayes is a generative model and Logistic Regression is a discriminitive model.

→ In Naive Bayes, it models the joint distribution of the feature X and target Y and then predicts the posterior probability given as $P(Y/X)$.

→ The logistic regression directly models the posterior probability of $P(Y/X)$ by learning the input to output mapping by minimising the error.

→ When there are huge datasets a discriminative model such as logistic regression pefoms better than a generative model such as naive Bayes.

→ In our case the dataset is small and hence we might consider Naive Bayes over Logistic regression. Although, naive Bayes assumes the input data to be conditionally independant, which is not the case in real. Moreover, Naive Bayes classifier converges quicker but has a higher error than Logistic Regression.

(ii) Differences between KNN and Logistic Regression.

→ KNN classifier is much faster to train because it does not really involve any training, but its predictions are much slower when compared to logistic Regression.

→ In our case KNN does not peform well with small number of observations, where as Logistic peforms well with small or large datasets.

3) Linear Discriminant Analysis

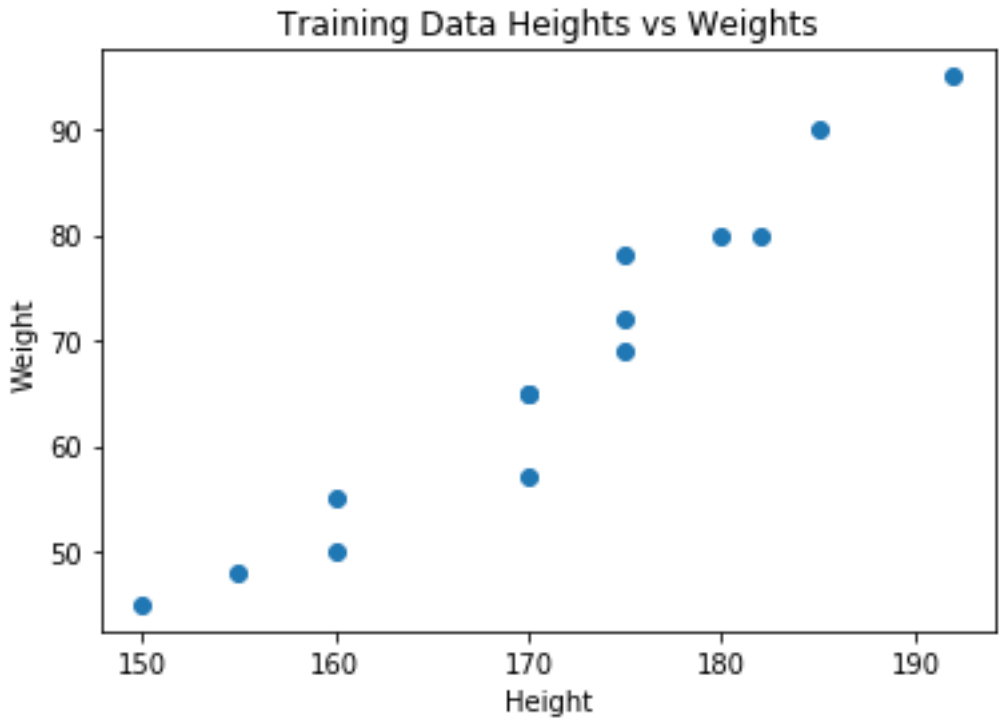
a) The Linear Discriminant Analysis implementation is done in Python for the training data provided in Question 2.

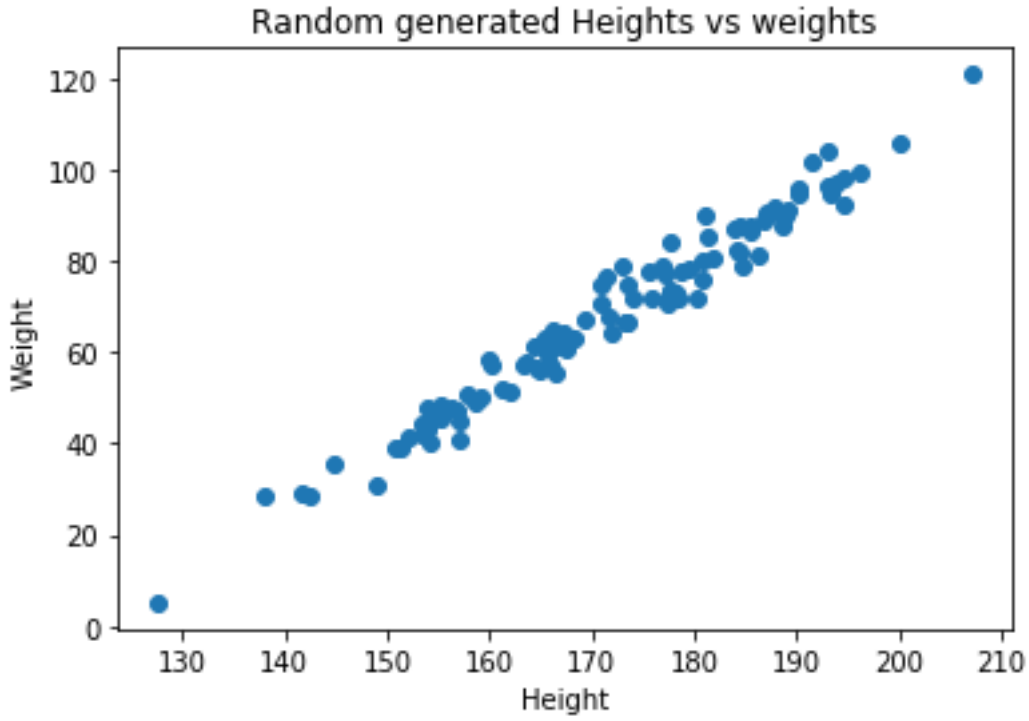
b) The classification results obtained in Question 2 and 3 are the same for the given test data set.
The Logistic Regression is based on Maximum Likelihood estimation and the LDA is based on Least squares estimation.
The decision boundary of LDA is based on the Pooled covariance matrix with respect to men and women.
But LDA is more prone to outliers because the covariance matrix is estimated by taking into account all data points.
But Logistic Regression apparently down weights points far from the decision boundary.

c) Based on the mean generated for Men and women, and the pooled covariance matrix, a number of samples are generated for data with respect to men and women.

Given below are the graphs plotted for heights vs weights for training data and random generated data.

We can observe that in the training data although the points are scattered they are linear.
In the random generated data graph the points appear more linear but there seems to be few outliers in the graph.


Training Data Heights vs Weights
Weight
Height
90
80
70
60
50
150
160
170
180
190


Random generated Heights vs weights
Weight
Height
120
100
80
60
40
20
0
130
140
150
160
170
180
190
200
210