CS195f Homework 3

Mark Johnson and Erik Sudderth

Homework due at 2pm, 5th November 2009

This problem set asks you to investigate exponential or "Maximum Entropy" classifiers. These involve probability distributions of the form:

$$\begin{aligned} \mathrm{P}(y \mid x) &= \frac{1}{Z_x(\boldsymbol{w})} \exp\left(\boldsymbol{w} \cdot \boldsymbol{f}(y, x)\right), \text{ where} \\ Z_x(\boldsymbol{w}) &= \sum_{y' \in \mathcal{Y}} \exp\left(\boldsymbol{w} \cdot \boldsymbol{f}(y', x)\right) \end{aligned}$$

where:

- $y \in \mathcal{Y}$ is the class label we want to predict,
- $x \in \mathcal{X}$ are the conditioning or predictive variables,
- $\boldsymbol{f}(y, x) \in \mathbb{R}^m$ is an m-dimensional feature vector for pair (y, x), and
- $\boldsymbol{w} \in \mathbb{R}^m$ is an m-dimensional weight vector, where w_j is the weight corresponding to feature $f_j(y, x)$.

Learning MaxEnt classifiers involves finding the weight vectors $\boldsymbol{w}$ given training data D and the vector of feature functions $\boldsymbol{f}$.

We'll use a uniform Gaussian prior on the feature weights $\boldsymbol{w}$, i.e.:

$$\mathrm{P}(w) \propto \exp\left(-\alpha \boldsymbol{w} \cdot \boldsymbol{w}\right)$$

where α is a user-settable parameter that controls the degree of regularization.

Question 1:

1. *Give an expression for the regularized negative log conditional likelihood of a generic data set $D = ((x_1, y_1), \ldots, (x_n, y_n))$, ignoring any terms and factors that do not depend on $\boldsymbol{w}$.*
2. *Give an expression for the derivative of the regularized negative log likelihood with respect to a feature weight w_j.*

Now we will construct an estimator for the feature weights $\boldsymbol{w}$. We will use the Nursery data set that was used in previous exercises, which you can find in

`/course/cs195f/asgn/naive_bayes/handout/nursery/nursery.mat`.

You should divide this data into equal-sized training, development and testing data sets as follows (the `reset` ensures that we'll all use the same training/test split).

```
load('/course/cs195f/asgn/naive_bayes/handout/nursery/nursery.mat');
reset(RandStream.getDefaultStream)
data = data(randperm(size(data,1)),:);
train = data(1:size(data,1)/3,:);
dev = data(size(data,1)/3+1:2*size(data,1)/3,:);
test = data(2*size(data,1)/3+1:end,:);
```

In this data set, each conditioning variable x_i is in fact a vector with $m = 8$ components or attributes, so $x_i = (x_{i,1}, \ldots, x_{i,8})$. Let $\mathcal{X}_j$ be the range of the jth attribute, so $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_8$. Let $\mathcal{Y}$ be the set of class labels.

The features you use should contain the following feature functions:

- For each $y' \in \mathcal{Y}$, include a feature function $f_{y'}(y, \boldsymbol{x}) = \mathbb{I}(y, y')$, where $\mathbb{I}(u, v)$ is the indicator function (i.e., $\mathbb{I}(u, v) = 1$ iff $u = v$).
- For each $j \in 1, \ldots, 8$, each $x' \in \mathcal{X}_j$ and each $y' \in \mathcal{Y}$, include a feature function $f_{y',j,x'}(y, \boldsymbol{x}) = \mathbb{I}(y, y')\,\mathbb{I}(x_j, x')$

You should implement a MaxEnt estimator using the `fminunc` function from the Matlab optimization toolbox to find the weight vector $\boldsymbol{w}$ that minimizes the regularized negative log conditional likelihood on the training data. You should provide the optimizer with the derivative of the regularized negative log likelihood.

The following Matlab commands tells `fminunc` to use the gradient information and perform at most 1,000 iterations.

```
options = optimset('MaxIter', 1000, 'GradObj', 'on');
argminw = fminunc(myfun, w0, options);
```

Here `w0` is an initial guess at the weight vector (a vector of all zeros is fine) and `myfun` is a function that you've written that takes a weight vector `w` as its argument and returns a pair of values; viz., the regularized negative log conditional likelihood and a vector of its derivatives. Other options you might find useful are

```
options = optimset('MaxIter', 100, 'GradObj', 'on', 'Display', 'iter');
```

which makes the optimizer print out debugging information on each iteration.

You may find it useful to pass parameters to `myfun` by using a *function handle*:

```
argminw = fminunc(@(u) myfun(u, e1, e2,...), w0, options);
```

where `e1`, `e2`, etc., are expressions whose values you want to pass to `myfun`.

For all of these questions, do all calculations for 20 logarithmically-spaced values of α between 10^{-4} and 10 as follows:

```
alpha = logspace(-4,1,20);
```

Deqing (our fearless grad TA) has prepared a code skeleton to help you, which you can find in `/course/cs195f/asgn/MaxEnt`.

Question 2:

1. *Plot the negative log conditional likelihood of the* `train` *data as a function of* α *when training and evaluating on* `train`*.*

2. *Plot of the accuracy of the classifier as a function of* α *when training and evaluating on* `train`*.*

3. *Plot the negative log conditional likelihood of the* `dev` *data as a function of* α*. That is, for each value of* α *estimate the feature weight* $\boldsymbol{w}$ *from the* `train` *data, and then calculate the negative log conditional likelihood of the* `dev` *data for that value of* $\boldsymbol{w}$*. Find a value of* α *which minimizes the negative log conditional likelihood of the* `dev` *data.*

4. *Plot the accuracy of the classifier when training on* `train` *and evaluating on* `dev` *data as a function of* α*. Find a value of* α *which maximizes the accuracy on the* `dev` *data.*

5. *For the weight vectors* $\boldsymbol{w}$ *corresponding to two values of* α *identified in (3) and (4), evaluate the corresponding classifier on* `test` *and report your results.*

Graduate credit problem section

This next question asks you to develop a maximum likelihood estimator for MaxEnt models for the situation where the training data is only *partially observed*. To keep the maths simple, we'll work with unconditioned models. That is, we're trying to learn the feature weights $\boldsymbol{w}$ for a model of the form:

$$\begin{aligned} \mathrm{P}_{\boldsymbol{w}}(y) &= \frac{1}{Z(\boldsymbol{w})} \exp\left(\boldsymbol{w} \cdot \boldsymbol{f}(y)\right) \\ Z(\boldsymbol{w}) &= \sum_{y' \in \mathcal{Y}} \exp\left(\boldsymbol{w} \cdot \boldsymbol{f}(y')\right) \end{aligned}$$

In class we gave the formulae for the log likelihood and its derivatives with respect to w_j for the situation where the data is *fully observed*, i.e., where $\mathcal{D} = (y_1, \ldots, y_n)$ and each $y_i \in \mathcal{Y}$. For this question, assume that the data is *partially observed*, i.e., where $\mathcal{D} = (\mathcal{Y}_1, \ldots, \mathcal{Y}_n)$ and each $\emptyset \subset \mathcal{Y}_i \subseteq \mathcal{Y}$. That is, the ith observation of the training data specifies a *subset* $\mathcal{Y}_i$ of values in which the true value lies (i.e., $y_i \in \mathcal{Y}_i$) but doesn't identify y_i itself.

Question 3: (graduate credit)

1. *Give the log likelihood of* $\boldsymbol{w}$ *given partially observed data* $\mathcal{D}$*.*

2. *Give the derivative of the log likelihood with respect to weight* w_j *in terms of expectations involving* $\mathrm{P}_{\boldsymbol{w}}$*.*