CPSC 340 Assignment 4 (due November 17 ATE)

1 Multi-Class Logistic

The function *example_multiClass* loads a multi-class classification datasetwith $y_i \in \{1, 2, 3, 4, 5\}$ and fits a 'one-vs-all' classification model using binary logistic regression, then reports the validation error and shows a plot of the data/classifier. The performance on the validation set is ok, but could be much better. For example, this classifier never even predicts that examples will be in class 1 (the green class).

1.1 Softmax Classification

Linear classifiers make their decisions by finding the class label c maximizing the quantity $w_c^T x_i$, so we want to train the model to make $w_{y_i}^T x_i$ larger than $w_{c'}^T x_i$ for all the classes c' that are not y_i. Here c is a possible label and $w_{c'}$ is column c' of W. Similarly, y_i is the training label, w_{y_i} is column y_i of W, and in this setting we are assuming a discrete label $y_i \in \{1, 2, \ldots, k\}$. Before we move on to implementing the softmax classifier to fix the issues raised in the introduction, let's do a simple example:

Consider the dataset below, which has 10 training examples, 2 features, and 3 class labels:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}.$$

Suppose that you want to classify the following test example:

$$\hat{x} = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

Suppose we fit a multi-class linear classifier using the softmax loss, and we obtain the following weight matrix:

$$W = \begin{bmatrix} +2 & +2 & +3 \\ -1 & +2 & -1 \end{bmatrix}$$

1. Why are the weights of this model a 2×3 matrix?
2. Under this model, what class label would we assign to the test example? (Show your work.)

1.2 Softmax Loss

Using a one-vs-all classifier hurts performance because the classifiers are fit independently, so there is no attempt to calibrate the columns of the matrix W. An alternative to this independent model is to use the softmax loss probability,

$$p(y_i|W, x_i) = \frac{\exp(w_{y_i}^T x_i)}{\sum_{c=1}^{k} \exp(w_c^T x_i)}.$$

The loss function corresponding to the negative logarithm of the softmax probability for n training examples is given by

$$f(W) = \sum_{i=1}^{n} \left[-w_{y_i}^T x_i + \log \left(\sum_{c'=1}^{k} \exp(w_{c'}^T x_i) \right) \right].$$

Derive the partial derivative of this loss function with respect to a particular element W_{jc}. Try to simplify the derivative as much as possible (but you can express the result in summation notation).

Hint: for the gradient you can use x_{ij} to refer to element j of example i. For the first term you can use an 'indicator' function, $I(y_i = c)$, which is 1 when $y_i = c$ and is 0 otherwise. Note that you can use the definition of the softmax probability to simplify the derivative.

1.3 Softmax Classifier

Make a new function, *softmaxClassifier*, which fits W using the softmax loss from the previous section instead of fitting k independent classifiers. Hand in the code and report the validation error.

Hint: you will want to use the *derivativeCheck* option in *findMin.jl* to check that your gradient code is correct. Also, note that *findMin.jl* expects that the parameter vector and gradient are *column vectors*. The easiest way to work around these issues is to use the *reshape* command: call *findMin.jl* with a $dk \times 1$ vector w and at the start of your objective function reshape w to be a $d \times k$ matrix W, then compute the $d \times k$ matrix of partial derivatives and finally reshape this to be a $dk \times 1$ vector.

1.4 Cost of Multinomial Logistic Regression

Assuming that we have

- n training examples.
- d features.
- k classes.
- t testing examples.
- T iterations of gradient descent for training.

1. In $O()$ notation, what is the cost of training the softmax classifier?
2. What is the cost of classifying the test examples?

2 MAP Estimation

In class, we considered MAP estimation in a regression model where we assumed that:

- The likelihood $p(y_i|x_i, w)$ is a normal distribution with a mean of $w^T x_i$ and a variance of 1.
- The prior for each variable j, $p(w_j)$, is a normal distribution with a mean of zero and a variance of λ^{-1}.

Under these assumptions, we showed computing the MAP estimate with n training examples leads to the standard L2-regularized least squares objective function:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2.$$

For each of the alternate assumptions below, write down the objective function that results (from minimizing the negative log-posterior, and simplifying as much as possible):

1. We use a Laplace likelihood with a mean of $w^T x_i$ and a scale of 1, and a normal prior with a variance of λ^{-1} and a mean that is some "guess" w^0 of the optimal parameter vector,

$$p(y_i|x_i, w) = \frac{1}{2}\exp(-|w^T x_i - y_i|), \quad p(w_j) \propto \exp\left(-\frac{\lambda(w_j - w_j^0)^2}{2}\right)$$

2. We use a normal likelihood with a mean of $w^T x_i$ but where each example i has its own positive variance σ_i^2, and we use a zero-mean Laplace prior for each variable with a scale parameter of λ^{-1},

$$p(y_i|x_i, w) = \frac{1}{\sqrt{2\sigma_i^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right), \quad p(w_j) = \frac{\lambda}{2}\exp(-\lambda|w_j|).$$

   The standard notation for this case is to use Σ as a diagonal matrix with the σ_i^2 values along the diagonal.

3. We use a Poisson likelihood with a mean of $\exp(w^T x_i)$,[1] and a zero-mean normal prior with a variance of σ^2,

$$p(y_i|x_i, w) = \frac{\exp(y_i w^T x_i)\exp(-\exp(w^T x_i))}{y_i!}, \quad p(w_j) = \frac{1}{\sqrt{2\sigma^2\pi}}\exp\left(-\frac{(w_j - 0)^2}{2\sigma^2}\right).$$

   For this sub-question you don't need to put likelihood in matrix notation.

3 Principal Component Analysis (2016)

3.1 PCA by Hand

Consider the following dataset, containing 5 examples with 2 features each:

x_1	x_2
-2	-1
-1	0
0	1
1	2
2	3

Recall that with PCA we usually assume that the PCs are normalized ($\|w\| = 1$), we need to center the data before we apply PCA, and that the direction of the first PC is the one that minimizes the orthogonal distance to all data points.

[1] This is one way to use regression to model *counts*, like "number of Facebook likes".

1. What is the first principal component?
2. What is the (L2-norm) reconstruction error of the point (3,3)? (Show your work.)
3. What is the (L2-norm) reconstruction error of the point (3,4)? (Show your work.)

3.2 Data Visualization

The script *example_PCA* will load a dataset containing 50 examples, and measuring 85 binary traits of these animals. It then standardizes these features and gives two unsatisfying visualizations of it. First it shows a plot of the matrix entries, which has too much information and thus gives little insight into the relationships between the animals. Next it shows a scatterplot based on two random features and displays the name of 10 randomly-chosen animals. Because of the binary features even a scatterplot matrix shows us almost nothing about the data.

The function *PCA*, which applies the classic PCA method (orthogonal bases via SVD) for a given k. Using this function, modify the demo so that the scatterplot uses the latent features z_i from the PCA model. Make a scatterplot of the two columns in Z, and use the *annotate* function to label a bunch of the points in the scatterplot.

1. Hand in your modified demo and the scatterplot.
2. Which trait of the animals has the largest influence (absolute value) on the first principal component? (Make sure not to forget the "+1" when looking for the name of the trait in the *dataTable*).
3. Which trait of the animals has the largest influence (absolute value) on the second principal component?

3.3 Data Compression

It is important to know how much of the information in our dataset is captured by the low-dimensional PCA representation. In class we discussed the "analysis" view that PCA maximizes the variance that is explained by the PCs, and the connection between the Frobenius norm and the variance of a centered data matrix X. Use this connection to answer the following:

1. How much of the variance is explained by our two-dimensional representation from the previous question?
2. How many PCs are required to explain 50% of the variance in the data?
3. How many PCs are required to explain 75% of the variance in the data?

Note: you can compute the Frobenius norm of a matrix using the function *vecnorm*.

4 Very-Short Answer Questions

1. What is the key advantage of stochastic gradient methods over gradient descent methods?
2. Does stochastic gradient descent with a fixed step-size converge to the minimum of a convex function in general?
3. What is the difference between multi-label and multi-class classification?
4. What is the difference between MLE and MAP?

5. Linear regression with one feature and PCA with 2 features (and $k = 1$) both find a line in a two-dimensional space. Do they find the same line? Briefly justify your answer.
6. Are the vectors minimizing the PCA objective unique? Briefly justify your answer
7. Name two methods for promoting sparse solutions in a linear regression model that we discussed in class that result in convex problems.
8. Can we always use the normal equations to solve non-negative least squares problems?

We're looking for short and concise 1-sentence answers, not long and complicated answers. Also, there is roughly 1-2 questions per lecture.