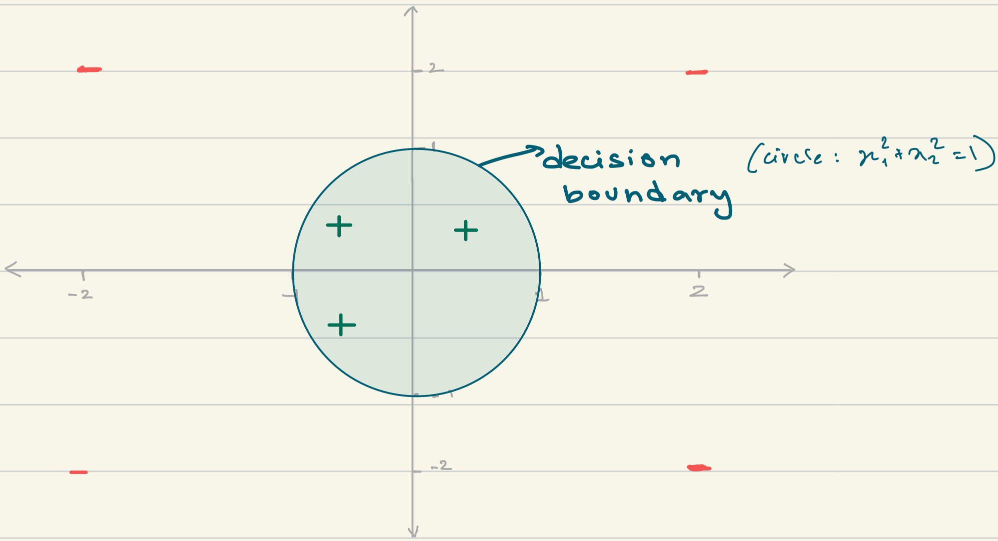
Homework 3

Jyotsna Rajaraman

Q1.

(a)



(b) The boundary is a circle $\rightarrow$ $x_1^2 + x_2^2 = 1$

The parameters are $\rightarrow$ $\theta_0, \theta_1, \theta_2$

hyp func,

$$h_\theta(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1^2 + \theta_2 x_2^2)}}$$

$$L(\theta_0, \theta_1, \theta_2) = \prod \left[h_\theta(x)^y * (1 - h_\theta(x))^{1-y} \right]$$

we need to maximize

(a)

Q2. RSE → Residual standard Error

↳ $RSE = \sqrt{RSS/n-2}$

RSE is a measure of the avg amount by which the actual value deviates from the values predicted by the linear regression model → calculated by RSS (residual sum of squares)/ degrees of freedom.

Advantage of RSE: Accounts for variability of the residuals and provides the estimate of S.D for errors of the model & indicates a good fit.

R^2 → R^2 statistic

↳ $R^2 = 1 - RSS/TSS$

R^2 is a measure of the proportion of total variability in the models response variable.

Advantage of R^2: It can provide a good picture of the proportion of variability and assess the performance on a fixed scale ∵ it varies from 0 to 1 → 1 being the best fit

(b) For simple linear regression

i/p $\rightarrow x$

o/p $\rightarrow y$.

$$y = \text{intercept} + \text{slope} \times x + \varepsilon$$

regression coeff error.

$\hookrightarrow$ $y = \beta_0 + \beta_1 \cdot x + \varepsilon \sim \beta_0 + \beta_1 x$

$R^2 \rightarrow$ coeff of determination

$r \rightarrow \text{Cor}(X, Y) \rightarrow$ correlation.

We need to show that R^2 = squ. correlation

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

$$r = \text{Cor}(X, Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

W.K.T $TSS = \sqrt{(y_i - \bar{y})^2}$

$$RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\therefore R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

$$r^2 = \left(\frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}} \right)^2$$

$$= \frac{\left[\sum_{i=1}^{n} (x_i - \bar{x})\right]^2 \left[\sum_{i=1}^{n} (y_i - \bar{y})\right]^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2 \cdot \sum_{i=1}^{n} (y_i - \bar{y})^2}$$

$$\because \hat{y}_i = \beta_0 + \beta_1 x_i$$

$$r^2 = \frac{\left[\sum_{i=1}^{n} (x_i - \bar{x}_i)\right]^2 \left[\sum_{i=1}^{n} (\beta_0 + \beta_1 x_i - \bar{y})\right]^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2 \cdot \sum_{i=1}^{n} (y_i - \bar{y})^2}$$

$$r^2 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2 - \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} = R^2$$

Q8) $P(Y=k|x) = \dfrac{e^{w_k^T x}}{\sum_{j \in \text{classes}} e^{w_j^T x}}$

predicted class $\rightarrow \hat{y} = \text{argmax}_k \{P(Y=k|x)\}$.

P.T $\rightarrow$: linear classifier & class boundaries are linear functions of i/p attributes in x

For class boundary between 2 classes say i, k where $i \neq k$ $\rightarrow$ set eq prob and solve for x

$$P(Y=k|x) = P(Y=i|x)$$

$$\frac{e^{w_k^T x}}{\sum_{j \in \text{classes}} e^{w_j^T x}} = \frac{e^{w_i^T x}}{\sum_{j = \text{classes}} e^{w_j^T x}}$$

$\because$ denom = same

$$e^{w_k^T x} = e^{w_i^T x}$$

$$w_k^T x \cdot (\ln e) = w_i^T x \, (\ln e)$$

$$\Rightarrow \quad w_k^T x = w_i^T x$$

$$\Rightarrow \quad (w_k^T - w_i^T)\, x = 0$$

↳ This is a linear function of x & a class boundary.

$\therefore$ For any classes $(k, i) \in$ Classes such that $k \neq i$ we can see that class boundaries are linear functions of i/p attributes in x

$\therefore$ It is a linear classifier

Q4)

Data:

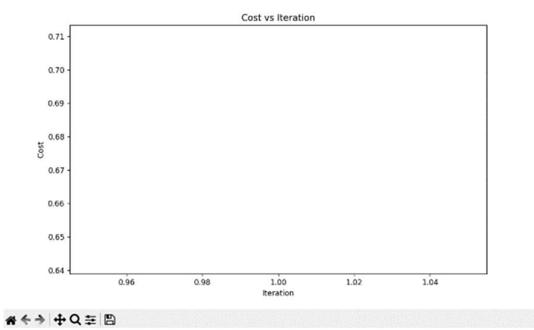
```
X = [[0.31885351, -1.60298056], [-1.53521787, -0.57040089], [-0.2167283, 0.2548743],
     [-0.14944994, 2.01078257], [-0.09678416, 0.42220166], [-0.22546156, -0.63794309],
     [-0.0162863,  1.04421678], [-1.08488033, -2.20592483], [-0.95121901,  0.83297319],
     [-1.00020817,  0.34346274]]
y = [0, 0, 0, 1, 0, 0, 1, 0, 0, 0]
```

Parameters:

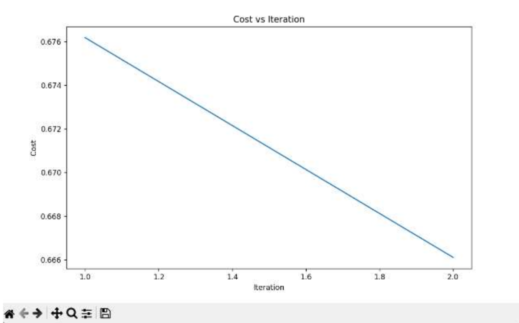
alpha = 0.1

lamb = 1

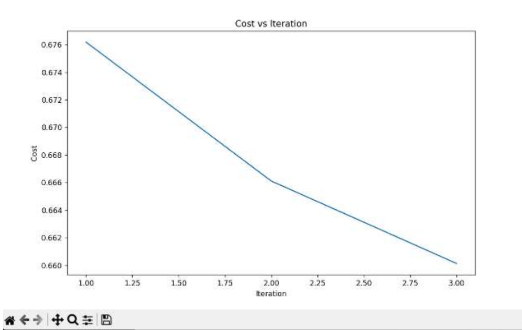
Cost vs Iteration Graphs:



For number of iterations = 1



For number of iterations = 2



For number of iterations = 3

Q5)

Parameters:

Learning rate, alpha = 0.01

Regularization strength, lamb = 1

Number of iterations, T = 1000

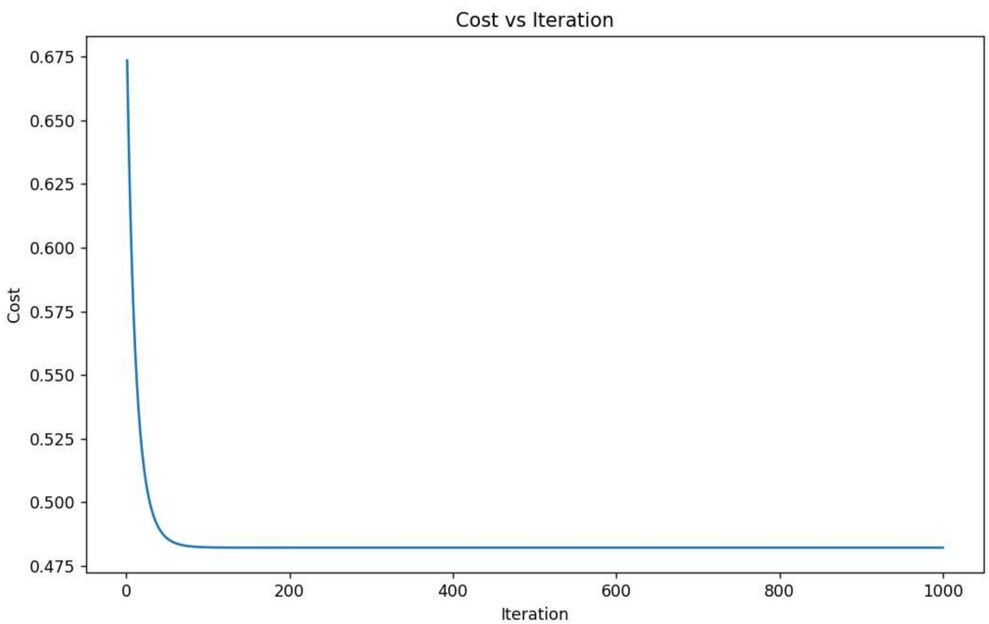
Initial parameter vector, theta_init = np.zeros(n + 1)

Additionally, input data was normalized.

Output theta values:

```
[ 0.05115606 -0.08154328 -0.05227694 -0.08211722 -0.0780795 -0.03546892
 -0.05523512 -0.07010991 -0.08345708 -0.03139901 0.01357594 -0.06000551
  0.00245318 -0.05663162 -0.05670919 0.00921041 -0.01630741 -0.01209534
 -0.03385924 0.00598137 0.00688884 -0.0884145 -0.05976276 -0.08779673
 -0.08186825 -0.05057787 -0.06038826 -0.06879334 -0.08762994 -0.05013313
 -0.03126501]
```

Cost vs Iteration Graph:



Q6)

Output theta values – for my implementation:

```
[ 0.05115606 -0.08154328 -0.05227694 -0.08211722 -0.0780795 -0.03546892
 -0.05523512 -0.07010991 -0.08345708 -0.03139901 0.01357594 -0.06000551
  0.00245318 -0.05663162 -0.05670919 0.00921041 -0.01630741 -0.01209534
 -0.03385924 0.00598137 0.00688884 -0.0884145 -0.05976276 -0.08779673
 -0.08186825 -0.05057787 -0.06038826 -0.06879334 -0.08762994 -0.05013313
 -0.03126501]
```

Output theta values – for scikit-learn Logistic Regression:

```
[ 0.21456573 -0.36317072 -0.38770299 -0.35111444 -0.43555896 -0.16176493
  0.56259548 -0.85989759 -0.96222175 0.07616775 0.32218585 -1.29103479
  0.26892526 -0.65988685 -1.01250087 -0.27717046 0.7362824   0.11052647
 -0.3335099   0.29590225 0.68091528 -1.02936114 -1.31459312 -0.82331706
 -1.01059356 -0.67073135 0.04465888 -0.87330064 -0.91195815 -0.88789694
 -0.47983137]
```

Possible reasons for the difference:

1. The **alpha (learning rate)** that I chose to specify is probably different from the optimized way that scikit-learn’s implementation of alpha
2. The **number of iterations** used in my implementation of gradient descent may be different from what was used by scikit-learn's LogisticRegression.
3. The **cost function** used for ‘l2’ penalty is slightly different between the two. Sci-kit learn adds (C/2)*∑(θ_j^2) whereas our function adds 1/C*∑(θ_j^2) (note: C = 1/lamb)

Q7) Scatter plot: malignant cases: red
benign cases: green
decision boundary: blue