CS534 — Homework Assignment 1 — Due Oct 6th 11:59pm, 2018

Written assignment

Individual assignment. Submit electronically via TEACH (https://teach.engr.oregonstate.edu/).

1. (Probability) Consider two coins, one is fair and the other one has a 1/10 probability for head. Now you randomly pick one of the coins, and toss it twice. Answer the following questions.

   (a) What is the probability that you picked the fair coin? What is the probability of the first toss being head?

   Let x_1 denote the outcome of the first toss and let y denote the coin that is selected. We can write down the following probabilities.
   $P(y = f) = P(y = uf) = \frac{1}{2}$
   $P(x_1 = h|y = f) = \frac{1}{2}$
   $p(x_1 = h|y = uf) = 1/10$

   Now we can write out the probability of the first toss being head as:
   $P(x_1 = h) = P(x_1 = h, y = f) + P(x_1 = h, y = uf)$
   $= P(y = f)P(x_1 = h|y = f) + P(y = uf)P(x_1 = h|y = uf) = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{10}$
   $= \frac{1}{2} \times \frac{6}{10} = \frac{3}{10}$

   (b) If both tosses are heads, what is the probability that you have chosen the fair coin (Hint: Bayes Rule)?

   Let x_1, x_2 denote the outputs of the first two tosses. It is easy to see that
   $P(x_1 = h, x_2 = h|y = f) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$
   $P(x_1 = h, x_2 = h|y = uf) = \frac{1}{10} \times \frac{1}{10} = \frac{1}{100}$

   Now we need to compute $P(y = f|x_1 = h, x_2 = h)$, to do so, we use Bayes Theorem:
   $P(y = f|x_1 = h, x_2 = h) = \frac{P(x_1=h,x_2=h|y=f)P(y=f)}{P(x_1=h,x_2=h)}$
   To compute the denominator, we use the same approach as used in (a):
   $P(x_1 = h, x_2 = h) = P(x_1 = h, x_2 = h|y = f)P(y = f) + P(x_1 = h, x_2 = h|y = uf)P(y = uf)$
   $= \frac{1}{4} \times \frac{1}{2} + \frac{1}{100} \times \frac{1}{2} = \frac{13}{100}$

   Plug this into the Bayes Theorem, we have:
   $P(y = f|x_1 = h, x_2 = h) = \frac{P(x_1=h,x_2=h|y=f)P(y=f)}{P(x_1=h,x_2=h)} = \frac{1/4 \times 1/2}{13/100} = \frac{25}{26}$

2. (Maximum likelihood estimation for uniform distribution.) Given a set of i.i.d. samples $x_1, x_2, ..., x_n \sim uniform(0, \theta)$.

   (a) Write down the likelihood function of θ.

$$L(\theta) = \prod_{i=1}^{n} p(x_i; \theta) \tag{1}$$

$$= \prod_{i=1}^{n} \frac{1}{\theta} I(x_i \leq \theta) = \begin{cases} \frac{1}{\theta^n} & \text{if } \forall x_i \leq \theta \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

   (b) Find the maximum likelihood estimator for θ.

$$\underset{\theta}{\operatorname{argmax}}\, L = \underset{\theta}{\operatorname{argmax}}\, \frac{1}{\theta^n} \quad \text{subject to} \quad \forall x_i \leq \theta \tag{3}$$

$$= \max\{x_1, ..., x_n\} = x_{max} \tag{4}$$

From (3) to (4) is because in order to maximize L, we want θ to be as small as possible, but θ has to be no smaller than any observed values (otherwise L goes to zero). So the smallest value θ is allowed to take is x_{max}.

3. (Weighted linear regression) In class when discussing linear regression, we assume that the Gaussian noise is independently identically distributed. Now we assume the noises $\epsilon_1, \epsilon_2, \cdots \epsilon_n$ are independent but each $\epsilon_m \sim N(0, \sigma_m^2)$, i.e., it has its own distinct variance.

   (a) Write down the log likelihood function of $\mathbf{w}$.

$$l(\mathbf{w}) = \log p(D|M) = \sum_{i=1}^{n} \log N(y_i|\mathbf{x}_i^T\mathbf{w}, \sigma_i^2) \tag{5}$$

$$= \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2\sigma_i^2}(y_i - \mathbf{x}_i^T\mathbf{w})^2} \tag{6}$$

$$= \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi}\sigma_i} - \sum_{i=1}^{N} \frac{1}{2\sigma_i^2}(y_i - \mathbf{x}_i^T\mathbf{w})^2 \tag{7}$$

   (b) Show that maximizing the log likelihood is equivalent to minimizing a weighted least square loss function $J(\mathbf{W}) = \frac{1}{2}\sum_{m=1}^{n} a_m(\mathbf{w}^T\mathbf{x}_m - y_m)^2$, and express each a_m in terms of σ_m.
   Maximizing the log likelihood is equivalent to minimizing the second term in the above equation, which can be represented as:

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1,\ldots n} a_i(y_i - \mathbf{x}_i^T\mathbf{w})^2 \tag{8}$$

   where $a_i = \frac{1}{\sigma_i^2}$.

   (c) Derive a batch gradient descent algorithm for optimizing this objective.
   Take the gradient of J (using equation 8):

$$\nabla J(\mathbf{w}) = \sum_i a_i(y_i - \mathbf{x}_i^T\mathbf{w})\mathbf{x}_i \tag{9}$$

   I will not write out the full batch gradient algorithm, but it will look very similar to the one presented in class, with a slightly different update rule:

$$\mathbf{w} \leftarrow \mathbf{w} - \lambda \sum_i a_i(y_i - \mathbf{x}_i^T\mathbf{w})\mathbf{x}_i$$

   As can be seen from this solution, this is equivalent to weighting each instance differently according to the noise variance for each instance. Instances with larger noise variance are less reliable (due to larger noise), and thus contributes less (due to smaller weight) in the learning process, which is intuitive.

   (d) Derive a closed form solution to this optimization problem.
   Let $\mathbf{y} = \frac{1}{2}\{y_1, y_2, \cdots, y_N\}^T$, $\mathbf{X}$ be the data matrix whose rows correspond to training examples, and A be a diagonal matrix with $A(i,i) = a_i$. Rewriting the objective in matrix form:

$$J(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^T A(\mathbf{y} - \mathbf{X}\mathbf{w})$$

   Take the gradient of J in vector form (you can also start from equation 9 and write in the vector form) and set it to zero:

$$\nabla J(\mathbf{w}) = \mathbf{X}^T A(\mathbf{X}\mathbf{w} - \mathbf{y}) = 0$$
$$2\mathbf{X}^T A\mathbf{X}\mathbf{w} = 2\mathbf{X}^T A\mathbf{y}$$
$$\mathbf{w}^* = (\mathbf{X}^T A\mathbf{X})^{-1}\mathbf{X}^T A\mathbf{y}$$

4. (Decision theory). Consider a binary classification task with the following loss matrix:

predicted label $\hat{y}$	true label y	
	0	1
0	0	10
1	5	0

We have build a probabilistic model that for each example x gives us an estimated $P(y = 1|x)$. It can be shown that, to minimize the expected loss for our decision, we should set a probability threshold θ and predict $\hat{y} = 1$ if $P(y = 1|x) > \theta$ and $\hat{y} = 0$ otherwise.

(a) Please compute the θ for the above given loss matrix.

We want to predict $\hat{y} = 1$ if the expected loss of predicting 1 is less than the expected loss of predicting zero. The loss for predicting a particular value $\hat{y}$ is:

$$L(\hat{y}, 0) * P(y = 0|x) + L(\hat{y}, 1) * P(y = 1|x)$$

where $L(\hat{y}, y)$ is the loss of predicting $\hat{y}$ when the true label is y. So the loss of predicting 1 is $0 * P(y = 1|x) + 5 * P(y = 0|x)$ whereas the loss of predicting 0 is $10 * P(y = 1|x) + 0 * P(y = 0|x)$. Thus we will predict 1 iff:

$$P(y = 0|\mathbf{x})5 \quad < \quad P(y = 1|\mathbf{x})10 \tag{10}$$

Define $p_1 = P(y = 1|\mathbf{x})$, then this becomes

$$(1 - p_1) \times 5 \quad < \quad p_1 \times 10 \tag{11}$$

$$p_1 \quad > \quad \frac{1}{3} \tag{12}$$

So we should set the threshold to 1/3.

(b) Show a loss matrix where the threshold is 0.1.
To get a threshold of 0.1, we could use the matrix:

predicted label $\hat{y}$	*true label y*	
	0	*1*
0	*0*	*9*
1	*1*	*0*

5. Consider the maximum likelihood estimation problem for multi-class logistic regression using the softmax function defined below:

$$p(y = k|\mathbf{x}) = \frac{\exp(\mathbf{w}_k^T \mathbf{x})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^T \mathbf{x})}$$

We can write out the likelihood function as:

$$L(\mathbf{w}) = \prod_{i=1}^{N} \prod_{k=1}^{K} p(y = k|\mathbf{x}_i)^{I(y_i = k)}$$

where $I(y_i = k)$ is the indicator function, taking value 1 if y_i is k.

(a) What are i and k in this likelihood function? *The first product is over $i = 1$ to N for all instances in the dataset and the second product is for $k = 1$ to K over all classes.*

(b) Compute the log-likelihood function. *Take the log of the likelihood function, we have:*

$$l(\mathbf{w}) = \sum_{i=1}^{N}\sum_{k=1}^{K} I(y_i = k)\log p(y_i = k|\mathbf{x}_i) = \sum_{k=1}^{K}\sum_{y_i=k}\log p(y_i = k|\mathbf{x}_i)$$

(c) What is the gradient of the log-likelihood function w.r.t the weight vector $\mathbf{w}_c$ of class c? (Precursor to this question, which terms are relevant for $\mathbf{w}_c$ in the loglikelihood function?)

We want to take partial gradient with respect to $\mathbf{w}_c$. Before doing that, let's break l into two parts:

$$l(\mathbf{w}) = \sum_{k\neq c}^{K}\sum_{y_i=k}\log p(y_i = k|\mathbf{x}_i) + \sum_{y_i=c}\log p(y_i = c|\mathbf{x}_i)$$

Note that we have

$$p(y_i = k|\mathbf{x}_i) = \frac{\exp \mathbf{w}_k \cdot \mathbf{x}_i}{\sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i)}$$

Take the log:

$$\log p(y_i = k|\mathbf{x}_i) = \mathbf{w}_k \cdot \mathbf{x}_i - \log(\sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i))$$

Let $z_i = \sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i)$, we have:

$$\log p(y_i = k|\mathbf{x}_i) = \mathbf{w}_k \cdot \mathbf{x}_i - \log z_i$$

Plog this into l, we have:

$$l(\mathbf{w}) = \sum_{k\neq c}^{K}\sum_{y_i=k}(\mathbf{w}_k \cdot \mathbf{x}_i - \log z_i) + \sum_{y_i=c}(\mathbf{w}_c \cdot \mathbf{x}_i - \log z_i)$$

Now take the partial gradient:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{w}_c}l &= -\sum_{k\neq c}^{K}\sum_{y_i=k}\frac{1}{z_i}\frac{\partial z_i}{\partial \mathbf{w}_c} + \sum_{y_i=c}\left(\mathbf{x}_i - \frac{1}{z_i}\frac{\partial z_i}{\partial \mathbf{w}_c}\right)\\ &= \sum_{y_i=c}\mathbf{x}_i - \sum_{k=1}^{K}\sum_{y_i=k}\frac{1}{z_i}\frac{\partial z_i}{\partial \mathbf{w}_c}\end{aligned}$$

Note that the second double summation can be simplified to $\sum_{i=1}^{N}$, where N is the total number of points. We now plug in

$$\frac{\partial z_i}{\partial \mathbf{w}_c} = \mathbf{x}_i \exp(\mathbf{w}_c \cdot \mathbf{x}_i)$$

and $z_i = \sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i)$, we have:

$$\frac{\partial}{\partial \mathbf{w}_c}l = \sum_{y_i=c}\mathbf{x}_i - \sum_{i=1}^{N}\frac{\exp(\mathbf{w}_c \cdot \mathbf{x}_i)}{\sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i)}\mathbf{x}_i$$

We will use $\hat{y}_{ic}$ (intuitively meaning the probability of instance i belong to class c) to denote

$$\frac{\exp(\mathbf{w}_c \cdot \mathbf{x}_i)}{\sum_{j=1}^{K}\exp(\mathbf{w}_j \cdot \mathbf{x}_i)}$$

and use y_{ic} to denote a binary indicator variable such that $y_{ic} = 1$ if $y_i = c$ and 0 otherwise. Putting these new notations to use, we arrive at:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{w}_c} l &= \sum_{i=1}^{N} y_{ic}\mathbf{x}_i - \sum_{i=1}^{N} \hat{y}_{ic}\mathbf{x}_i \\ &= \sum_{i=1}^{N} (y_{ic} - \hat{y}_{ic})\mathbf{x}_i\end{aligned}$$

Therefore, a gradient ascent update rule will be:

$$\forall c \in \{1, \cdots, k\}\text{: } \mathbf{w}_c \leftarrow \mathbf{w}_c + \lambda \sum_{i=1}^{N} (y_{ic} - \hat{y}_{ic})\mathbf{x}_i$$