CPSC-540 Machine Learning 2011

Homework # 3

NAME:_____

Signature:_____

STD. NUM: _____

General guidelines for homeworks:

You are encouraged to discuss the problems with others in the class, but all write-ups are to be done on your own.
Homework grades will be based not only on getting the "correct answer," but also on good writing style and clear presentation of your solution. It is your responsibility to make sure that the graders can easily follow your line of reasoning.
Try every problem. Even if you can't solve the problem, you will receive partial credit for explaining why you got stuck on a promising line of attack. More importantly, you will get valuable feedback that will help you learn the material.
Please acknowledge the people with whom you discussed the problems and what sources you used to help you solve the problem (e.g. books from the library, websites, etc.). This won't affect your grade but is important as academic honesty.
When dealing with python exercises, please attach a printout with all your code and show your results clearly.

1. **Regularized linear regression**

(i) Visualizing the data. Download the prostate cancer dataset from the course website.

In this prostate cancer study 9 features—including age, log weight, log cancer volume, etc.—were measured for 97 patients. We will begin by visualizing this data. To do so you must first load the data, using

```
X = np.loadtxt('prostate.data', skiprows=1)
```

and then create a "scatterplot matrix", i.e. a set of subplots which plots each variable against every other variable, as is done in the slides (or see Hastie et al., page 3). *Hand in this plot, along with the code used to generate it.*

(ii) Ridge regression. We will now construct a model using ridge regression to predict the 9th variable as a linear combination of the other 8.

(a) First, split the data into a response variable and predictor variables respectively:

```
y = X[:,-1]
X = X[:,0:-1]
```

and then zero-mean both sets of variables and standardize the predictors to have unit variance.

```
y -= mean(y)
X -= mean(X, axis=0)
X /= std(X, axis=0)
```

(b) Choose the first 67 patients as the training data. The remaining patients will be the test data.

```
ytrain, ytest = y[0:67], y[67:]
Xtrain, Xtest = X[0:67], X[67:]
```

(c) Write code for ridge regression starting from the following skeleton:

```
def ridge(X, y, d2):
    ???
    return theta
```

Compute the ridge regression solutions for a range of regularizers (δ^2). Plot the values of each $\boldsymbol{\theta}$ in the y-axis against δ^2 in the x-axis. This set of plotted values is known as a regularization path. *Hand in this plot, along with the code used to generate it.*

Note: this should look similar to the plot given in the slides (or see Hastie et al., page 65). The plot shown in the book, however, is plotted against the following quantity:

$$df(\delta^2) = \sum_i \frac{\sigma_i^2}{\sigma_i^2 + \delta^2}$$

on the x-axis, where σ_i are the singular values of X.

(d) For each computed value of $\boldsymbol{\theta}$, compute the train and test error. Choose a value of δ^2 that gives you the lowest minmax crossvalidation error. What is this value? Plot the train and test errors as a function of δ^2. *Hand in this plot, along with the code used to generate it.*

(e) For the best $\boldsymbol{\theta}$, plot separately (using subplots) the train and test error as a function of the patient number. That is, for each patient show the actual response and the prediction. *Hand in this plot, along with the code used to generate it.*

(iii) Lasso. We will now implement the Lasso and try this code out on the prostate cancer data.

(a) Implement the coordinate descent (aka "shooting" method) for solving Lasso. Pseudo-code for this solver is given in the slides. You should start with the following skeleton code:

```
def lasso(X, y, d2):
    theta_ = np.zeros(k)
    theta = ridge(X, y, d2)
    while np.sum(np.abs(theta-theta_)) > 1e-5:
        theta_ = theta.copy()
        ???
    return theta
```

(b) Find the solutions and generate the plots from (c–e) of the previous question, but now using this new Lasso solver.

Note: as with the previous question your regularization path should look similar to the plot shown in the slides (or Hastie et al., page 70). The plot shown in the book/slides, however, uses the quantity $1/\sum_j |\theta_j|$ on the x-axis.

2. **Google trends**

In this question we will try and apply some of these techniques to data available via google trends. To start, first download the following code:
`https://github.com/suryasev/unofficial-google-trends-api`

We can grab data from google trends using the following code:

```
import pyGTrends as t
import numpy as np

words = ['flu', 'bread', 'apple', 'bananas', 'walnuts']

# connect using your username and password and get the words.
connector = t.pyGTrends('username','password')
connector.download_report(words)

# construct a numpy array of the word occurences.
lines = connector.csv().split('\n')
X = np.loadtxt(lines, delimiter=',', skiprows=1,
               usecols=range(1, 2*len(words), 2))
```

(Note: when you turn in your code you'll probably want to remove the password!)

The code uses your username/password to sign into google and download a CSV file containing weekly search traffic data amount of searches for that term and their standard deviations. We then construct a matrix `X` such that `X[i,j]` is the weekly traffic for term j during week i.

(a) Construct a new list `words` consisting of flu and 4 other terms (how should you select these terms?). After running the given code we can then split this into output and predictor variables:

```
y = X[:,0]
X = X[:,1:]
```

Standardize this data and split it into test and training sets just as in question (ii).

(b) Use ridge or Lasso regression to predict the frequency of "flu" given the frequencies of the 4 other terms. For the best $\boldsymbol{\theta}$ chosen via cross-validation (i.e. when varying δ^2) plot y and the predicted values $\hat{y}$ for both test and training data. Print the coefficients for each word. Do these coefficients

tell you which terms are more predictive of the term "flu"? How could this strategy be used for *query expansion* in a search engine?

3. **Stock data**

   You will now use Google trends to predict the volume of stock in Apple (AAPL) traded during this same interval. Come up with 5 words you think will be correlated with Apple (e.g. "apple", "ipad", ...) and generate the matrix `X` as we did in the previous question. *Standardize* this matrix as we did before.

   Now, rather than using one of these terms as y, download stock data from the following source:
   `http://www.google.com/finance/historical?cid=22144&startdate=Jan+4%2C+2004&histperiod=weekly&num=30&output=csv`

   If you save this file as `aapl.csv` we can load the data using the following code:

```
y = np.loadtxt('aapl.csv', delimiter=',', skiprows=1, usecols=(5,))
y = y[::-1]
y /= max(y)
```

   This just loads the volume data from 5th column of the file and reverses the vector so that *older* dates come first.

   Use ridge or lasso regression to predict the stock volume. For the best $\boldsymbol{\theta}$ chosen via cross-validation (i.e. when varying δ^2) plot y and the predicted values $\hat{y}$ for both test and training data. Print the coefficients for each word. Does your method do better at predicting stock volumes than a random choice of words as input?

4. **Student T as infinite mixture of Gaussians:** It turns out that several distributions of interest can be expressed as an "infinite" weighted sum of Gaussians, where each Gaussian has a different variance. That is,

$$p(x) = \int \mathcal{N}(x|\mu, \tau^2)\pi(\tau^2)d\tau^2$$

   for some distribution $\pi(\tau^2)$. This is called a *Gaussian scale mixture*.

   One important case is the Student T distribution. This distribution plays a big role in robust statistics. It has the following pdf:

$$\mathcal{T}(x|\mu, \sigma^2, \nu) \triangleq \frac{\Gamma(\frac{\nu}{2}+\frac{1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{(}\sigma^2\nu\pi)} \times \left[1 + \frac{1}{\nu}\left(\frac{x-\mu}{\sigma}\right)^2\right]^{-(\frac{\nu+1}{2})}$$

   where μ is the mean, $\sigma^2 > 0$ is the scale parameter, and $\nu > 0$ is called the *degrees of freedom*. This distribution has the following properties:

$$\text{mean} = \mu, \text{mode} = \mu, \text{ var} = \frac{\nu\sigma^2}{(\nu-2)}$$

   The variance is only defined if $\nu \geq 2$. The mean is only defined if $\nu > 1$.

   Show that the Student T distribution can be written as follows:

$$\mathcal{T}(x|\mu, \sigma^2, \nu) = \int_0^\infty \mathcal{N}(x|\mu, \sigma^2/\lambda)\text{Ga}\left(\lambda|\frac{\nu}{2}, \frac{\nu}{2}\right)d\lambda,$$

   where $\text{Ga}(\lambda|\frac{\nu}{2}, \frac{\nu}{2})$ denotes the Gamma distribution. Explain why the Student T can be used to deal with outliers in regression problems.

5. **Normalization constant for a 1D Gaussian:** The normalization constant for a zero-mean Gaussian is given by

$$Z = \int_a^b \exp\left(-\frac{x^2}{2\sigma^2}\right) dx$$

where $a = -\infty$ and $b = \infty$. To compute this, consider its square

$$Z^2 = \int_a^b \int_a^b \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) dxdy$$

Let us change variables from cartesian (x, y) to polar (r, θ) using $x = r\cos\theta$ and $y = r\sin\theta$. Since $dxdy = rdrd\theta$, and $cos^2\theta + \sin^2\theta = 1$, we have

$$Z^2 = \int_0^{2\pi} \int_0^{\infty} r\exp\left(-\frac{r^2}{2\sigma^2}\right) drd\theta$$

Evaluate this integral and hence show $Z = \sigma\sqrt{(}2\pi)$.

Hint 1: separate the integral into a product of two terms, the first of which (involving $d\theta$) is constant, so is easy.

Hint 2: if $u = e^{-r^2/2\sigma^2}$ then $du/dr = -\frac{1}{\sigma^2} re^{-r^2/2\sigma^2}$, so the second integral is also easy (since $\int u'(r)dr = u(r)$).

6. **Entropy of a Bernoulli variable:**

The Bernoulli for $x \in \{0, 1\}$ can be written as in class or in *exponential family form* as follows:

$$\text{Ber}(x|\mu) = \mu^x(1-\mu)^{1-x} = \exp[x\log(\mu) + (1-x)\log(1-\mu)] = \exp[\boldsymbol{\phi}(x)^T\boldsymbol{\theta}] \quad (1)$$

where $\boldsymbol{\phi}(x) = [\mathbb{I}(x=0), \mathbb{I}(x=1)]$ and $\boldsymbol{\theta} = [\log(\mu), \log(1-\mu)]$. However, this representation is *over-complete* since there is a linear dependendence between the features:

$$\mathbf{1}^T\boldsymbol{\phi}(x) = \mathbb{I}(x=0) + \mathbb{I}(x=1) = 1 \quad (2)$$

Consequently $\boldsymbol{\theta}$ is not uniquely *identifiable*. It is common to require that the representation be *minimal*, which means there is a unique $\boldsymbol{\theta}$ associated with the distribution. In this case, we can just define

$$\text{Ber}(x|\mu) = (1-\mu)\exp\left[x\log\left(\frac{\mu}{1-\mu}\right)\right]$$

Now we have $\phi(x) = x$, $\theta = \log\left(\frac{\mu}{1-\mu}\right)$, which is the *log-odds ratio* or *logit* function, and $Z = 1/(1-\mu)$.

(a) Show that we can recover the mean parameter μ from the canonical parameter using

$$\mu = \text{sigm}(\theta) = \frac{1}{1+e^{-\theta}}$$

(b) Derive an expression for the entropy $H(\mu)$ of a Bernoulli variable. Plot this expression and explain the plot briefly.

(c) Prove that the derivative of the binary entropy can be expressed as the negative of the logit function. That is,

$$dH(\mu)/d\mu = -logit(\mu) = -\log\left(\frac{\mu}{1-\mu}\right).$$