CSE6363 Project 1

Manish Munikar
(1001826846)

March 9, 2021

1. Linear Regression

(a)

A polynomial linear regression solver is implemented in `linear_regression.py`. Instructions on how to run the code is given in `README.txt`. It uses a linear function of the input features as the hypothesis space, sum of squared error as the performance metric, and uses analytic solution to optimize the performance.

(b)

The surface plots for the fitted model of orders 1, 2, 3, and 4 are shown in the Fig. 1. For order 1, the model surface is a flat plane. For orders higher than 1, the surface is curved. Even though the range of x and y was between 0 and 10 in the dataset, the curvature of the model's function is not clearly visible within those range. That's why I have used a range of -10 to 20 for x and y.

(c)

The error on the training set and the test set for various order of polynomials are as follows:

Order	Train error	Test error
1	15.7243	9.5403
2	0.7338	0.1882
3	0.6684	0.2546
4	0.5851	0.3409

Training error decreases monotonically as we increase the model complexity (by increasing the polynomial order). However, the test error decreases up to 2nd order and then if we increase the order, the test error start increasing. From this, we can say that at order 1, the model is underfit, and for order greater than 2, the model is overfit. The model with a polynomial order 2 seems to be the best model for this dataset.

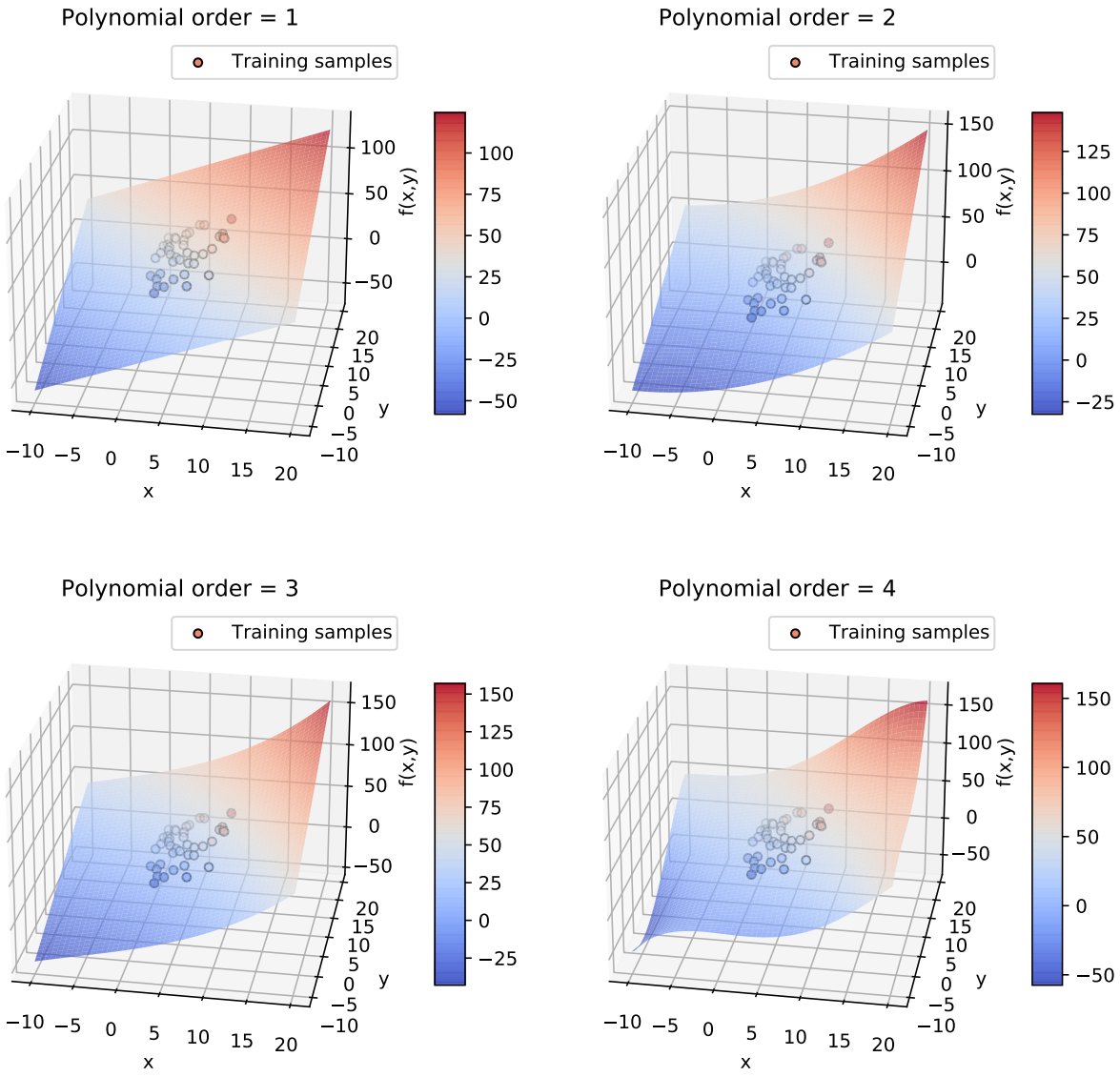

Figure 1: The results of the polynomial linear regression model of various orders.

2. Logistic Regression

(a)

A logistic regression classifier is implemented in `logistic_regression.py`. Instructions on how to run the code is given in `README.txt`. In this implementation, the hypothesis space includes the sigmoid function on all the linear transformation of the input features (height, weight, and age). Similarly, the performance metric is the log likelihood of the parameters, given the training set. It uses batch gradient descent as the optimizer.

(b)

After training with the given training set (with output being 1 for M and 0 for W), the prediction for the test set is as follows:

Input	Actual class	Predicted Probability	Predicted Class
$(155, 40, 35)$	0 (W)	0.00	0 (W)
$(170, 70, 32)$	1 (M)	0.99	1 (M)
$(175, 70, 35)$	0 (W)	0.35	0 (W)
$(180, 90, 20)$	1 (M)	0.99	1 (M)

The weights for the input features (constant/bias, height, weight, and age) are 0.138, -1.620, 4.657, and -1.237, respectively. From these weights, we can say that the constant/bias term is the very insignificant to classify gender. Weight and height are the most important features to classify gender.

When we used KNN and Gaussian Naive Bayes classifiers, the predicted class for these data points, respectively, were: 0, 1, 1, 1. Actually, that is also what Logistic Regression predicts when the weights are initialized with some different small random numbers. But with appropriate weight initialization and sufficient training, Logistic Regression can make better predictions if the decision boundary is somewhat linear with respect to the input features. Men are usually taller and heavier than women, which is very effectively captured by the discriminative Logistic Regression model.

KNN uses the concept of *similarity* (of *distance*) from a few neighboring points. It doesn't consider all dataset when making a prediction. On the other hand, Gaussian Naive Bayes is a generative model that learns the probability of input features, assuming they have Gaussian distribution, which may not be true in all cases. Since Logistic Regression is a strictly discriminative classifier with very few assumptions, it can make better classifications.

3. Linear Discriminant Analysis

(a)

The implementation is provided in `linear_discriminant_analysis.py`. Instructions on how to run the code is given in `README.txt`.

(b)

The classification result is as follows:

Input	LDA Prediction	Logistic Prediction
$(155, 40, 35)$	0 (W)	0 (W)
$(170, 70, 32)$	1 (M)	1 (M)
$(175, 70, 35)$	1 (M)	0 (W)
$(180, 90, 20)$	1 (M)	1 (M)

The decision boundary for logistic regression are the points where the predicted probability is 0.5:

$$\begin{aligned} & p(y = 1 \mid x) = h(x) = 0.5 \\ \implies \quad & \sigma(\theta^T x) = 0.5 \\ \implies \quad & \theta^T x = 0 \end{aligned}$$

If we use the weights from §2b, we get:

$$\begin{aligned} & 0.138 - 1.620h + 4.657w - 1.237a = 0 \\ \implies \quad & 1.62h - 4.657w + 1.237a = 0.138 \end{aligned} \tag{1}$$

where h, w, and a represent height, weight, and age, respectively.

Similarly, the decision boundary for LDA lies where the log likelihood of both classes are equal:

$$\begin{aligned} & \log\big(p(y = 0 \mid x)\big) = \log\big(p(y = 1 \mid x)\big) \\ \implies \quad & (x - \mu_0)^T \Sigma^{-1} (x - \mu_0) + \log|\Sigma| + c_1 = (x - \mu_1)^T \Sigma^{-1} (x - \mu_1) + \log|\Sigma| + c_2 \\ \implies \quad & -2\mu_0^T \Sigma^{-1} x + 2\mu_1 \Sigma^{-1} x = c + -\mu_0^T \Sigma^{-1} \mu_0 + \mu_1^T \Sigma^{-1} \mu_1 \\ \implies \quad & (\mu_1 - \mu_0)\Sigma^{-1} x = \frac{1}{2}(c + \mu_1^T \Sigma^{-1} \mu_1 - \mu_0^T \Sigma^{-1} \mu_0) \end{aligned}$$

where c_1, c_2, and c are some constant terms.

The parameters for this particular training set are:

$$\begin{aligned} \mu_0 &= \begin{pmatrix} 162.86 & 55.57 & 30.0 \end{pmatrix} \\ \mu_1 &= \begin{pmatrix} 179.86 & 80.0 & 30.14 \end{pmatrix} \\ \Sigma &= \begin{pmatrix} 58.27 & 63.11 & -4.56 \\ 63.11 & 77.84 & -4.36 \\ -4.56 & -4.36 & 4.06 \end{pmatrix} \\ \Sigma^{-1} &= \begin{pmatrix} 0.15 & -0.12 & 0.04 \\ -0.12 & 0.11 & -0.02 \\ 0.04 & -0.02 & 0.27 \end{pmatrix} \end{aligned}$$

So the decision boundary becomes,

$$-0.35h + 0.62w + 0.3a = \frac{1}{2}(c + 2663.95 - 2683.03)$$

$$\implies \quad -0.35h + 0.62w + 0.3a = c - 19.08 \tag{2}$$

Now, the boundary has to pass through the midpoint of the centroids, i.e.

$$\frac{\mu_0 + \mu_1}{2} = \begin{pmatrix} 171.36 & 67.79 & 30.07 \end{pmatrix}$$

By using this value in Eq. (2), we can get the value for the constant c,

$$-0.35(171.36) + 0.62(67.79) + 0.3(30.07) = c - 19.08$$

$$\implies \quad -9.54 = c - 19.08$$

$$\implies \quad c = 9.54$$

Therefore, the LDA decision boundary is,

$$-0.35h + 0.62w + 0.3a = -9.54 \tag{3}$$

From this we can see that Logistic Regression and LDA both produce linear boundary (polynomial of order 1 with respect to the input features), but may be different. Particularly, the LDA boundary passes through the midpoint of the centroids of two classes and is perpendicular to the line joining the two centroids, whereas the Logistic Regression boundary doesn't have the concept of centroids; it just tries to minimize the cross-boundary misclassifications.

(c)

The plot showing both the generated samples and the actual data points from the training set for each of the classes is shown in Fig. 2. Since the heights and weights of people are actually approximately Normally distributed in real world, this LDA model can capture the distribution parameters pretty accurately from the training samples.

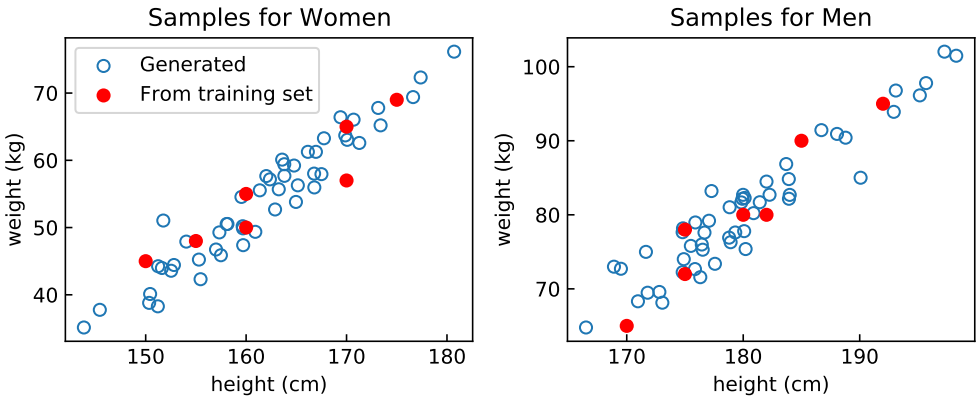

Figure 2: Generated samples and actual data points for the Women/Men classification problem.

Since the synthetic data points are generated from the distribution which is learned from the training set itself, there won't be much difference in the distributions (means and variances) of the training samples and the generated samples. However, we must note that in LDA, we make the homoscedastic assumption, i.e., we assume that the covariances of all classes are identical, which might not hold true for the real training data.