COMS 4771 Machine Learning (Spring 2015)
Problem Set #2

Solutions - coms4771@machinelearning.nyc
Discussants: None

February 27, 2015

Problem 1

(a) The log-likelihood of the parameters $\boldsymbol{\theta} = (\pi_1, \ldots, \pi_K, p_{1,1}, \ldots, p_{d,K})$ given the training data S is

$$
\begin{aligned}
&\ln \prod_{(\boldsymbol{x},y)\in S} \Pr[\boldsymbol{X} = \boldsymbol{x} \wedge Y = y] \\
&= \sum_{(\boldsymbol{x},y)\in S} \ln \Pr[\boldsymbol{X} = \boldsymbol{x} \wedge Y = y] \\
&= \sum_{(\boldsymbol{x},y)\in S} \ln \left(\prod_{k=1}^{K} \Pr[\boldsymbol{X} = \boldsymbol{x} \wedge Y = k]^{\mathbb{1}\{y=k\}} \right) \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \ln \Pr[\boldsymbol{X} = \boldsymbol{x} \wedge Y = k] \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \ln \left(\Pr[Y = k] \Pr[\boldsymbol{X} = \boldsymbol{x} \,|\, Y = k] \right) \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \ln \left(\Pr[Y = k] \prod_{j=1}^{d} \Pr[X_j = x_j \,|\, Y = k] \right) \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \ln \left(\pi_k \prod_{j=1}^{d} (1 - p_{j,k})^{1-x_j} p_{j,k}^{x_j} \right) \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \left(\ln(\pi_k) + \sum_{j=1}^{d} \left((1 - x_j) \ln(1 - p_{j,k}) + x_j \ln(p_{j,k}) \right) \right) \\
&= \sum_{(\boldsymbol{x},y)\in S} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \ln(\pi_k) + \sum_{(\boldsymbol{x},y)\in S} \sum_{j=1}^{d} \sum_{k=1}^{K} \mathbb{1}\{y = k\} \left((1 - x_j) \ln(1 - p_{j,k}) + x_j \ln(p_{j,k}) \right).
\end{aligned}
$$

(It's not necessary to show all of those steps above ....)

Now fix some pair $(j,k) \in [d] \times [K]$. The derivative of the log-likelihood with respect to $p_{j,k}$ is

$$\sum_{(\boldsymbol{x},y)\in S} \mathbb{1}\{y=k\}\left(\frac{x_j}{p_{j,k}} - \frac{1-x_j}{1-p_{j,k}}\right) = \frac{N_{j,k}}{p_{j,k}} - \frac{N_k - N_{j,k}}{1-p_{j,k}}$$

where

- N_k is the number of training examples $(\boldsymbol{x},y) \in S$ with $y=k$, and
- $N_{j,k}$ is the number of training examples $(\boldsymbol{x},y) \in S$ with $x_j = 1$ and $y=k$.

The derivative is zero exactly when

$$p_{j,k} = \frac{N_{j,k}}{N_k}.$$

(b) Suppose $\hat{f}\colon \{0,1\}^d \to \{1,2\}$ is the BNB plug-in classifier with parameters

$$\hat{\boldsymbol{\theta}} = (\hat{\pi}_1, \hat{\pi}_2, \hat{p}_{1,1}, \ldots, \hat{p}_{d,1}, \hat{p}_{1,2}, \ldots, \hat{p}_{d,2}).$$

Then, for any $\boldsymbol{x} \in \{0,1\}^d$, $\hat{f}(\boldsymbol{x}) = 1$ iff

$$\frac{\hat{\pi}_1 \prod_{j=1}^d (1-\hat{p}_{j,1})^{1-x_j}\hat{p}_{j,1}^{x_j}}{\hat{\pi}_2 \prod_{j=1}^d (1-\hat{p}_{j,2})^{1-x_j}\hat{p}_{j,2}^{x_j}} = \prod_{j=1}^d \left(\frac{\hat{p}_{j,1}}{1-\hat{p}_{j,1}} \cdot \frac{1-\hat{p}_{j,2}}{\hat{p}_{j,2}}\right)^{x_j} \cdot \frac{\hat{\pi}_1}{\hat{\pi}_2} \prod_{j=1}^d \left(\frac{1-\hat{p}_{j,1}}{1-\hat{p}_{j,2}}\right) > 1.$$

Taking ln of both sides, we see that $\hat{f}(\boldsymbol{x}) = 1$ iff

$$\sum_{j=1}^d x_j \cdot \underbrace{\ln\left(\frac{\hat{p}_{j,1}}{1-\hat{p}_{j,1}} \cdot \frac{1-\hat{p}_{j,2}}{\hat{p}_{j,2}}\right)}_{w_j} > \underbrace{\ln\left(\frac{\hat{\pi}_2}{\hat{\pi}_1} \prod_{j=1}^d \left(\frac{1-\hat{p}_{j,2}}{1-\hat{p}_{j,1}}\right)\right)}_{\theta}.$$

So the classifier $\hat{f}$ is a linear classifier of the form

$$\hat{f}(\boldsymbol{x}) = \operatorname{sign}\left(\langle \boldsymbol{w}, \boldsymbol{x}\rangle - \theta\right)$$

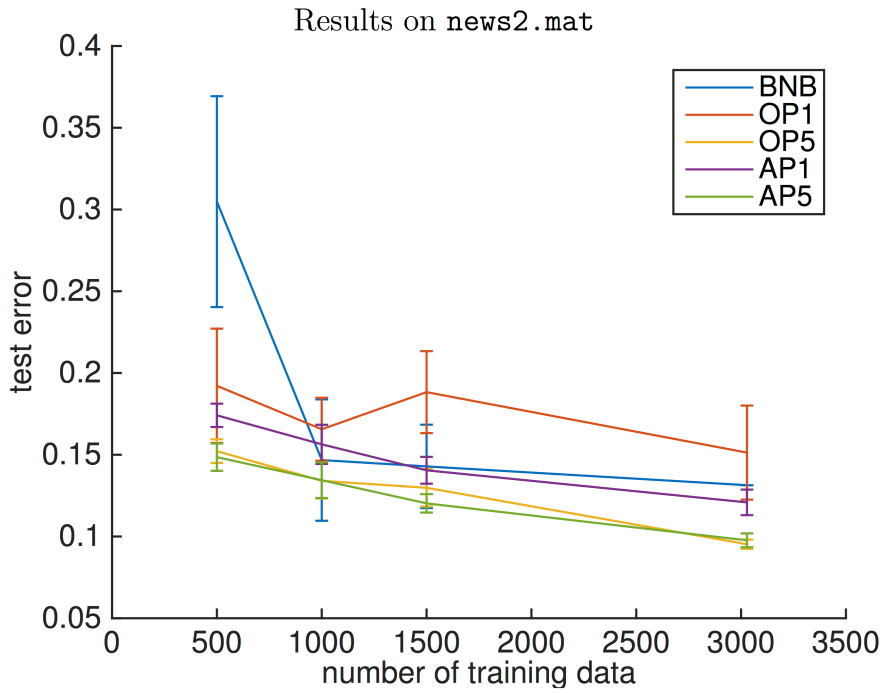
(by mapping class "2" to "-1").

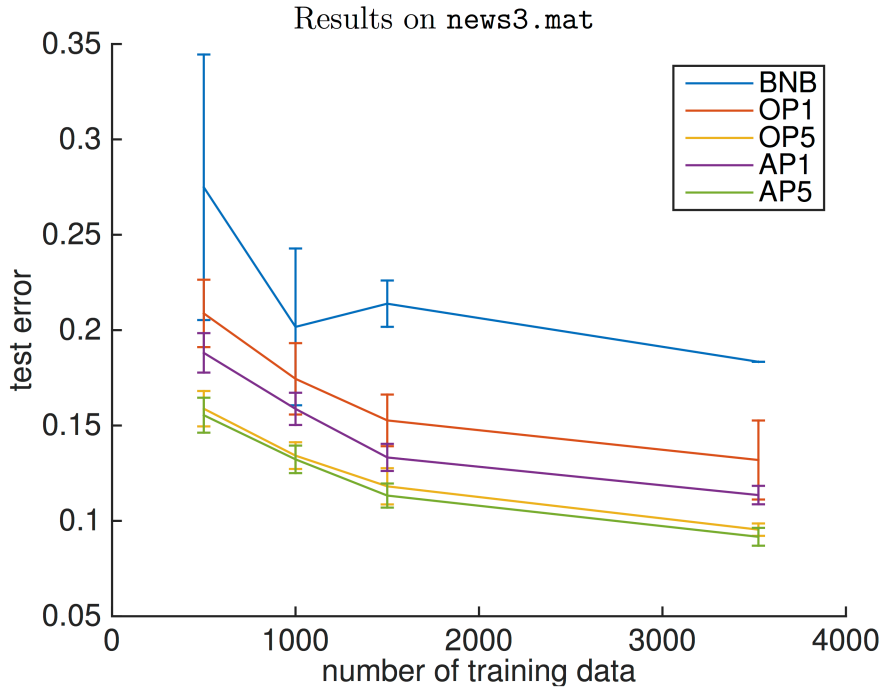
(c) Training error: 0.2163, test error: 0.3760.

Problem 2

(a) **if** $y\left(\langle \boldsymbol{w}, \boldsymbol{x} \rangle - \theta\right) \leq 0$ **then**
$\boldsymbol{w} := \boldsymbol{w} + y\boldsymbol{x}$; $\theta := \theta - y$.
end if

(b)





Most (but not all) variants of Perceptron seems to dominate BNB.

Multiple passes helps both Online Perceptron and Averaged Perceptron; it has more of an effect on Online Perceptron than it does on Averaged Perceptron. Multiple passes seems to be more important than averaging. (However, it is not clear if many more passes would be beneficial.)

(c)

`news2.mat` top words

BNB word	BNB weight	AP5 word	AP5 weight
athos	-5.18157	bible	-303070
atheism	-4.79569	god	-274201
atheists	-4.7642	jesus	-234455
clh	-4.73184	christian	-229330
firearms	4.6845	church	-218180
occupied	4.54867	israeli	202541
teachings	-4.4191	news	199433
israelis	4.40706	israel	187028
serdar	4.39442	arab	185619
argic	4.39442	moral	-184779
ohanus	4.23024	gun	182665
appressian	4.23024	government	182278
sahak	4.23024	religion	-178263
melkonian	4.23024	christianity	-177444
villages	4.2154	athos	-168470
revelation	-4.21364	atheism	-165419
testament	-4.17831	christians	-163410
livesey	-4.1602	christ	-162178
atheist	-4.15256	keith	-162108
solntze	-4.12305	politics	159258

`news3.mat` top words

BNB word	BNB weight	AP5 word	AP5 weight
encryption	5.54713	windows	-445271
nsa	4.94057	graphics	-326501
escrow	4.88184	space	257130
secret	4.8469	image	-252661
pgp	4.82903	power	247940
crypto	4.67505	encryption	227290
enforcement	4.42174	key	225124
government	4.35999	window	-211012
motif	-4.26096	pgp	194462
xlib	-4.23046	circuit	184868
lunar	4.20815	low	184178
font	-4.18142	motif	-183014
voltage	4.15988	package	-181013
wiretap	4.15988	electronics	177416
xterm	-4.14744	mouse	-176339
eff	4.10935	government	167992
orbit	4.09871	orbit	161300
denning	4.09196	voltage	160850
sternlight	4.07428	file	-155418
vehicle	4.05631	win	-154698

Problem 3

(a) Centering cannot affect the classifier: the estimated class conditional means will all be shifted by $\boldsymbol{\mu}$, and the classification is based on Euclidean distances which are invariant to translation. But standardization can affect the classifier. One way to see why is by an example. Suppose the data set $S \subset \mathbb{R}^2 \times \{-1, 1\}$ is comprised of the following four training examples:

	x_1	x_2	y
Example 1	$-1 - 10^6$	$+1$	$+1$
Example 2	$-1 + 10^6$	$+1$	$+1$
Example 3	$+1 - 10^6$	-1	-1
Example 4	$+1 + 10^6$	-1	-1

Here, $\boldsymbol{\mu} = (0, 0)$, but $\sigma_1 = 10^6$ and $\sigma_2 = 1$. Without standardization, the decision boundary is the line $x_1 = x_2$; with it, the decision boundary is very close to the line $x_2 = 0$ (i.e., x_1 is almost completely ignored). In particular, the classification of the point $\boldsymbol{x} = (1, 1/2)$ changes.

(b) Centering cannot affect the classifier since Euclidean distances are invariant to translation. But standardization can affect the classifier. Suppose the data set $S \subset \mathbb{R}^2 \times \{-1, 1\}$ is comprised of the following four training examples:

	x_1	x_2	y
Example 1	$-\sqrt{2} \times 10^6$	0	$+1$
Example 2	-1	$+\sqrt{2}$	$+1$
Example 3	$+\sqrt{2} \times 10^6$	0	-1
Example 4	$+1$	$-\sqrt{2}$	-1

Here, $\boldsymbol{\mu} = (0, 0)$, but σ_1 is a little larger than 10^6 and $\sigma_2 = 1$. Without standardization, the prediction on $\boldsymbol{x} = (1, 1/2)$ is -1; with it, the prediction on $\boldsymbol{x} = (1, 1/2)$ is $+1$.

(c) Neither centering nor standardization affects the classifier. This is because for any data set S and any coordinate j, the greedy step of the algorithm considers the same set of possible ways to split S, whether or not the transformations are applied.

(d) Neither centering nor standardization affects the classifier. The set of linear classifiers in $\mathbb{R}^d$ is invariant to any (invertible) affine transformations. So ERM considers the same set of linear classifiers whether or not the transformation (centering or standardization) is applied.

Actually, (c) and (d) are not quite right because there may be multiple best axis-aligned splits (in (c)) or multiple linear classifiers that minimize the training error (in (d)). It is possible, depending on any "tie-breaking" rule being applied, that the centering or standardization affects which actual splitting rule or linear classifier is returned in the event of a tie. So it is also acceptable to say that *both* centering and standardization can affect the classifier in these cases, provided that you explain this issue. (Ties are also potentially an issue with (a) and (b) at prediction time, but this is a less significant issue.)

Problem 4

(a) Let $\boldsymbol{\phi}\colon \mathbb{R}^d \to \mathbb{R}^{d+\binom{d}{2}}$ be the feature map defined by

$$\boldsymbol{\phi}(\boldsymbol{x}) := (\underbrace{x_1^2, \ldots, x_d^2}_{x_i^2 : i \in [d]}, \underbrace{\sqrt{2}x_1x_2, \ldots, \sqrt{2}x_1x_d, \sqrt{2}x_2x_3, \ldots, \sqrt{2}x_2x_d, \ldots, \sqrt{2}x_{d-1}x_d}_{\sqrt{2}x_ix_j : 1 \le i < j \le d}).$$

Then

$$\langle \boldsymbol{x}, \boldsymbol{x}' \rangle^2 = \left(\sum_{i=1}^d x_i x_i' \right)^2 = \sum_{i=1}^d x_i^2 (x_i')^2 + 2 \sum_{1 \le i < j \le d} x_i x_i' x_j x_j' = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}') \rangle.$$

(b) Let $\boldsymbol{\phi}_1\colon \mathbb{R}^d \to \mathbb{R}^D$ be the feature map corresponding to K_1, and let $\boldsymbol{\phi}_2\colon \mathbb{R}^d \to \mathbb{R}^D$ be the feature map corresponding to K_2. We define a feature map $\boldsymbol{\phi}_3\colon \mathbb{R}^d \to \mathbb{R}^{D \times D}$ so that $\boldsymbol{\phi}_3(\boldsymbol{x}) \in \mathbb{R}^{D \times D}$ is a matrix whose (i,j)-th entry is $\phi_{1,i}(\boldsymbol{x})\phi_{2,j}(\boldsymbol{x})$. (Here, I use $\phi_{a,i}(\boldsymbol{x})$ to denote the i-th entry in $\boldsymbol{\phi}_a(\boldsymbol{x})$ for $a \in \{1,2\}$.) Then

$$\begin{aligned}
K_3(\boldsymbol{x}, \boldsymbol{x}') &= K_1(\boldsymbol{x}, \boldsymbol{x}') \cdot K_2(\boldsymbol{x}, \boldsymbol{x}') \\
&= \langle \boldsymbol{\phi}_1(\boldsymbol{x}), \boldsymbol{\phi}_1(\boldsymbol{x}') \rangle \cdot \langle \boldsymbol{\phi}_2(\boldsymbol{x}), \boldsymbol{\phi}_2(\boldsymbol{x}') \rangle \\
&= \left(\sum_{i=1}^D \phi_{1,i}(\boldsymbol{x}) \phi_{1,i}(\boldsymbol{x}') \right) \left(\sum_{j=1}^D \phi_{2,j}(\boldsymbol{x})_i \phi_{2,j}(\boldsymbol{x}') \right) \\
&= \sum_{i=1}^D \sum_{j=1}^D \phi_{1,i}(\boldsymbol{x}) \phi_{1,i}(\boldsymbol{x}') \phi_{2,j}(\boldsymbol{x})_i \phi_{2,j}(\boldsymbol{x}') \\
&= \langle \boldsymbol{\phi}_3(\boldsymbol{x}), \boldsymbol{\phi}_3(\boldsymbol{x}') \rangle.
\end{aligned}$$

Here, in the last step, we treat $D \times D$ matrices as D^2-dimensional vectors when considering the inner product between them.

hw2_train_bnb.m

```
function params = hw2_train_bnb(X,Y,alpha)

if nargin < 3
  alpha = 1;
end

params.classes = unique(Y);
K = length(params.classes);
params.w = zeros(K,size(X,2));
params.theta = zeros(K,1);

for k = 1:K
  y = params.classes(k);
  p = (alpha + sum(X(Y == y,:))) / (2*alpha + sum(Y == y));
  params.w(k,:) = log(p ./ (1-p));
  params.theta(k) = -sum(log(1-p),2) - log(mean(Y == y));
end
```

hw2_test_bnb.m

```
function preds = hw2_test_bnb(params,test)

scores = bsxfun(@minus,test * params.w', params.theta');
[~,I] = max(scores,[],2);
preds = params.classes(I);
```

hw2_train_perc.m

```
function params = hw2_train_perc(X,Y,num_passes)

if nargin < 3
  num_passes = 1;
end
params.w = zeros(1,size(X,2));
params.theta = 0;

for pass=1:num_passes
  for t=1:size(X,1)
    if Y(t) * (dot(X(t,:), params.w) - params.theta) <= 0
      params.w = params.w + Y(t) * X(t,:);
      params.theta = params.theta - Y(t);
    end
  end
end
```

hw2_train_avgperc.m

```
function params = hw2_train_avgperc(X,Y,num_passes)

if nargin < 3
  num_passes = 1;
end
d = size(X,2);
params.w = zeros(1,d);
params.theta = 0;

w = zeros(1,d);
theta = 0;
c = 1;

for pass=1:num_passes
  for t=1:size(X,1)
    if Y(t) * (dot(X(t,:), w) - theta) <= 0
      params.w = params.w + c * w;
      params.theta = params.theta + c * theta;
      w = w + Y(t) * X(t,:);
      theta = theta - Y(t);
      c = 1;
    else
      c = c + 1;
    end
  end
end
params.w = params.w + c * w;
params.theta = params.theta + c * theta;
```

hw2_test_perc.m

```
function preds = hw2_test_perc(params,test)

preds = 2*(test * params.w' > params.theta) - 1;
```