Machine Learning 1 Homework Week 3

Monday, September 14, 2015
Deadline: Thursday, September 24, 2015, 19:59

1 Naive Bayes Spam Classification

Naive Bayes is a particular form of classification that makes strong independence assumptions regarding the features of the data, conditional on the classes (see Bishop section 4.2.3). Specifically, NB assumes each feature is independent given the class label. In contrast, when we looked at probabilistic generative models for classification in the lecture, we used a full-covariance Gaussian to model data from each class, which incorporates correlation between all the input features (i.e. they are not conditionally independent).

If correlated features are treated independently, the evidence for a class gets overcounted. However, Naive Bayes is very simple to construct because, by ignoring correlations, the class-conditional likelihood is a product of D univariate distributions, each of which is simple to learn:

$$p(\mathbf{x}|\mathcal{C}_k) = \prod_{d=1}^{D} p(x_d|\mathcal{C}_k)$$

Consider a spam filter that classifies your emails into two classes $\mathcal{C}_1$ (spam) and $\mathcal{C}_2$ (non-spam). To do this you first make a *bag-of-words* (BoW) representation of your entire training set (a bunch of spam emails and non-spam emails). A BoW is a vector of dimension D of word counts, one for each document (i.e. the words go into a bag and are shaken, losing their order so only their count matters). You can think of D as the vocabulary size of the training set, but it may also be tokens or special features you think are important for spam detection. Your training set is therefore an N by D matrix of word counts $\mathbf{X}$, and $\mathbf{t}$, where $t_n = 0$ if $n \in \mathcal{C}_1$ and $t_n = 1$ if $n \in \mathcal{C}_2$. Assume we know $p(\mathcal{C}_1) = \pi_1$ (and $\pi_2 = 1 - \pi_1$). We can model the word counts using different distributions, for this question we will use a Poisson

distribution model:

$$\begin{aligned} p(x_d|\mathcal{C}_k, \theta_{dk}) &= \mathcal{P}\left(x_d|\lambda_{dk}\right) \\ &= \frac{\lambda_{dk}^{x_d}}{x_d!}\exp(-\lambda_{dk}) \end{aligned}$$

i.e. the parameters $\theta_{dk} = \lambda_{dk}$.

With this information answer the following questions:

1. Write down the likelihood for the *general* two class naive Bayes classifier.
2. Write down the likelihood for the Poisson model.
3. Write down the log-likelihood for the Poisson model.
4. Solve for the MLE estimators for λ_{dk}.
5. Write $p(\mathcal{C}_1|\mathbf{x})$ for the *general* two class naive Bayes classifier.
6. Write $p(\mathcal{C}_1|\mathbf{x})$ for the Poisson model.
7. Rewrite $p(\mathcal{C}_1|\mathbf{x})$ as a sigmoid $\sigma(a) = \frac{1}{1+\exp(-a)}$; solve for a for the Poisson model.
8. Assume $a = \mathbf{w}^T\mathbf{x} + w_0$; solve for $\mathbf{w}$ and w_0.
9. Is the decision boundary a linear function of $\mathbf{x}$? Why?

2 Multi-class Logistic Regression

In class we saw the binary classification version of logistic regression. Here you will derive the gradients for the general case $K > 2$. Much of the preliminaries are in Bishop 4.3.4. This will be useful for Lab 2.

For $K > 2$ the posterior probabilities take a generalized form of the sigmoid called the softmax:

$$y_k(\boldsymbol{\phi}) = p(\mathcal{C}_k|\boldsymbol{\phi}) = \frac{\exp(a_k)}{\sum_i \exp(a_i)}$$

where $a_k = \mathbf{w}_k^T\boldsymbol{\phi}$. NB: The posterior for class k depends on all the other classes i; keep this in mind when working out the derivatives for $\mathbf{w}_k$. The training set is a pair of matrices $\boldsymbol{\Phi}$ and $\mathbf{T}$. Each row of $\mathbf{T}$ uses a one-hot encoding of the class labeling for that training example.

Answer the following questions:

1. Derive $\frac{\partial y_k}{\partial \mathbf{w}_j}$. Bishop uses an indicator function I_{kj}, entries of the identity matrix; previously we used $[k = j]$—they are the same thing.

2. Write down the likelihood as a product over N and K then write down the log-likelihood. Use the entries of $\mathbf{T}$ as selectors of the correct class.

3. Derive the gradient of the log-likelihood with respect to $\mathbf{w}_j$.

4. What is the objective function we minimize that is equivalent to maximizing the log-likelihood?

5. Write a stochastic gradient algorithm for logistic regression using this objective function. Make sure to include indices for time and to define the learning rate. The gradients may differ in sign switching from maximizing to minimizing; don't overlook this.

6. Explain why is this a stochastic optimization procedure?

7. Logistic regression is not free from overfitting. How would you modify the cross-entropy error to regularize your weights? Write down the new objective. If we optimized for $\mathbf{w}$, would this be the maximum likelihood estimator or the maximum-a-posterior estimator?