Megan Chu
A12814536
Assignment 5
6/11/18

Q1.
Note that there exist floating point errors in many of our dist values. For example, -0.9460000000000003 should actually be just -0.946 but python stores it as -0.9460000000000003 which is the closest value it can be stored as. TO account for this, we used the round() method in python to round each dist value to 10 decimal places (for security to get all significant decimal places) to get a more accurate dist value. This would possibly introduce minor rounding errors, but not as much as floating point errors would cause.

Program Output:
```
Running Q1(calculate distance from hyperplane)...
Beta = [-0.15,0.9,0.05,-0.02]^T
Beta0 = 0.88
gene1   gene2   gene3   gene4   label   dist
7.4     1.0     6.2     8.3     -       -0.946
7.6     0.1     7.4     0.5     -       -1.57
7.54    2.6     6.3     1.3     -       0.618
7.4     1.0     4.6     2.0     -       -0.9
7.34    1.1     1.3     2.2     -       -0.97
6.01    0.62    1.0     4.2     -       -1.2575
5.8     0.7     7.7     0.9     -       -0.753
5.2     0.8     9.8     9.6     -       -0.642
5.1     0.62    3.3     5.5     -       -1.032
4.9     2.55    0.5     6.1     -       0.583
4.9     1.4     5.0     3.0     -       -0.165
4.6     0.5     7.3     2.7     -       -0.809
4.5     1.22    6.6     7.5     -       -0.277
4.1     0.4     3.3     9.3     -       -1.156
4.12    0.5     4.0     5.9     -       -0.966
3.9     0.7     0.3     7.4     -       -0.968
3.85    0.84    6.1     5.7     -       -0.5105
3.6     0.9     4.4     0.5     -       -0.4
1.8     0.7     6.8     4.0     -       -0.26
1.9     0.61    5.1     2.0     -       -0.401
1.94    0.62    8.7     3.0     -       -0.238
2.05    0.8     8.5     6.7     -       -0.1765
2.1     0.75    3.9     7.1     -       -0.467
2.2     0.8     4.6     8.6     -       -0.432
2.25    1.15    6.5     1.9     -       0.1045
2.4     0.65    0.4     6.5     -       -0.765
2.5     0.9     6.2     9.5     -       -0.325
2.8     0.3     1.5     4.6     -       -1.047
3.2     0.2     9.7     7.4     -       -0.843
2.7     0.6     6.4     5.2     -       -0.529
3.3     0.7     7.9     2.8     -       -0.406
5.8     2.4     3.8     2.6     +       0.548
0.8     0.8     1.2     7.5     +       -0.37
```

0.76	1.1	5.7	4.2	+	0.197
0.8	1.3	1.3	9.0	+	0.055
0.9	1.4	0.6	9.3	+	0.089
1.6	2.4	1.1	6.9	+	0.957
1.8	2.45	7.3	1.2	+	1.396
1.9	2.8	3.4	2.5	+	1.475
1.93	2.9	7.1	4.0	+	1.7155
2.1	2.7	5.3	8.0	+	1.34
2.1	3.3	0.6	2.6	+	1.753
2.55	3.2	0.3	5.8	+	1.5165
2.6	3.3	0.2	0.7	+	1.696
4.2	3.6	9.7	7.1	+	2.073
0.8	3.8	4.3	9.0	+	2.455
0.76	3.9	8.5	3.5	+	2.871
0.8	4.7	4.3	8.3	+	3.279
1.1	5.0	0.2	0.4	+	3.457
0.9	6.4	3.3	1.5	+	4.88
1.7	6.6	0.8	7.3	+	4.699
2.0	6.7	3.5	9.1	+	4.843
1.9	7.4	2.2	9.9	+	5.407

Point closest to hyperplane:

```
0.8     1.3     1.3     9.0     +       0.055
```

Answer:
After calculating the distance of each point to L, we found the closest point to L by finding the point with minimum |dist|. Thus the point closest to L was determined to be [gene1,gene2,gene3,gene4] = [0.8,1.3,1.3,9.0] which is labelled as a "+" point and has a distance of exactly .055 from L.

Q2.

Program Output1:

```
Running Q2(Misclassified points and classification error)...
gene1   gene2   gene3   gene4   label   dist
7.54    2.6     6.3     1.3     -       0.6180000000000001      Misclassified
4.9     2.55    0.5     6.1     -       0.583   Misclassified
2.25    1.15    6.5     1.9     -       0.1044999999999999      Misclassified
0.8     0.8     1.2     7.5     +       -0.3699999999999999     Misclassified
Number of misclassified points: 4
Classification error: 1.6754999999999998
```

Note that we again see floating point errors in the dist values. Thus the round() method was used to round each dist to 10 decimal places when calculating the classification error. The final classification error was also rounded to 10 decimal places.

Program Output2:

```
Running Q2(misclassified points and classification error)...
Beta = [-0.15,0.9,0.05,-0.02]^T
Beta0 = 0.88
gene1   gene2   gene3   gene4   label   dist
7.54    2.6     6.3     1.3     -       0.618   Misclassified
4.9     2.55    0.5     6.1     -       0.583   Misclassified
2.25    1.15    6.5     1.9     -       0.1045  Misclassified
0.8     0.8     1.2     7.5     +       -0.37   Misclassified
Number of misclassified points: 4
Classification error: 1.6755
```

Answer:

We defined all misclassified points as any points with the "+" label that had dist < 0 or any points with the "-" label that had dist > 0. This gave us a total of 4 misclassified points. We then calculated the classification error by summing up |dist| for all the misclassified points and obtained a classification error of exactly 1.6755 after eliminating the floating point errors.

Q3.
Note that Beta values were calculated using all combinations of -1 <= Bi <= 1, where 1 <= i <= 4 and Bi was incremented by a step size of 0.1. The Beta values were then normalized by dividing by their $||B||_2$ values, and the normalized Beta values (BetaNorm) rather than the original Beta values were used to calculate and find the minimum classification error. Also note that we tried out all B_0 values 0 <= B_0 <= 1 in increments of 0.1 for each BetaNorm value to find the minimum classification error.

Program Output:

```
Running Q3(grid search)...
Beta = [-0.3,0.9,0,0]^T
BetaNorm = [-0.31622776601683794,0.9486832980505139,0.0,0.0]^T
Beta0 = 0.5
Minimum Classification Error = 0.42690748412273116
```

Answer:
We found the BetaNorm value (~[-.316,.949,0,0]) and Beta0 (.5) values that gave us the minimum classification of ~.427, noting that there may be floating point errors from python that make our answer slightly off from the actual value. However, it is hard to manage/minimize floating point errors in this problem because it is possible BetaNorm will hold irrational values after division by $||B||_2$.

Q4.
Note that the standard equation for variance in a sample requires dividing the sum of (differences from the mean)^2 by the degrees of freedom. However, the equation for scatter within a sample given by the slides in class does not include dividing by degrees of freedom, thus our output is also calculated without dividing the variances by degrees of freedom. Also note that the F values we calculated using the code are trivially off from the F values that were calculated by hand. This is due to floating point errors in python that give inexact values between calculations, and it is hard to manage/minimize floating point errors for this question since many small calculations are needed to calculate the F function.

Program Output:

```
Running Q4(calculate F function)...
Beta = [-0.4,0.45,0.01,0]^T
        F = 0.12734258535188633
Beta = [-0.15,1,0.1,0.2]^T
        F = 0.08279694667738216
```

Answer:
Since Beta = [-0.4,0.45,0.01,0]^T has an F-value of ~.127 which is clearly greater than the F-value of ~.083 of Beta = [-0.15,1,0.1,0.2]^T, we can conclude that the vector **Beta = [-0.4,0.45,0.01,0]^T is a better choice for clustering the points**. We see that for F calculated by hand, we get F = exactly 0.127342585351886 for Beta = [-0.4,0.45,0.01,0]^T and F = exactly 0.0827969466773821 for Beta = [-0.15,1,0.1,0.2]^T which aligns with the conclusion that Beta = [-0.4,0.45,0.01,0]^T is a better choice for clustering the points. Note that

F = [(mean1-mean2)^2]/(sum of variances).

Thus, a better classification will have a greater difference in means (greater numerator) and a smaller variances within each class (smaller denominator). Thus, a better classification will naturally have a greater F value.