Machine Learning and Data Mining I (DS 4400)
Homework 2 Solutions

$\left\{ \begin{array}{l} \text{Analytical : 58 points} \\ \text{Programming : 42 points} \end{array} \right.$

Instructor: Ehsan Elhamifar
Due Date: February 14, 2020, 1:35pm

1) Linear Regression: Consider the modified linear regression problem

12 points

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \sum_{i=1}^{N} \left(\boldsymbol{\theta}^\top \phi(\boldsymbol{x}_i) - y_i\right)^2 + \lambda \|\boldsymbol{\theta} - \boldsymbol{a}\|_2^2, \tag{1}$$

where $\boldsymbol{a}$ is a known and given vector of the same dimension as that of $\boldsymbol{\theta}$. Derive the closed-form solution for (1). Provide all steps of the derivation.

Solution: Take the derivative of the cost function ($J(\theta)$) wrt θ and set to zero :

$$\frac{\partial J}{\partial \theta} = \sum_i \frac{\partial (\varphi(x_i)^T\theta - y_i)^2}{\partial \theta} + \lambda \frac{\partial \|\theta - a\|_2^2}{\partial \theta}$$

$$= \sum_i 2\,\varphi(x_i)\left(\varphi(x_i)^T\theta - y_i\right) + 2\lambda\,(\theta - a)$$

$$= 2\sum_{i=1}^{N} \varphi(x_i)\varphi(x_i)^T\,\theta - 2\sum_{i=1}^{N}\varphi(x_i)\,y_i + 2\lambda\,(\theta - a) = 0$$

$$\Rightarrow 2\left[\sum_{i=1}^{N} \varphi(x_i)\,\varphi(x_i)^T + \lambda I\right]\theta = 2\sum_{i=1}^{N}\varphi(x_i)\,y_i + 2\lambda a$$

$$\Rightarrow \theta^* = \left[\sum_{i=1}^{N} \varphi(x_i)\varphi(x_i)^T + \lambda I\right]^{-1}\left(\sum_{i=1}^{N}\varphi(x_i)y_i + \lambda a\right)$$

2) Robust Regression using Huber Loss: In the class, we defined the Huber loss as

10 points

$$\ell_\delta(e) = \begin{cases} \frac{1}{2}e^2 & \text{if } |e| \leq \delta \\ \delta|e| - \frac{1}{2}\delta^2 & \text{if } |e| > \delta \end{cases}$$

Consider the robust regression model

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell_\delta(y_i - \boldsymbol{\theta}^\top \phi(\boldsymbol{x}_i)),$$

where $\phi(\boldsymbol{x}_i)$ and y_i denote the i-th input sample and output/response, respectively and $\boldsymbol{\theta}$ is the unknown parameter vector.

a) Provide the steps of the batch gradient descent in order to obtain the solution for $\boldsymbol{\theta}$.

b) Provide the steps of the stochastic gradient descent using mini-batches of size 1, i.e., one sample in each mini-batch, in order to obtain the solution for $\boldsymbol{\theta}$.

Solution: The gradient of Huber loss Function is $\frac{\partial l_\delta}{\partial e} = \begin{cases} -\delta & ; \text{ if } e<-\delta \\ e & ; \text{ if } -\delta<e<\delta \\ +\delta & ; \text{ if } e>\delta \end{cases}$.

Notice that the gradient is continuous at $e=-\delta$ and $e=+\delta$. Let $e_i \triangleq y_i - \theta^T x_i$. Then

$$\frac{\partial l}{\partial \theta}(y_i - \theta^T x_i) = \frac{\partial l}{\partial e_i} \times \frac{\partial e_i}{\partial \theta} = \frac{\partial l}{\partial e_i} \times (-x_i) .$$

Thus, for $J(\theta) = \sum_{i=1}^{N} l_\delta(y_i - \theta^T x_i)$, we have $\frac{\partial J}{\partial \theta} = -\sum_{i=1}^{N} x_i \frac{\partial l_\delta(e_i)}{\partial e_i}$. δ

* Batch GD: (1) Initialize $\theta^{(0)}$

(2) Repeat until convergence or reaching maximum # iterations

$$\theta^{(t+1)} \longleftarrow \theta^{(t)} - f \left.\frac{\partial J}{\partial \theta}\right|_{\theta^{(t)}} .$$

* Stochastic GD: (1) Initialize $\theta^{(0)}$

(2) Repeat until convergence or reaching maximum # iterations

(2-1) Draw a sample i from $\{1,2,\dots,N\}$ at random

(2-2) Calculate the gradient of $l_\delta(y_i - \theta^T x_i)$ and let $e_i \triangleq y_i - \theta^{(t)T} x_i$.

(2-3) Update $\theta^{(t+1)} \leftarrow \theta^{(t)} - f \frac{\partial l_\delta(e_i)}{\partial e_i} \times (-x_i)$.

3) Probability and Random Variables: State true or false. If true, prove it. If false, either prove or demonstrate by a counter example. Here Ω denotes the sample space and A^c denotes the complement of the event A. X and Y denote random variables.

16 points

4 points each part

1. For any $A, B \subseteq \Omega$ such that $0 < P(A) < 1$, $P(A|B) + P(A|B^c) = 1$.
2. For any $A, B \subseteq \Omega$, $P(B^c \cap (A \cup B)) + P(A^c \cup B) = 1$.
3. $P(A_1, \dots, A_n) = P(A_1)P(A_2|A_1)P(A_3|A_1, A_2) \times \cdots \times P(A_n|A_1, \dots, A_{n-1})$.
4. If X and Y are independent discrete random variables, then $E[XY] = E[X]E[Y]$, where $E[\cdot]$ denotes expectation.

(1) False. Let $A=B$ and $P(A)=0.1$.

We have $P(A|B)+P(A|B^c) = \underbrace{P(A|A)}_{=P(A)} + \underbrace{P(A|A^c)}_{=0} = P(A) = 0.1 \neq 1$.

(2) True. Notice that $B^c \cap (A\cup B) = (B^c\cap A) \cup \underbrace{(B^c\cap B)}_{\phi \text{ emptyset}} = (B^c\cap A)\cup \phi = B^c\cap A$.

On the other hand, $(B^c\cap A)^c = B\cup A^c$, hence we have: $\underbrace{P(B^c\cap(A\cup B))}_{=P(B^c\cap A)} + \underbrace{P(A^c\cup B)}_{=P((B^c\cap A)^c)} = 1$.

(3) True. $P(A_1)\,P(A_2|A_1) = P(A_1)\,\dfrac{P(A_2,A_1)}{P(A_1)} = P(A_1,A_2)$.

$P(A_1)\,P(A_2|A_1)\,P(A_3|A_1,A_2) \underset{\text{from above}}{=} P(A_1,A_2)\times P(A_3|A_1,A_2) = P(A_1,A_2)\times\dfrac{P(A_1,A_2,A_3)}{P(A_1,A_2)} = P(A_1,A_2,A_3)$.

$\vdots$

$P(A_1)\,P(A_2|A_1)\times\cdots\times P(A_n|A_1,\dots,A_{n-1}) = P(A_1,A_2,\dots,A_n)$.

(4) True. $E[XY] = \sum_x\sum_y xy\,p(x,y) = \sum_x\sum_y xy\,p(x)\,p(y)$ $\quad\left(X \perp\!\!\!\perp Y \text{ indep.} \iff p(x,y)=p(x)\,p(y) \right)$

$= \sum_x \left(x\,p(x) \sum_y y\,p(y)\right) = \underbrace{\sum_x x\,p(x)}_{=E[X]} \times \underbrace{\sum_y y\,p(y)}_{=E[Y]} = E[X]\cdot E[Y]$.

4) Maximum Likelihood Estimation: Assume $X_1, X_2, \dots, X_N$ are i.i.d. random variables each taking a real value, where

$$p_\delta(X_i = x_i) = e^{-(\delta^2+\delta x_i)}.$$

δ is the parameter of the distribution. Assume, we observe $X_1 = x_1, X_2 = x_2, \dots, X_N = x_N$.

a) Write down the likelihood function $L(\delta)$.
b) Derive the maximum likelihood or log-likelihood estimation of δ for the given observations. Provide all steps of derivations.

10 points

5 points each part

Solution: a) $L(\delta) = P_\delta(X_1=x_1,\dots,X_N=x_N) \overset{iid}{=} \prod_{i=1}^N P_\delta(X_i=x_i) = \prod_{i=1}^N e^{-(\delta^2+\delta x_i)} = e^{-\sum_{i=1}^N(\delta^2+\delta x_i)}$

$= e^{-(N\delta^2+\delta\sum_i x_i)}$

b) To find MLE, first compute log-likelihood: $\ell(\delta) = \log L(\delta) = -(N\delta^2+\delta\sum_{i=1}^N x_i)$.

Then set $\dfrac{\partial \ell(\delta)}{\partial\delta} = 0 \longrightarrow \dfrac{\partial\ell}{\partial\delta} = -2N\delta + \sum_{i=1}^N x_i = 0 \longrightarrow \hat{\delta}_{MLE} = \sum_{i=1}^N x_i / 2N$.

5) Logistic Regression: In the logistic regression for binary classification ($y \in \{0, 1\}$), we defined $p(y = 1|x) = \sigma(\boldsymbol{w}^\top \boldsymbol{x})$, where the sigmoid function is defined as

10 points

5 points each part

$$\sigma(z) \triangleq \frac{1}{1 + e^{-z}}.$$

Assume we have trained the logistic regression model using a given dataset and have learned $\boldsymbol{w}$. Let $\boldsymbol{x}_n$ be a test sample.

1. Assume $\boldsymbol{w}^\top \boldsymbol{x}_n < 0.3$. To which class $\boldsymbol{x}_n$ belongs? Provide details of your derivations.
2. Assume $\frac{1}{1+e^{\boldsymbol{w}^\top \boldsymbol{x}_n}} = 0.7$. To which class $\boldsymbol{x}_n$ belongs and with what probability? Provide details of your derivations.

Solution : 1. We know that when $\omega^T x_n > 0$, x_n belongs to class 1 ,
$\omega^T x_n < 0$, x_n belongs to class 0 .

Since $\omega^T x_n < 0.3$, it is possible that $\begin{cases} 0 < \omega^T x_n < 0.3 \longrightarrow \text{hence, } x_n \text{ belongs to class } 1 \\ \omega^T x_n < 0 \longrightarrow \text{hence, } x_n \text{ belongs to class } 0 \end{cases}$

Thus, x_n could be in either class 0 or 1.

However, given the logistic model $p(y=1|x) = \frac{1}{1+e^{-\omega^T x}}$, when $\omega^T x < 0.3$, we can say that the probability of x^n belonging to class 1 is always less than or equal to $\frac{1}{1+e^{-0.3}}$.

2 . $$\frac{1}{1+e^{\omega^T x_n}} = \frac{e^{-\omega^T x_n}}{e^{-\omega^T x_n}} \times \frac{1}{1+e^{\omega^T x_n}} = \frac{e^{-\omega^T x_n}}{1+e^{-\omega^T x_n}} = \frac{1+e^{-\omega^T x_n}-1}{1+e^{-\omega^T x_n}}$$

$$= 1 - \frac{1}{1+e^{-\omega^T x_n}} = 1 - p(y=1|x_n) = p(y=0|x_n)$$

$\Rightarrow$ $p(y_n=0|x_n) = 0.7$, Thus, x_n belongs to class 0 (since $0.7 > 0.5$) or more precisely,

x_n belongs to class 0 with probability 0.7 . 5

17 points

6) Gradient Descent. Consider the function $f(x_1, x_2) = (-2x_1^2 - 3x_1x_2 + 2x_2^2) \times \sin(x_1)$. Write down the expression of the gradient of f. Use the python code you wrote in HW1. The code, similar to before, gets as input i) an initial point (x_1^0, x_2^0), ii) the maximum number of iterations of the gradient descent, iii) learning rate ρ. The algorithm terminates when the stopping criteria is met. The stopping criteria in this case are: when convergence happens ($\epsilon = 0.01$) or the maximum number of iterations (10,000) is achieved or when $|x_1| > 6$ or when $|x_2| > 6$. The output of the code in this case must be the last point (obtained when GD terminates) as well as the function value at the last point.

1. Run the GD using the following initial points: $(x_1, x_2) = (-3, -4)$, $(x_1, x_2) = (4, -3)$, $(x_1, x_2) = (1, 5)$, $(x_1, x_2) = (-4, -3)$, $(x_1, x_2) = (-5, 5)$. In all these cases, choose ρ sufficiently small so that GD does not diverge. In a 2D plot show the final 5 points obtained by GD when starting from the 5 initializations. Are the finals points the same?

2. In a table, report the function value at each of the 5 final points and report the best minimum value.

3. Visualize the function in 3D (i.e., plot $(x_1, x_2, f(x_1, x_2))$ for $x_1 \in [-6, 6]$ and $x_2 \in [-6, 6]$. Based on the this plot explain why the 5 final points are or are not the same.

7) Linear Regression Implementation.

25 points

a) Write a code in Python whose input is a training dataset $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{N_1}, y_{N_1})\}$, a validation dataset $\{(\boldsymbol{x}'_1, y'_1), \ldots, (\boldsymbol{x}'_{N_2}, y'_{N_2})\}$, a test dataset $\{(\boldsymbol{x}''_1, y''_1), \ldots, (\boldsymbol{x}''_{N_3}, y''_{N_3})\}$ as well as a variable called s, which specifies the type of the solver: when $s = 0$ it uses closed-form implementation and when $s = 1$ it uses gradient descent. In other words, you need to implement both closed-form and gradient descent in your code. The outputs are the optimal weight vector $\boldsymbol{\theta}$ in the linear regression model $y = \boldsymbol{\theta}^\top \phi(\boldsymbol{x})$, for a given nonlinear mapping $\phi(\cdot)$, as well as the regression errors (i.e., the difference between the true response and the model prediction) on the training set, validation set and the test set for the optimal parameter.

b) Consider a basis function expansion of the form of n-degree polynomials, i.e., for each sample $\boldsymbol{x}_i = [x_{1i}, \ldots, x_{di}]^\top$ with d features, we have

$$\phi(\boldsymbol{x}_i) = \begin{bmatrix} x_{1i} & x_{1i}^2 & \cdots & x_{1i}^n & \cdots & x_{di} & x_{di}^2 & \cdots & x_{di}^n & 1 \end{bmatrix}^\top.$$

Download the dataset on the course piazza webpage. Notice the dataset is in matlab format (.mat); you can use scipy.io.savemat to convert it to python format (.npy). Also, in this dataset $d = 1$, i.e, there is only one feature for each sample. Run the code on the training data to compute $\boldsymbol{\theta}$ for $n \in \{1, 2, 3, \ldots, 9\}$. Plot the training and testing regression errors as a function of n for closed-form approach and gradient descent separately. How does each error change as a function of n? For each $n \in \{1, 2, 3, \ldots, 9\}$, plot the training data and the learned function together.

c) Write a second code in Python (i.e., modify the above code) that applies Ridge regression to the same dataset to compute $\boldsymbol{\theta}$, training, validation and testing errors for a given λ, using a closed-form solution and a gradient descent method. Set $n = 7$. Use the validation dataset and plot the validation error as a function of the regularization parameter $\lambda \in \{0.001, 0.01, 0.1, 1, 10, 100\}$. Based on the results, report the best λ. Use the optimal $\boldsymbol{\theta}$ associated with this best λ to compute the regression error on test samples and report the result.

Homework Submission Instructions: Please submit the analytical part of the homework in class (put a paper copy of the HW on the instructor's desk before the class starts). Handwritten or printed documents generated by Latex/Word are both acceptable. Please submit all your plots and your Python code (.py file) via email, by the DEADLINE. To submit, please send an email to the instructor and cc the TAs.

– The title of your email must be "DS4400: HW02:YourLastName".

– Please attach a single zip file to your email that contains all python codes and plots and a readme file on how to run your files.

– Please name your zip file as "HW02-YourLastName".