1. (A) Murphy Exercise 8.1, as expanded and explained below. Note that in this dataset, $y_i = 1$ denotes spam.

Exercise 8.1 Spam classification using logistic regression

Consider the email spam data set discussed on p300 of (Hastie et al. 2009). This consists of 4601 email messages, from which 57 features have been extracted. These are as follows:

- 48 features, in $[0, 100]$, giving the percentage of words in a given message which match a given word on the list. The list contains words such as "business", "free", "george", etc. (The data was collected by George Forman, so his name occurs quite a lot.)
- 6 features, in $[0, 100]$, giving the percentage of characters in the email that match a given character on the list. The characters are ; ([! $ #
- Feature 55: The average length of an uninterrupted sequence of capital letters (max is 40.3, mean is 4.9)
- Feature 56: The length of the longest uninterrupted sequence of capital letters (max is 45.0, mean is 52.6)
- Feature 57: The sum of the lengts of uninterrupted sequence of capital letters (max is 25.6, mean is 282.2)

Load the data from `spamData.mat`, which contains a training set (of size 3065) and a test set (of size 1536).

One can imagine performing several kinds of preprocessing to this data. Try each of the following separately:

a. Standardize the columns so they all have mean 0 and unit variance.
b. Transform the features using $\log(x_{ij} + 0.1)$.
c. Binarize the features using $\mathbb{I}(x_{ij} > 0)$.

For each version of the data, fit a logistic regression model. Use cross validation to choose the strength of the ℓ_2 regularizer. Report the mean error rate on the training and test sets. You should get numbers similar to this:

method	train	test
stnd	0.082	0.079
log	0.052	0.059
binary	0.065	0.072

(viii) For model selection (choosing your value of **λ**), use 5-fold cross validation, and run the cross-validation 5 times, taking average validation errors over the multiple runs for each value of **λ**. Note that at each run you should partition the given training set randomly.

(ix) **Report your selected value of λ and explain why you chose that particular λ.** This value of λ defines your "selected model".

(x) After choosing your value of λ, train again using all the training data, and then test using the test data. **Report on the following classification errors for your selected value of λ.** Please use a table like the example in Murphy except with 5 columns instead of 3 as follows:

Column 1: preprocessing method
Column 2: value of **λ**
Column 3: average cross-validation error from the validation sets.
Column 4: error on the full given training set (trained on the full given training set)
Column 5: error on the full given test set (trained on the full given training set)

```
When we standardized the data as the preprocessing method
we have our best λ= 0.016768329368110083
average cross-validation error from the validation sets=0.07614981796237275
error on the full given training set= 0.07177814029363783
error on the full given test set= 0.09505208333333337

When we transformed the features using log(xij + 0.1) as the preprocessing method
we have our best λ=2.6826957952797246
average cross-validation error from the validation sets=0.05735620948026532
error on the full given training set= 0.05579119086460027
error on the full given test set= 0.05729166666666663

When we binarized the features using I(xij > 0) as the preprocessing method
we have our best λ=1.151395399326447
average cross-validation error from the validation sets=0.057551863853518404
error on the full given training set= 0.06329526916802608
error on the full given test set= 0.072265625
```

I extracted 50 points as our λ from 0.001 to 1000 and used 5-fold cross-validation for 5 times. We averaged 5 errors for each λ and took the one that had the lowest value as chosen λ.

Method	λ
Standardize the columns	0.016768
Transform the features	2.682696
Binarize the features	1.151396

After choosing the value of λ, we trained again using all the training data and then test using the test data.

Method	λ	average cross-validation error	error on the full training set	error on the full test set
Standardize the columns	0.016768	0.076	0.072	0.095
Transform the features	2.682696	0.057	0.056	0.057
Binarize the features	1.151396	0.057	0.063	0.072

(B) Additional Question:

This problem pertains only to the given test data, as provided with the dataset. After using preprocessing method (c) on the given test data, use **sum of features 1-48 (total count of keywords in percentage)** as x axis, and **sum of features 49-54 (total count of special characters in percentage)** as y axis, and draw the following plots:

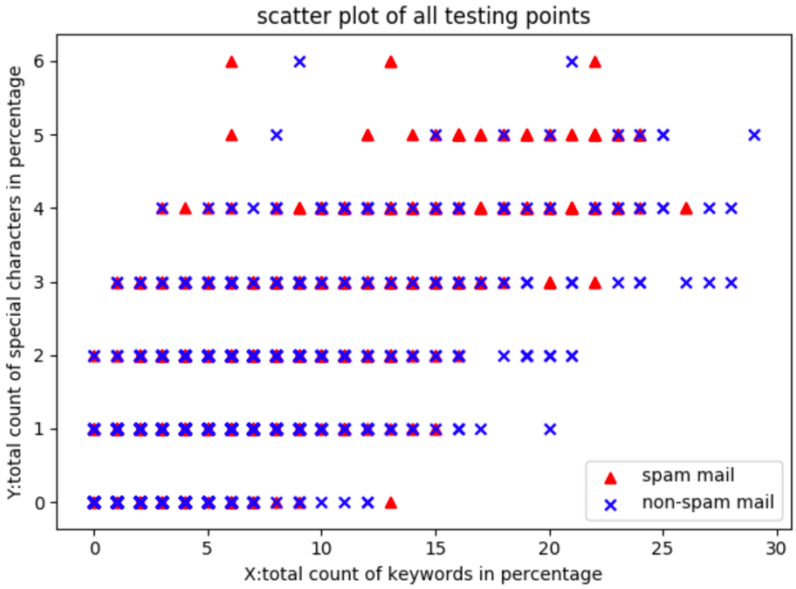
(i) A scatter plot of all testing points, using different colors for spam and non-spam emails.

(ii) For emails labeled spam, generate a 3D histogram using function `hist3()`.

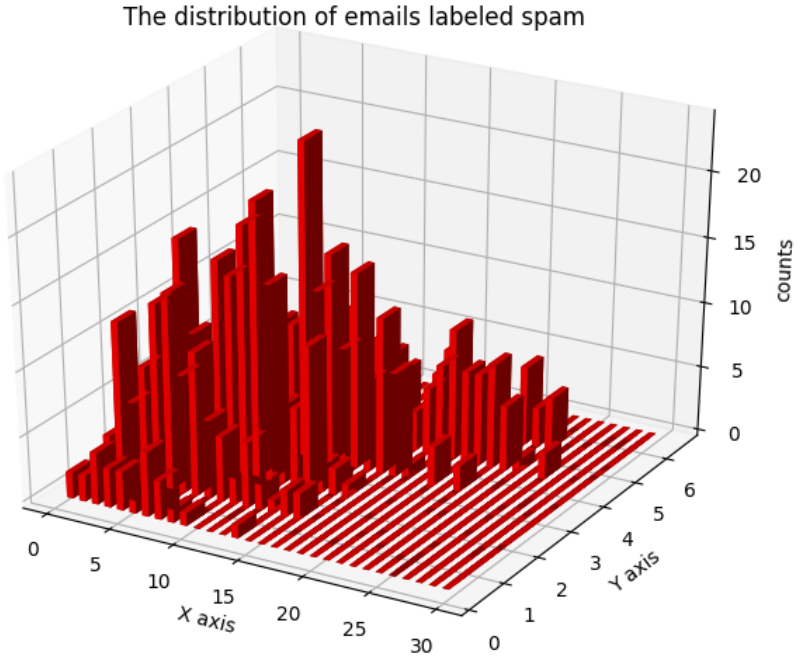
(iii) For emails labeled non-spam, generate a 3D histogram using function `hist3()`.

(iv) Do you notice any significant difference between the two histograms generated in (ii) and (iii)? If so, briefly describe.

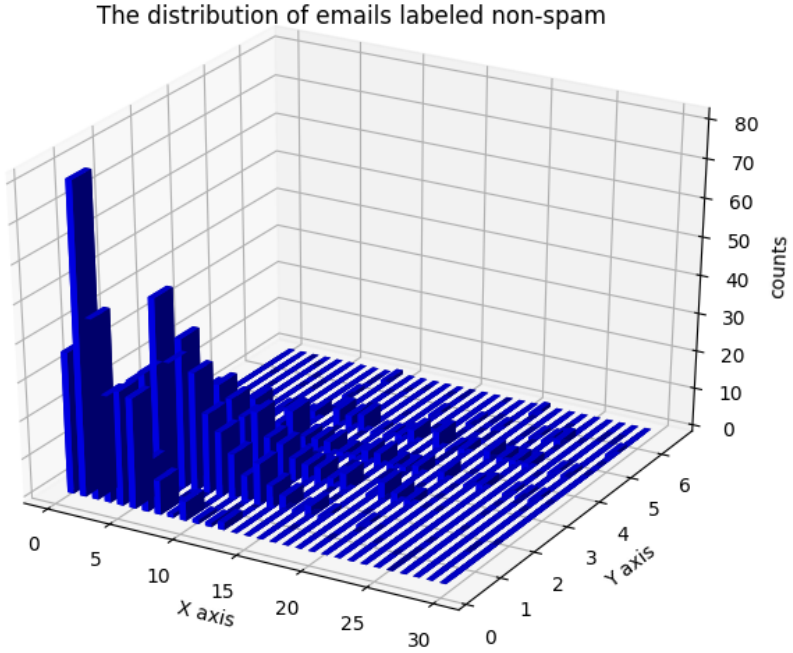
(i)


scatter plot of all testing points
Y:total count of special characters in percentage
X:total count of keywords in percentage
spam mail
non-spam mail

(ii)


The distribution of emails labeled spam
X axis
Y axis
counts

(iii)


The distribution of emails labeled non-spam
X axis
Y axis
counts

(iv)Yes. From the plots above, x axis is the sum of features 1-48 (total count of keywords in percentage) and y axis is the sum of features 49-54 (total count of special characters in percentage). When we compared the red histogram which is the distribution of spam mails with the blue one which is the distribution of non-spam mails, we can find that most of the spam mails contain a lot of keywords and special characters in its contents, however, only a few non-spam mails have a lot of keywords and special characters. Most of the non-spam mails have lower values in both x axis and y axis.

2. Murphy Exercise 8.3.

a. Let $\sigma(a) = \frac{1}{1+e^{-a}}$ be the sigmoid function. Show that

$$\frac{d\sigma(a)}{da} = \sigma(a)(1-\sigma(a)) \tag{8.124}$$

b. Using the previous result and the chain rule of calculus, derive an expression for the gradient of the log likelihood (Equation 8.5).

c. The Hessian can be written as $\mathbf{H} = \mathbf{X}^T\mathbf{S}\mathbf{X}$, where $\mathbf{S} \triangleq \text{diag}(\mu_1(1-\mu_1), \ldots, \mu_n(1-\mu_n))$. Show that $\mathbf{H}$ is positive definite. (You may assume that $0 < \mu_i < 1$, so the elements of $\mathbf{S}$ will be strictly positive, and that $\mathbf{X}$ is full rank.)

For part (b), also answer the question: is Eq. (8.5) the gradient of the log likelihood, or of the negative log likelihood?

For part (c), also answer the question: $\underline{\underline{\mathbf{H}}}$ is positive definite implies what about the negative log likelihood function?

a. $\sigma(a) = \frac{1}{1+e^{-a}} = (1+e^{-a})^{-1}$

$$\frac{\partial \sigma(a)}{\partial a} = -(1+e^{-a})^{-2}\cdot e^{-a}(-1) = \frac{1}{1+e^{-a}} \times \frac{e^{-a}}{1+e^{-a}} = \sigma(a)\cdot(1-\sigma(a))$$

b. $f(w) = -\sum_{n=1}^{N}[y_i \log \mu_i + (1-y_i)\log(1-\mu_i)]$

$a = w^T x$, $\frac{\partial a}{\partial w} = x$, $\mu_i = \frac{1}{1+e^{-a}} = \sigma(a)$

$$g = \frac{\partial f(w)}{\partial w} = \frac{\partial f(w)}{\partial \mu_i}\times\frac{\partial \mu_i(a)}{\partial a}\times\frac{\partial a}{\partial w} = -\sum_{n=1}^{N}\left[y_i\cdot\frac{1}{\mu_i}\cdot\mu_i(a)(1-\mu_i(a))x_i + (1-y)\times\frac{-1}{1-\mu_i}\cdot\mu_i(a)(1-\mu_i(a))\cdot x_i\right]$$

$$= \sum_{n=1}^{N}(y_i x_i \mu_i - x_i y_i + x_i\mu_i - y_i x_i \mu_i)$$

$$= \sum_{n=1}^{N}(\mu_i - y_i)x_i$$

$$= X^T(\mu - y) \quad \text{Q.E.D.}$$

Eq(8.5) is the gradient of the negative log likelihood.

c. $\underline{\underline{H}} = \underline{\underline{X}}^T\underline{\underline{S}}\underline{\underline{X}} > 0 \qquad \underline{\underline{X}} \in \mathbb{R}^{N\times d}$

$\underline{\underline{S}} = \text{diag}(\mu_i(1-\mu_i)) > 0$ ($\because 0 < \mu_i < 1$) and $\underline{\underline{X}}$ is full rank.

Assume $\delta \in \mathbb{R}^d$ and δ is any non-zero vector.

$\delta^T\underline{\underline{S}}\delta > 0$ ($\because \underline{\underline{S}}$ is positive definite)

$\delta^T\underline{\underline{X}}^T\underline{\underline{S}}\underline{\underline{X}}\delta = (\underline{\underline{X}}\delta)^T\underline{\underline{S}}(\underline{\underline{X}}\delta) > 0 \Rightarrow H$ is positive definite

When $\underline{\underline{H}}$ is positive definite, it implies that the negative log likelihood is convex and we can find its global minimum.

3. Suppose our "learning algorithm" uses a standard linear model for $\hat{f}$ in a classification problem, in which there are D input variables (features), and augmented notation is used:

$$\hat{f}(\underline{x}) = \mathrm{sgn}\left(\underline{w}^T \underline{x}\right)$$

in which $\mathrm{sgn}(u) \triangleq 1 \cdot [\![u > 0]\!] - 1 \cdot [\![u < 0]\!]$, and $[\![\ ]\!]$ denotes the indicator function. The learning algorithm picks the best weight vector $\hat{\underline{w}}$ using the training data $\mathcal{D}$, based on minimizing some objective function $J(\underline{w}, \mathcal{D})$, with each component of $\underline{w}$ restricted to:

$$w_0 = 1; \quad w_j \in \{1, 2\} \quad \forall j \in \{1, 2, \cdots, D\}.$$

(a) How many elements (hypotheses) are there in the hypothesis set $\mathcal{H}$?

(b) How would the Hoeffding Inequality be applied to this case? That is, give an expression, if possible, for an upper bound on $P\left[\left|E_{in}(\hat{h}) - E_{out}(\hat{h})\right| > \varepsilon\right]$.

(a) Since there are D features, and each w might be 1 or 2 (2 choices), there are 2^D choices for each output. Therefore, there are 2^D elements in the hypothesis set $\mathcal{H}$.

(b)
$$P[|E_{in}(\hat{h}) - E_{out}(\hat{h})| > \epsilon] \le P[|E_{in}(h_1) - E_{out}(h_1)| > \epsilon \text{ or } |E_{in}(h_2) - E_{out}(h_2)| > \epsilon \text{ or } \cdots\cdots \text{ or } |E_{in}(h_M) - E_{out}(h_M)| > \epsilon]$$
$$\le \sum_{m=1}^{M} P[|E_{in}(h_m) - E_{out}(h_m)| > \epsilon], \quad M = 2^D$$
$$\Rightarrow P[|E_{in}(\hat{h}) - E_{out}(\hat{h})| > \epsilon] \le 2M e^{-2\epsilon^2 N} = 2^{D+1} e^{-2\epsilon^2 N}$$

for any $\epsilon > 0$, N: number of training samples