10-701 Machine Learning: Assignment 1

Instructions: Failure to follow these directions may result in loss of points.

- Your solutions for this assignment need to be in a pdf format and should be submitted to the blackboard and a webpage (to be specified later) for peer-reviewing.
- For the programming question, your code should be well-documented, so a TA can understand what is happening.
- DO NOT include any identification (your name, andrew id, or email) in both the content and filename of your submission.

MLE, MAP, Concentration

1. MLE of Uniform Distributions [5 pts]

Given a set of i.i.d samples $X_1, ..., X_n \quad Uniform(0, \theta)$, find the maximum likelihood estimator of θ.

(a) Write down the likelihood function (3 pts)

(b) Find the maximum likelihood estimator (2 pts)

2. Concentration [5 pts]

The instructors would like to know what percentage of the students like the Introduction to Machine Learning course. Let this unknown—but hopefully very close to 1—quantity be denoted by μ. To estimate μ, the instructors created an anonymous survey which contains this question:

”Do you like the Intro to ML course? Yes or No”

Each student can only answer this question once, and we assume that the distribution of the answers is i.i.d.

(a) What is the MLE estimation of μ? (1 pts)

(b) Let the above estimator be denoted by $\hat{\mu}$. How many students should the instructors ask if they want the estimated value $\hat{\mu}$ to be so close to the unknown μ such that

$$\mathbb{P}(|\hat{\mu} - \mu| > 0.1) < 0.05, \qquad \text{(4pts)}$$

3. MAP of Multinational Distribution [10 pts]

You have just got a loaded 6-sided dice from your statistician friend. Unfortunately, he does not remember its exact probability distribution $p_1, p_2, ..., p_6$. He remembers, however, that he generated the vector $(p_1, p_2, \ldots, p_6)$ from the following Dirichlet distribution.

$$\mathbb{P}(p_1, p_2, \ldots, p_6) = \frac{\Gamma(\sum_{i=1}^6 u_i)}{\prod_{i=1}^6 \Gamma(u_i)} \prod_{i=1}^6 p_i^{u_i - 1} \delta(\sum_{i=1}^6 p_i - 1),$$

where he chose $u_i = i$ for all $i = 1, \ldots, 6$. Here Γ denotes the gamma function, and δ is the Dirac delta. To estimate the probabilities $p_1, p_2, \ldots, p_6$, you roll the dice 1000 times and then observe that side i occurred n_i times ($\sum_{i=1}^6 n_i = 1000$).

(a) Prove that the Dirichlet distribution is conjugate prior for the multinomial distribution.

(b) What is the posterior distribution of the side probabilities, $\mathbb{P}(p_1, p_2, \ldots, p_6 | n_1, n_2, \ldots, n_6)$?

Linear Regression

1. Optimal MSE rule [10 pts]

Suppose we knew the joint distribution P_{XY}. The optimal rule $f^* : X \to Y$ which minimizes the MSE (Mean Square Error) is given as:

$$f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$$

Show that $f^*(X) = \mathbb{E}[Y|X]$.

2. Ridge Regression [10 pts]

In class, we discussed ℓ_2 penalized linear regression:

$$\widehat{\beta} = \arg\min_\beta \sum_{i=1}^n (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

where $X_i = [X_i^{(1)} \ldots X_i^{(p)}]$.

a) Show that a closed form expression for the ridge estimator is $\widehat{\beta} = (\boldsymbol{A}^\top \boldsymbol{A} + \lambda \boldsymbol{I})^{-1} \boldsymbol{A}^\top \boldsymbol{Y}$ where $\boldsymbol{A} = [X_1; \ldots; X_n]$ and $\boldsymbol{Y} = [Y_1; ...; Y_n]$.

b) An advantage of ridge regression is that a unique solution always exists since $(\boldsymbol{A}^\top \boldsymbol{A} + \lambda \boldsymbol{I})$ is invertible. To be invertible, a matrix needs to be full rank. Argue that $(\boldsymbol{A}^\top \boldsymbol{A} + \lambda \boldsymbol{I})$ is full rank by characterizing its p eigenvalues in terms of the singular values of $\boldsymbol{A}$ and λ.

Logistic Regression

1. Overfitting and Regularized Logistic Regression [10 pts]

a) Plot the sigmoid function $1/(1 + e^{-wX})$ vs. $X \in \mathbb{R}$ for increasing weight $w \in \{1, 5, 100\}$. A qualitative sketch is enough. Use these plots to argue why a solution with large weights can cause logistic regression to overfit.

b) To prevent overfitting, we want the weights to be small. To achieve this, instead of maximum conditional likelihood estimation M(C)LE for logistic regression:

$$\max_{w_0,\ldots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \ldots, w_d),$$

we can consider maximum conditional a posterior M(C)AP estimation:

$$\max_{w_0,\ldots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \ldots, w_d) P(w_0, \ldots, w_d)$$

where $P(w_0, \ldots, w_d)$ is a prior on the weights.

Assuming a standard Gaussian prior $\mathcal{N}(0, \boldsymbol{I})$ for the weight vector, derive the gradient ascent update rules for the weights.

2. Multi-class Logistic Regression [10 pts]

One way to extend logistic regression to multi-class (say K class labels) setting is to consider (K-1) sets of weight vectors and define

$$P(Y = y_k|X) \propto \exp(w_{k0} + \sum_{i=1}^{d} w_{ki} X_i) \text{ for } k = 1, \ldots, K-1$$

a) What model does this imply for $P(Y = y_K|X)$?

b) What would be the classification rule in this case?

c) Draw a set of training data with three labels and the decision boundary resulting from a multi-class logistic regression. (The boundary does not need to be quantitatively correct but should qualitatively depict how a typical boundary from multi-class logistic regression would look like.)

Naive Bayes Classifier

1. Naive Bayes Classification Implementation [25 pts]

In this question, you will write a Naive Bayes classifier and verify its performance on a news-group data-set. As you learned in class, Naive Bayes is a simple classification algorithm that makes an assumption about conditional independence of features, but it works quite well in practice. You will implement the Naive Bayes algorithm (Multinomial Model) to classify a news corpus into 20 different categories.

Handout - `http://www.cs.cmu.edu/~aarti/Class/10701_Spring14/assignments/homework1.tar`

You have been provided with the following data files in the handout:

- train.data - Contains bag-of-words data for each training document. Each row of the file represents the number of occurrences of a particular term in some document. The format of each row is (docId, termId, Count).
- train.label - Contains a label for each document in the training data.

- test.data - Contains bag-of-words data for each testing document. The format of this file is the same as that of the train.data file.
- test.label - Contains a label for each document in the testing data.

For this assignment, you need to write code to complete the following functions:

- logPrior(trainLabels) - Computes the log prior of the training data-set. (5 pts)
- logLikelihood(trainData, trainLabels) - Computes the log likelihood of the training data-set. (7 pts)
- naiveBayesClassify(trainData, trainLabels, testData) - Classifies the data using the Naive Bayes algorithm. (13 pts)

Implementation Notes

1. We compute the log probabilities to prevent numerical underflow when calculating multiplicative probabilities. You may refer to this article on how to perform addition and multiplication in log space.
2. You may encounter words during classification that you haven't during training. This may be for a particular class or over all. Your code should deal with that. Hint: Laplace Smoothing
3. Be memory efficient and please do not create a document-term-matrix in your code. That would require upwards of 600MB of memory.

Due to popular demand, we are allowing the solution to be coded in 3 languages: Octave, Julia, and Python.

Octave is an industry standard in numerical computing. Unlike MATLAB, it is an open-source language and has similar capabilities and syntax.

Julia is a popular new open-source language developed for numerical and scientific computing was well as beginning effective for general programming purposes. This is the first time this language is being supported in a CMU course.

Python is an extremely flexible language and is popular in industry and the data science community. Powerful python libraries would not be available to you.

For Octave and Julia, a blank function interface has been provided for you (in the handout). However, for Python, you will need to perform the I/O for the data files and ensure the results are written to the correct output files.

Challenge Question

This question is not graded, but it is highly recommended that you try it. In the above question, we are using all the terms from the vocabulary to make a prediction. This would lead to a lot of noisy features. Although it seems counter-intuitive, classifiers built from a smaller vocabulary perform better because they generalize better over unseen data. Noisy features that are not well-represented often skew the perceived distribution of words, leading to classification errors. Therefore, the classification can be improved by selecting a subset of extremely effective words.

Write a program to select a subset of the words from the vocabulary provided to you and then use this subset to run your naive bayes classification again. Verify changes in accuracy. TF-IDF and Information Theory are good places to start looking.

Support Vector Machines

1. SVM Matching [15 points]

Figure 1 (at the end of this problem) plots SVM decision boundries resulting from using different kernels and/or different slack penalties. In Figure 1, there are two classes of training data, with labels $y_i \in \{-1, 1\}$, represented by circles and squares respectively. The SOLID circles and squares represent the Support Vectors. Determine which plot in Figure 1 was generated by each of the following optimization problems. Explain your reasoning for each choice.

1.
$$\min \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{n} \xi_i$$

s.t. $\forall i = 1, \cdots, n$:
$\xi_i \geq 0$
$(\mathbf{w} \cdot \mathbf{x_i} + b) y_i - (1 - \xi_i) \geq 0$
and $C = 0.1$.

2.
$$\min \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{n} \xi_i \tag{1}$$

s.t. $\forall i = 1, \cdots, n$:
$\xi_i \geq 0$
$(\mathbf{w} \cdot \mathbf{x_i} + b) y_i - (1 - \xi_i) \geq 0$
and $C = 1$.

3.
$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{2}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v} + (\mathbf{u} \cdot \mathbf{v})^2$.

4.
$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{3}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \exp(-\frac{\|\mathbf{u}-\mathbf{v}\|^2}{2})$.

5.
$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{4}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \exp(- \parallel \mathbf{u} - \mathbf{v} \parallel^2)$.

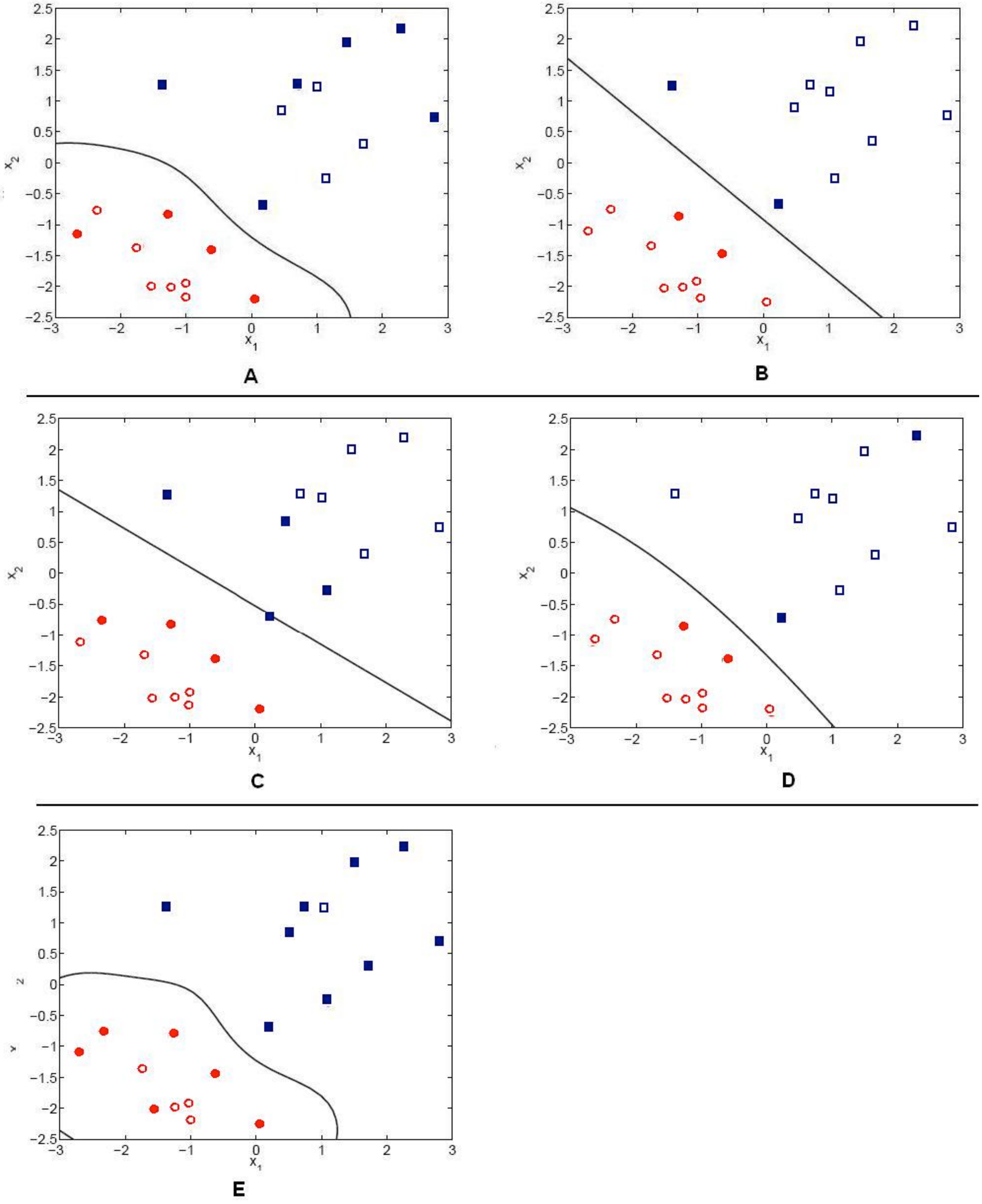

Figure 1: Induced Decision Boundaries