COMP540: Statistical Machine Learning
Assignment 2 Write-up

Created by
Yunyi Lin (yl146) & Pei Zeng (pz14)
Jan 28th, 2018

Problem1

Part a

Because $g(z) = \frac{1}{1+e^{-z}}$

Therefore,

$$\frac{\partial(g(z))}{\partial(z)} = \frac{0-(-e^{-z})}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}}\left(\frac{1+e^{-z}-1}{1+e^{-z}}\right) = g(z)(1-g(z))$$

Part b

The $NLL(\theta)$ is as showed below:

$$NLL(\theta) = -logP(D \mid \theta) = \sum_{i=1}^{n}[-y^{(i)}logh_\theta(x^{(i)}) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$$

where

$$h_\theta(x^{(i)}) = g(\theta^T x^{(i)}) = \frac{1}{1+e^{-\theta^T x^{(i)}}}$$

Let's set $z = \theta^T x^{(i)}$

Then we can obtain $\frac{\partial(NLL(\theta))}{\partial(\theta)} = \frac{\partial(NLL(\theta))}{\partial(z)}\frac{\partial(z)}{\partial(\theta)}$ by the chain rule of calculus.

Moreover,

$$\frac{\partial(z)}{\partial(\theta)} = \frac{\partial(\theta^T x^{(i)})}{\partial(\theta)} = x^{(i)}$$

While

$$\frac{\partial(NLL(\theta))}{\partial(z)} = \sum_{i=1}^{n}\left(-y^{(i)}\frac{(g(z)')}{g(z)} - (1-y^{(i)})\frac{(1-g(z))'}{1-g(z)}\right)$$

$$= \sum_{i=1}^{n}\left(-y^{(i)}\frac{g(z)(1-g(z))}{g(z)} - (1-y^{(i)})\frac{g(z)(g(z)-1)}{(1-g(z))}\right)$$

$$= \sum_{i=1}^{n}(-y^{(i)} + y^{(i)}g(z) - y^{(i)}g(z) + g(z))$$

$$= \sum_{i=1}^{n} (h_\theta(x^{(i)} - y^{(i)})$$

Therefore

$$\frac{\partial(NLL(\theta))}{\partial(\theta)} = \sum_{i=1}^{n} (h_\theta(x^{(i)} - y^{(i)}) x^{(i)}$$

Hence the proof.

Part c

The Hessian or second derivative of the $NLL(\theta)$ can be written as $H = X^T S X$ where

$S = diag(h_\theta(x^{(1)})(1 - h_\theta(x^{(1)})), \ldots, h_\theta(x^{(m)})(1 - h_\theta(x^{(m)}))$

Moreover, we assume that $0 < h_\theta(x^{(i)})(1 - h_\theta(x^{(i)})) < 1$, so the elements of S are strictly positive and that X is full rank.

We need to show that H is *positive definite*.

By definition, a symmetric n × n real matrix is said to be *positive definite* if the scalar $z^T X$z is positive for every non-zero column vector z of n real numbers.

Therefore,

$z_{1*d}^T H_{d*d} z_{d*1} = z_{1*d}^T (X_{d*m}^T S_{m*m} X_{m*d}) z_{d*1} = (Xz)^T S (Xz)$

$$= \left(\sum_{j=1}^{n} x_j^{(1)} z_j, \ldots, \sum_{j=1}^{n} x_j^{(m)} z_j \right) S \begin{pmatrix} \sum_{j=1}^{n} x_j^{(1)} z_j \\ , \ldots , \\ \sum_{j=1}^{n} x_j^{(m)} z_j \end{pmatrix}$$

$$= (\sum_{j=1}^{n} x_j^{(1)} z_j)^2 S_{11} +, \ldots, +(\sum_{j=1}^{n} x_j^{(m)} z_j)^2 S_{mm}$$

$$= \sum_{i=1}^{n} S_{ii} (\sum_{j=1}^{n} x_j^{(i)} z_j)^2$$

Because $S_{ii} > 0$, so it's obvious to see that scalar $\sum_{i=1}^{n} S_{ii} (\sum_{j=1}^{n} x_j^{(i)} z_j)^2 > 0$

Therefore, we proved H is *positive definite.*

Problem 2

Part a

False

Since $\lambda \geq 0, J(\theta)$ should be a convex function and would just have one global optimal solution.

Part b

False

L2 regularization can derive θ to zero because the tangent point cannot be on the either y/x axis based on the picture showed in lecture.

Part c

True

By definition, two sets are *linearly separable* if there exists at least one line in the plane with two point sets in the two sides of the line. There exists a line x=a, where a is a constant, separating the data into two symmetrical sets, the we can easily have $\theta = \infty$.

Part d

True

Increase λ will always lead to under-fitting (which is poorer fitting) which lead to decrease in θ thus lead to the decrease in $\frac{1}{1+e^{-\theta^T x^{(i)}}}$ and thus the first part of $J(\theta)$ would increase. Therefore, I think it's true that the first term of $J(\theta)$ always increases as we increase λ.

Problem 3

Download the data

Set up training and test sets

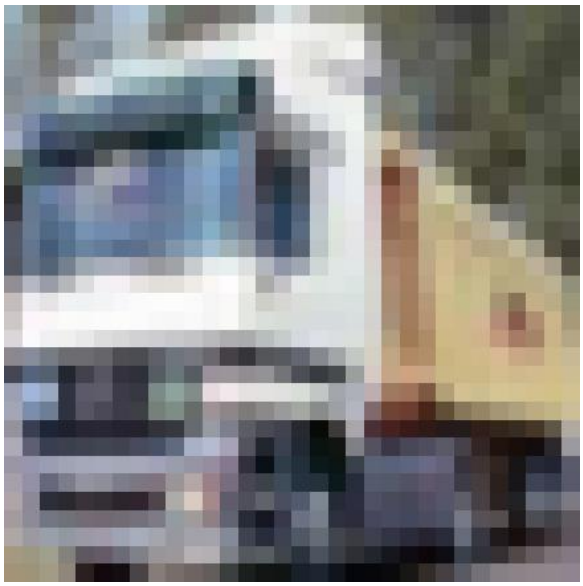
Training data shape:(50000, 32, 32, 3)

Training labels shape:(50000,)

Test data shape:(10000, 32, 32, 3)

Test labels shape:(10000,)

y train[1] 9



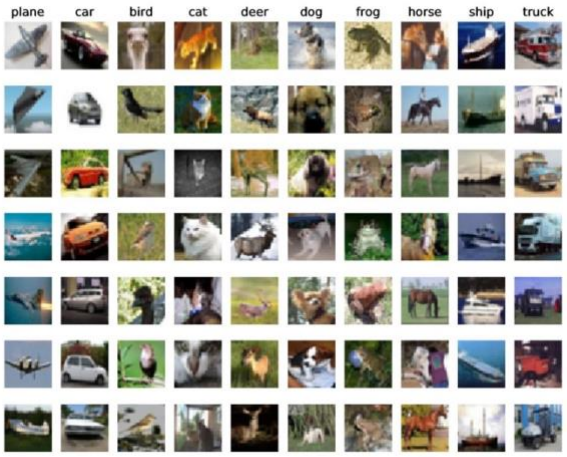


*Fig.*3-1

Flatten each image into a 1-dimensional array of size 3072 (i.e., 32×32×3)

X_train.shape:(5000, 3072)

X_test.shape:(500, 3072)

Create a k-nearest-neighbor classifier

Classify test data with a k-nearest-neighbor classifier

Distance matrix computation with two loops

dists.shape:(500, 5000)

*Fig.*3-2

Q1: What in the data is the cause behind the distinctly bright rows?

Answer1:

The rows with visible brighter color indicate those test samples are quite different from the major training data and easier to be identified.

Q2: What causes the columns?

Answer2:

The columns with visible brighter color indicate those training samples are quite different from the major test data. This could be caused by those training samples only share the similar features within the training data but not test data.

Compute majority label

When k = 1: Got 137 / 500 correct => accuracy: 0.274000

When k = 5: Got 139 / 500 correct => accuracy: 0.278000

Distance matrix computation with one loop

Difference was: 0.000000

Good! The distance matrices are the same

Distance matrix computation with no loops

Difference was: 0.000000

Good! The distance matrices are the same

Speeding up distance computations

Two loop version took 30.463630 seconds

One loop version took 63.875703 seconds

No loop version took 0.241801 seconds

Thus, we confirmed that the no loop version is significantly faster (about x100) than the two-loop version.

Choosing k by cross validation

k = 1, accuracy = 0.263000

k = 1, accuracy = 0.257000

k = 1, accuracy = 0.264000

k = 1, accuracy = 0.278000

k = 1, accuracy = 0.266000

k = 3, accuracy = 0.239000

k = 3, accuracy = 0.249000

k = 3, accuracy = 0.240000

k = 3, accuracy = 0.266000

k = 3, accuracy = 0.254000

k = 5, accuracy = 0.248000

k = 5, accuracy = 0.266000

k = 5, accuracy = 0.280000

k = 5, accuracy = 0.292000

k = 5, accuracy = 0.280000

k = 8, accuracy = 0.262000

k = 8, accuracy = 0.282000

k = 8, accuracy = 0.273000

k = 8, accuracy = 0.290000

k = 8, accuracy = 0.273000

k = 10, accuracy = 0.265000

k = 10, accuracy = 0.296000

k = 10, accuracy = 0.276000

k = 10, accuracy = 0.284000

k = 10, accuracy = 0.280000

k = 12, accuracy = 0.260000

k = 12, accuracy = 0.295000

k = 12, accuracy = 0.279000

k = 12, accuracy = 0.283000

k = 12, accuracy = 0.280000

k = 15, accuracy = 0.252000

k = 15, accuracy = 0.289000

k = 15, accuracy = 0.278000

k = 15, accuracy = 0.282000

k = 15, accuracy = 0.274000

k = 20, accuracy = 0.270000

k = 20, accuracy = 0.279000

k = 20, accuracy = 0.279000

k = 20, accuracy = 0.282000

k = 20, accuracy = 0.285000

k = 50, accuracy = 0.271000

k = 50, accuracy = 0.288000

k = 50, accuracy = 0.278000

k = 50, accuracy = 0.269000

k = 50, accuracy = 0.266000

k = 100, accuracy = 0.256000

k = 100, accuracy = 0.270000

k = 100, accuracy = 0.263000

k = 100, accuracy = 0.256000

k = 100, accuracy = 0.263000

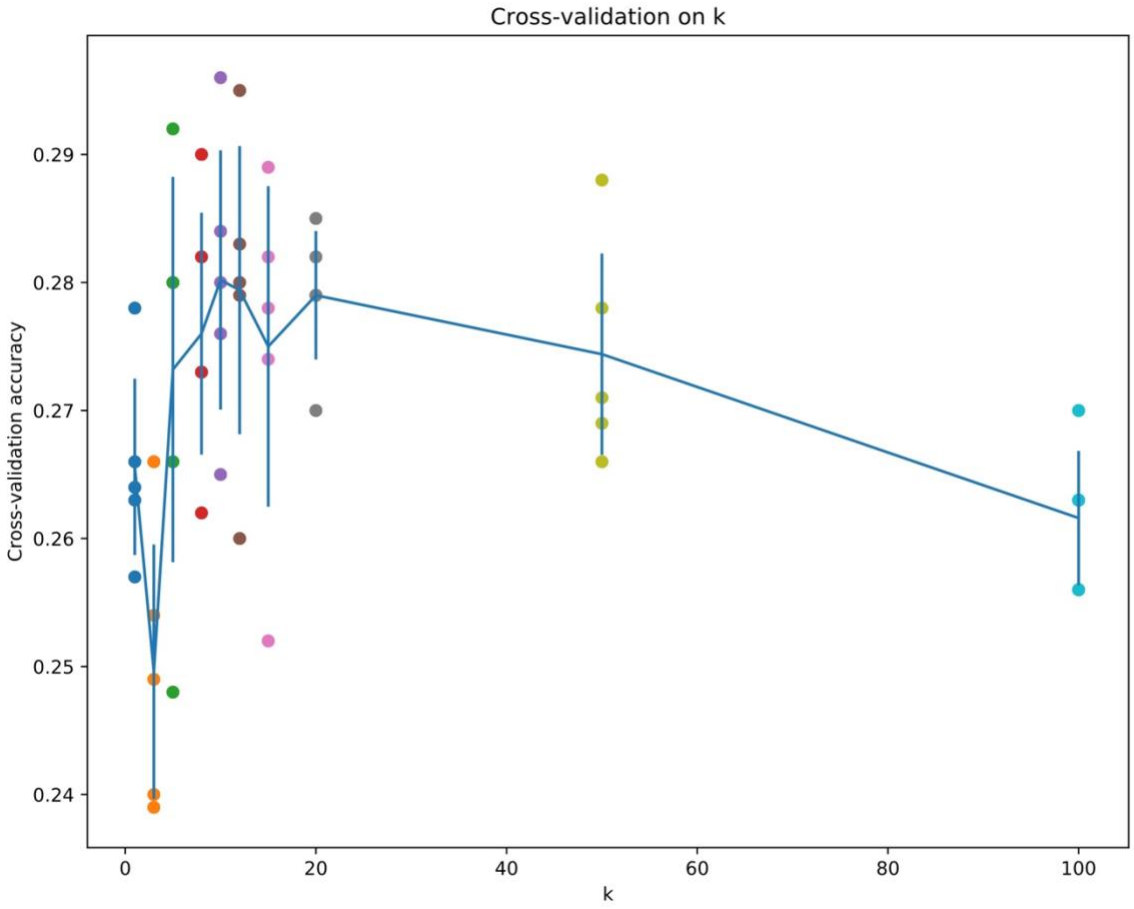


Fig.3-3

Based on the cross validation we picked the best k = 10, and the accuracy on the test data is:

Got 141 / 500 correct => accuracy: 0.282000

Problem 4

Problem 4, Part A: Logistic regression

Visualizing the dataset

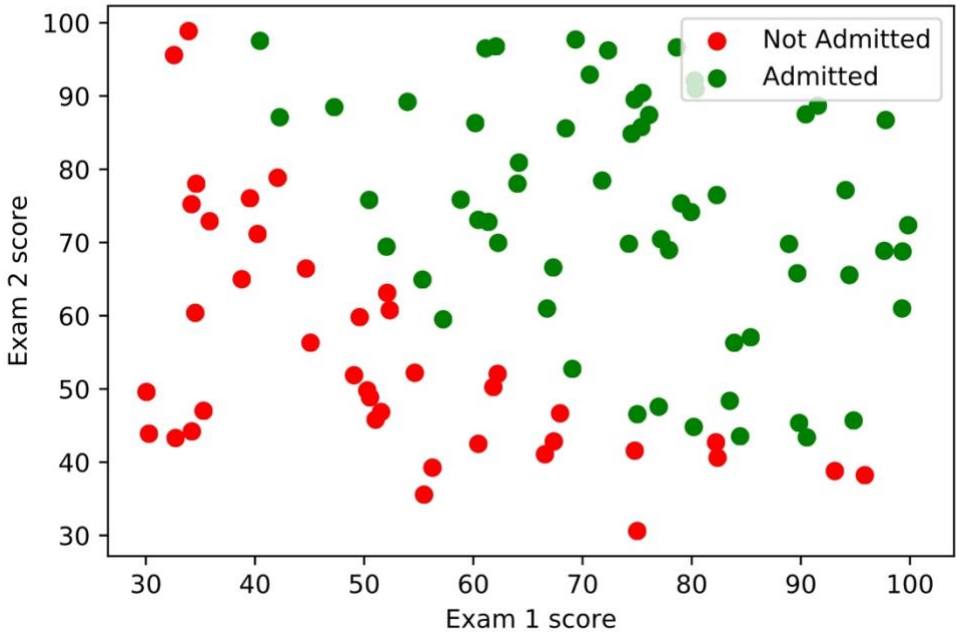


*Fig.*4A-1

Problem 4A1: Implementing logistic regression: the sigmoid function

For a matrix z = [[0, 1000, -1000]] my function calculated the sigmoid function on every element and returned a new matrix as below:

matrix([[0.5, 1. , 0.]])

Problem 4A2: Cost function and gradient of logistic regression

Learning parameters using fmin bfgs

Loss on all-zeros theta vector (should be around 0.693) = 0.69314718056

Gradient of loss wrt all-zeros theta vector (should be around [-0.1, -12.01, -11.26]) = [-0.1 -12.00921659 -11.26284221]

Optimization terminated successfully.

Current function value: 0.203498

Iterations: 19

Function evaluations: 20

Gradient evaluations: 20

Theta found by fmin_bfgs: [-25.16056945 0.20622963 0.20146073]

Final loss = 0.203497702351

Problem 4A3: Prediction using a logistic regression model

For a student with 45 on exam 1 and 85 on exam 2, the probability of admission = 0.776246678481

Accuracy on the training set = 0.89

Visualizing the decision boundary

Plot the decision surface

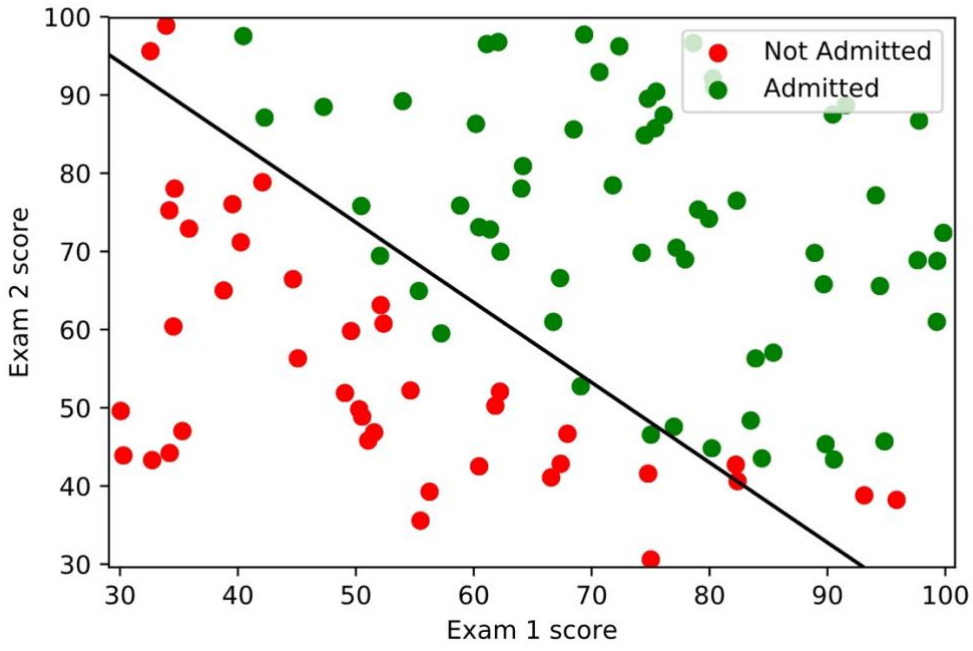


*Fig.*4A-2

Compare with sklearn logistic regression

Theta found by sklearn: [[-25.15293066 0.20616459 0.2014034 9]]

Plot the decision surface of sklearn logistic regression

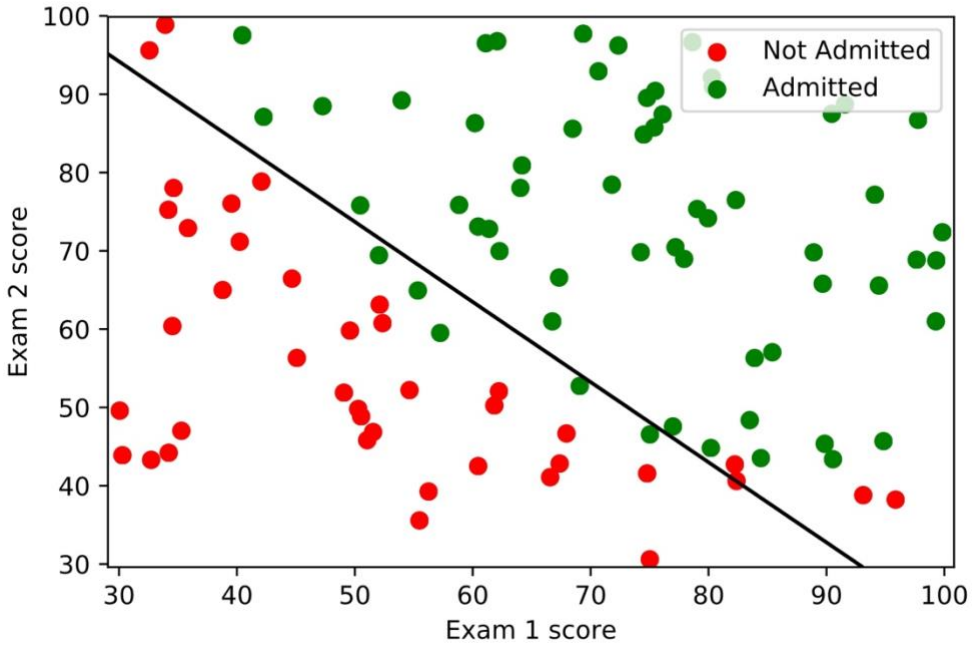


Fig.4A-3

Problem 4, Part B: Regularized logistic regression

Visualizing the data

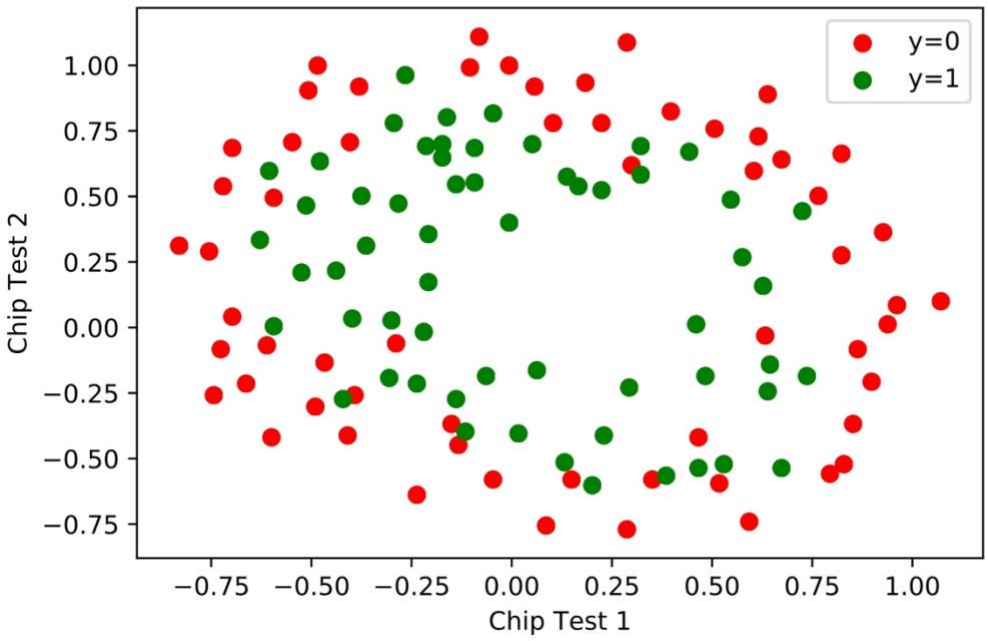


*Fig.*4B-1

Problem 4B1: Cost function and gradient for regularized logistic regression

Optimization terminated successfully.

Current function value: 0.529003

Iterations: 47

Function evaluations: 48

Gradient evaluations: 48

Theta found by fmin_bfgs:[1.27268739 0.62557016 1.1809665 -2.01919822 -0.91761468 -1.43194199

0.12375921 -0.36513086 -0.35703388 -0.17485805 -1.45843772 -0.05129676

-0.61603963 -0.2746414 -1.19282569 -0.24270336 -0.20570022 -0.04499768

-0.27782709 -0.29525851 -0.45613294 -1.04377851 0.02762813 -0.29265642

0.01543393 -0.32759318 -0.14389199 -0.92460119]

Final loss = 0.4624583499

Plotting the decision boundary

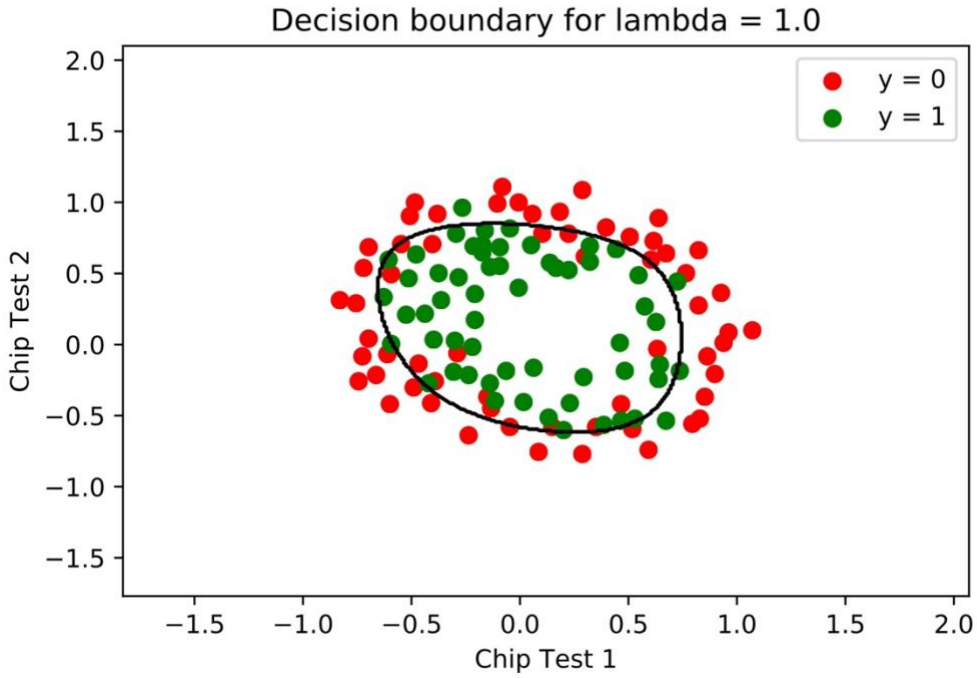


*Fig.*4B-2

Comparing learned model with sklearn's logistic ridge regression

Theta found by sklearn with L2 reg: [1.1421394 0.60141117 1.16712554 -1.87160974 -0.91574144 -1.26966693
0.12658629 -0.3686536 -0.34511687 -0.17368655 -1.42387465 -0.04870064
-0.60646669 -0.26935562 -1.16303832 -0.24327026 -0.20702143 -0.04326335
-0.28028058 -0.286921 -0.46908732 -1.03633961 0.02914775 -0.29263743
0.01728096 -0.32898422 -0.13801971 -0.93196832]

Loss with sklearn theta: 0.46843403006

Plotting the decision boundary

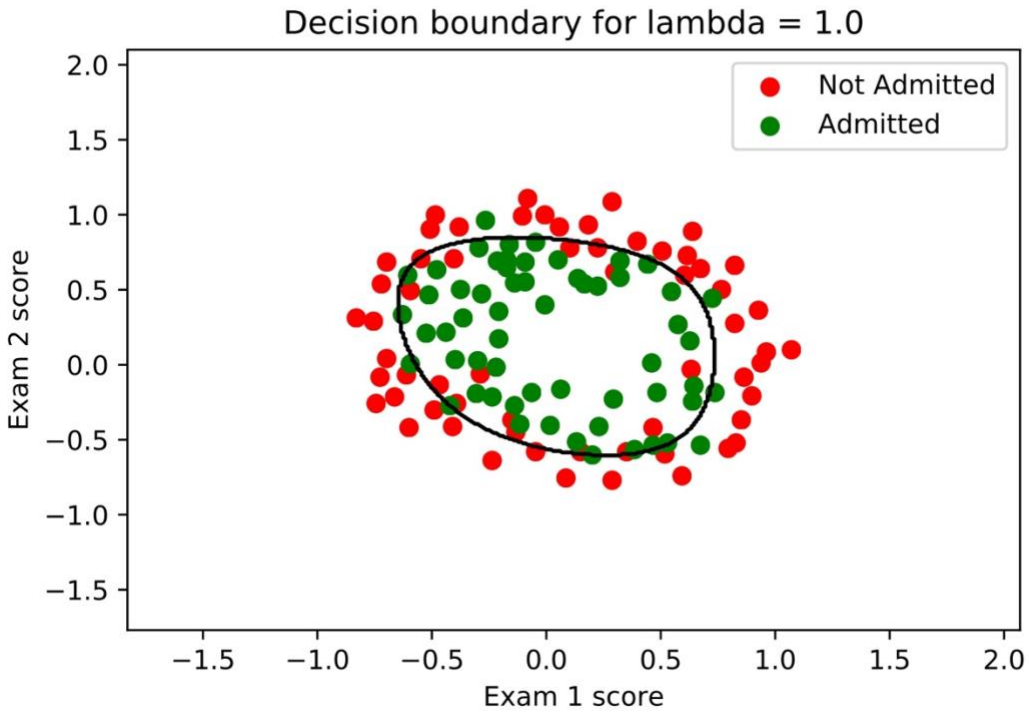


*Fig.*4B-3

Problem 4B2: Prediction using the model

Accuracy on the training set = 0.830508474576

Problem 4B3: Varying λ

When λ =100, the model is under-fitting; When λ = 0, the model is over-fitting.

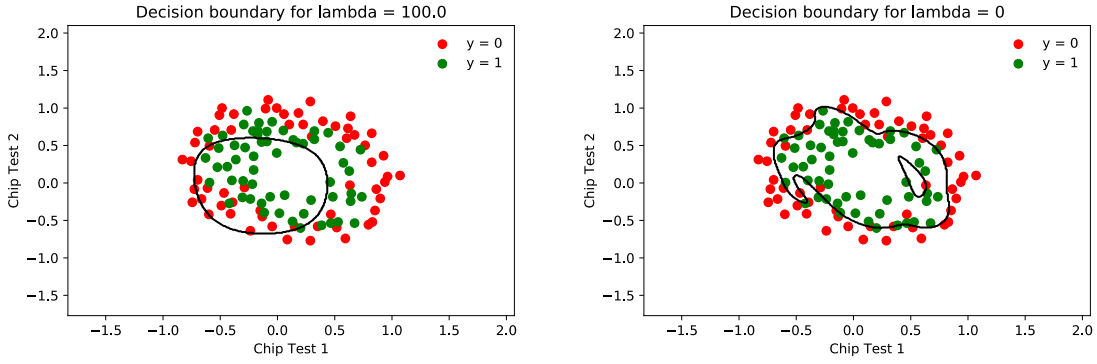


*Fig.*4B-4

L1 regularization witk sklearn LogisticRegression

Theta found by sklearn with L1 reg: [1.86963812 0.68660125 1.280459 -4.86252897 -1.62177647 -2.34227698

0. 0. 0. 0. 0. 0. 0.

0. -2.3674914 0. 0. 0. 0. 0.

0. 0. 0. 0. 0. 0. 0.

0.]

Loss with sklearn theta: 0.438146938908

Plot regularization paths for L1 regression

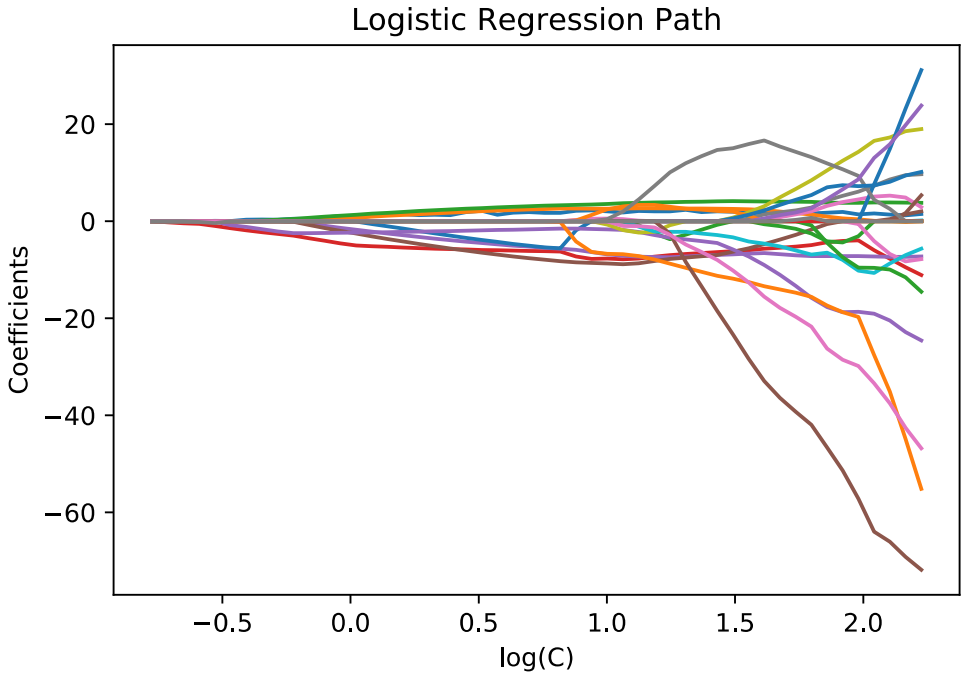


*Fig.*4B-5

Problem 4B4: Exploring L1 and L2 penalized logistic regression

L1 penalization

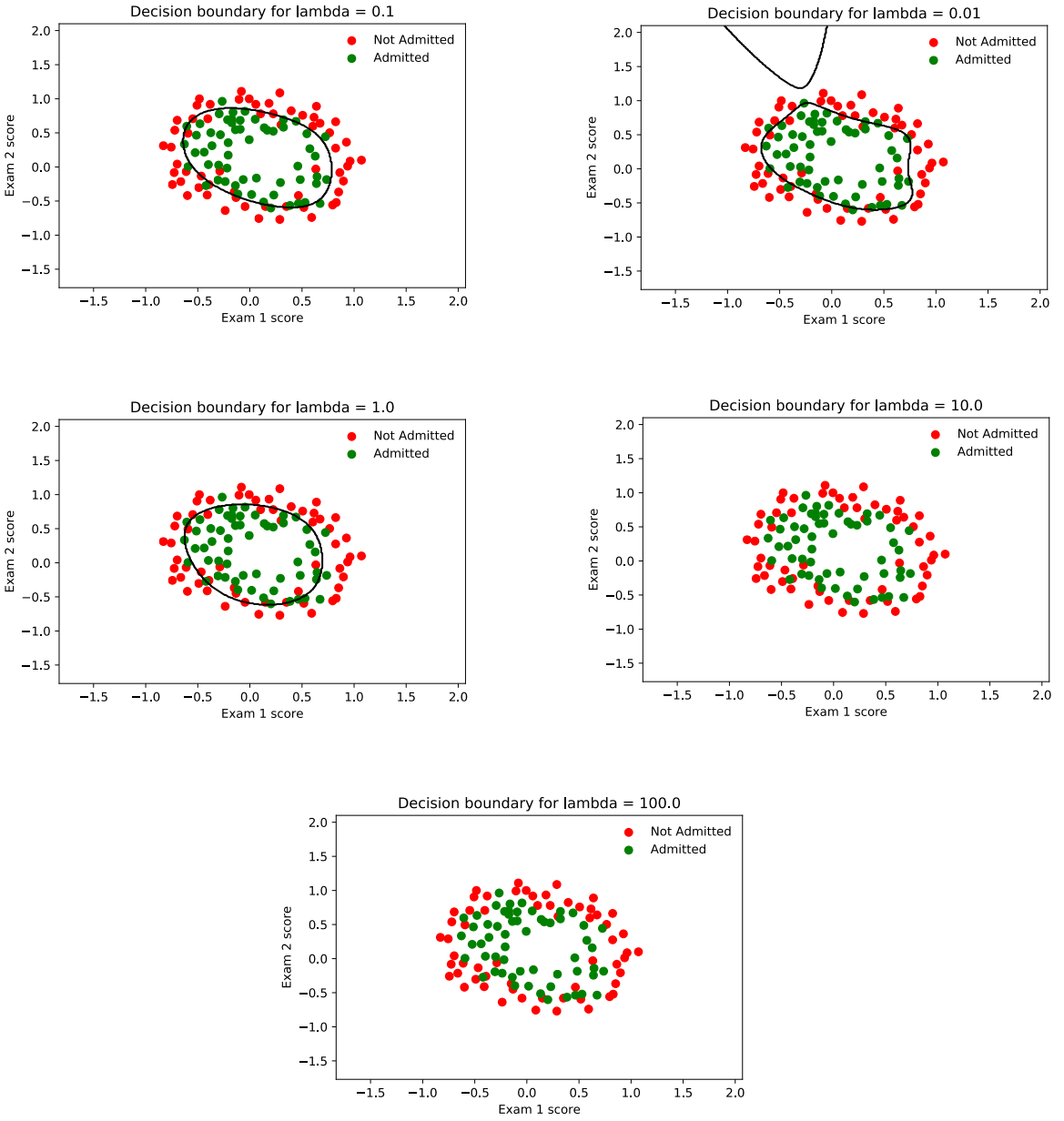


*Fig.*4B-6

L2 penalization

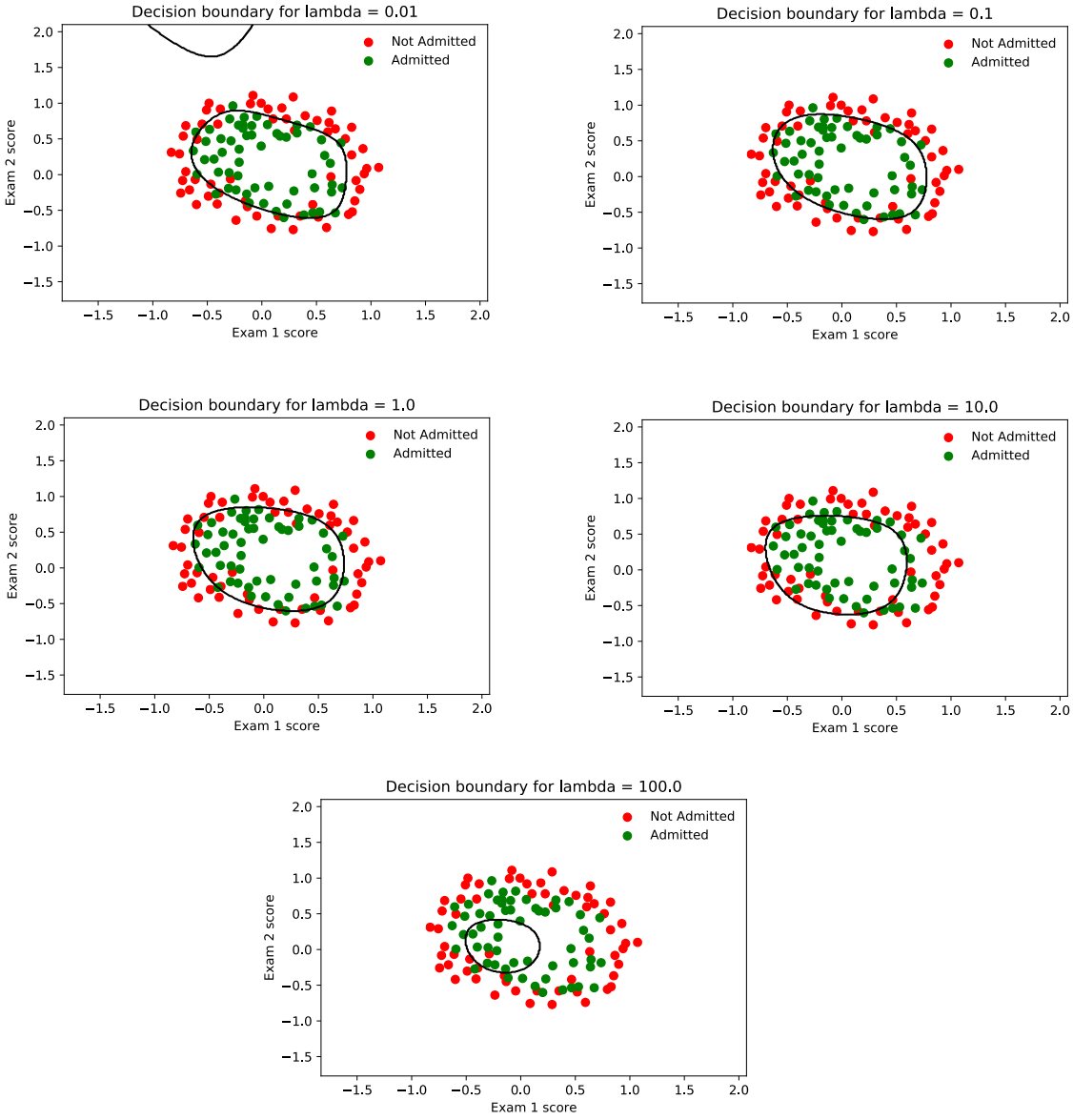


*Fig.*4B-7

Problem 4 Part C: Logistic regression for spam classification

Feature transformation

Fitting regularized logistic regression models (L2 and L1) to classify email

L2 Penalty experiments ----------

best_lambda = 0.1

Accuracy on set aside test set for std = 0.9296875

best_lambda = 0.6

Accuracy on set aside test set for logt = 0.943359375

best_lambda = 1.1

Accuracy on set aside test set for bin = 0.927734375

L1 Penalty experiments ----------

best_lambda = 4.6

Accuracy on set aside test set for std = 0.921875

best_lambda = 1.6

Accuracy on set aside test set for logt = 0.944010416667

best_lambda = 3.6

Accuracy on set aside test set for bin = 0.92578125

Q: Which class of models will you recommend for this data set and why?

Answer:

I will recommend L1 regularization with log transform features.

Although the accuracy of L1 and L2 regularization are very close to each other, L1 regularization will have more sparse coefficients after regularization. Which would simplify the model and focus on the important features instead.