PRACTICE MIDTERM EXAM SOLUTION

CSC311 Fall 2019

University of Toronto

1. kNN.

(a) When do we expect k-NN to be better than logistic regression?
Solution
We can expect k-NN to do better than logistic regression when the data is not linearly separable

(b) Describe a sensible method for setting k in a k-nearest neighbor classifier.
Solution
We can tune the value of k by comparing the classification accuracy on the a validation set

(c) Contrast the decision boundaries for logistic regression and kNN.
Solution
The decision boundary for logistic regression is linear (straight line), while kNN can be non-linear.

2. Entropy and Information Gain. Recall the definitions of information gain and entropy:

$$\begin{aligned} Entropy(C) &\equiv H(C) = \sum_c -P(C=c)\log_2 P(C=c) \\ Gain(C,A) &= H(C) - \sum_{v \in Values(A)} P(A=v)H(C|A=v) \end{aligned}$$

(a) Suppose that in a set of examples there are two classes, with 150 examples in the + class and 50 examples in the − class. What is the entropy of the class variable (you can leave this in terms of logs)?
Solution

$$\begin{aligned} H(C) &= -\frac{150}{150+50}\log\frac{150}{150+50} - \frac{50}{150+50}\log\frac{50}{150+50} \\ &= -\frac{3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4} \end{aligned}$$

(b) For this data, suppose the *Color* attribute takes on one of 3 values (red, green, and blue), and the split into the two classes across *red*/*green*/*blue* is + : (120/10/20) and − : (0/10/40). Write down an expression for the class entropy in the subset containing all *green* examples. Is this entropy greater or less than the entropy in the previous question?
Solution

$$H(C|Color = green) = -\frac{10}{10+10}\log\frac{10}{10+10} - \frac{10}{10+10}\log\frac{10}{10+10}$$

$$= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2}$$

(c) Is *Color* a good attribute to add to the tree? Explain your answer.
Solution
Yes – the other two attribute values have almost no entropy

(d) What is the information gain for a particular attribute if every value of the attribute has the same ratio between the number of + examples and the total number of examples?
Solution Conditional entropies all the same, so must be the same as $H(C)$, so gain is 0.

3. Linear Classifiers.

3.1. *Logistic regression.*

- **Note:** This question is harder than any problem that you will get on the exam, and you wil not be expected to do such a detailed derivation.

In class, we encoded the target values for logistic regression with $t^{(i)} \in \{0, +1\}$. In this problem, you will derive an equal formulation when targets are encoded with $\tilde{t}^{(i)}\{-1, +1\}$.

For a dataset $\mathcal{D}_N = \{(\mathbf{x}^{(i)}, t^{(i)})\}$ with $t^{(i)} \in \{0, +1\}$, logistic regression is defined using the following steps:

$$z = \mathbf{w}^\top \mathbf{x} + b$$
$$y = \sigma(z)$$
$$\mathcal{L}(y, z) = -t\log(y) - (1-t)\log(1-y).$$

(a) Write the equivalent cost minimization problem over training data by eliminating the intermediate variables y and z. Your cost function should only depend on variables $\mathbf{w}$ and b, and dataset $\mathcal{D}$.
Solution

We can substitute the expression for y, then later substitute for z:

$$\begin{aligned}
\mathcal{L}(z, \mathcal{D}) &= -t \log(\sigma(z)) - (1-t) \log((1-\sigma(z)) \\
&= -t \log\left(\frac{1}{1+\exp\{-(z)\}}\right) - (1-t) \log\left(1 - \frac{1}{1+\exp\{-(z)\}}\right) \\
&= -t \log\left(\frac{1}{1+\exp\{-(z)\}}\right) - (1-t) \log\left(\frac{\exp\{-(z)\}}{1+\exp\{-(z)\}}\right) \\
&= -t \log\left(\frac{1}{1+\exp\{-(z)\}}\right) - (1-t) \log\left(\frac{1}{1+\exp\{(z)\}}\right) \\
&= t \log\Big(1+\exp\{-(z)\}\Big) + (1-t) \log\Big(1+\exp\{(z)\}\Big) \\
\mathcal{L}(\mathbf{w}, b, \mathcal{D}) &= t \log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x} + b)\}\Big) + (1-t) \log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x} + b)\}\Big) \\
&= \sum_{i=1}^{N} t^{(i)} \log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big) + (1-t^{(i)}) \log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)
\end{aligned}$$

Thus the cost minimization problem is formulated as:

$$\text{minimize}_{\mathbf{w},b} \sum_{i=1}^{N} t^{(i)} \log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big) + (1-t^{(i)}) \log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)$$

(b) Show that if $\tilde{t}^{(i)} \in \{-1, +1\}$, the minimization problem takes the following form.

$$\text{minimize}_{\mathbf{w},b} \sum_{i=1}^{N} \log\Big(1+\exp\{-\tilde{t}^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)$$

Solution

We can substitute $t^{(i)} = \frac{\tilde{t}^{(i)}+1}{2}$ into the expression above:

$$\begin{aligned}
\mathcal{L}(\mathbf{w}, b, \mathcal{D}) &= \sum_{i=1}^{N} \frac{\tilde{t}^{(i)}+1}{2} \log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big) + \left(1 - \frac{\tilde{t}^{(i)}+1}{2}\right) \log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big) \\
&= \sum_{i=1}^{N} \frac{\tilde{t}^{(i)}+1}{2} \log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big) + \frac{1-\tilde{t}^{(i)}}{2} \log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)
\end{aligned}$$

When $\tilde{t}^{(i)} = +1$ for the i-th example, the second term disappears ($\frac{1-\tilde{t}^{(i)}}{2} = 0$), leading to the remaining term $\log\Big(1+\exp\{-(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)$.

When $\tilde{t}^{(i)} = -1$ for the i-th example, the first term disappears ($\frac{\tilde{t}^{(i)}+1}{2} = 0$), leading to the remaining term $\log\Big(1+\exp\{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)$.

Therefore, the only difference between the two cases is the sign inside the exponential term, which has the opposite sign as $\tilde{t}^{(i)}$. We can simplify to the desired expression:

$$\mathcal{L}(\mathbf{w}, b, \mathcal{D}) = \sum_{i=1}^{N} \log\Big(1+\exp\{-\tilde{t}^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}\Big)$$

3.2. *Linear decision boundary.* Assume that we trained a logistic regression model and our class probabilities can be found by

$$z(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b)$$

where $(\mathbf{w}_k, w_{k,0})$ are the parameters, and we classify using the rule

$$y(\mathbf{x}) = \mathbb{1}[z(\mathbf{x}) > 0.5].$$

Show that this corresponds to a linear decision boundary in the input space.
Solution
What decision boundary looks like:

- predict $y = 1$ if $z(\mathbf{x}) > 0.5 \iff \mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0 > 0$
- predict $y = 0$ if $z(\mathbf{x}) \leq 0.5 \iff \mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0 \leq 0$
- decision boundary: $\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0 = 0$

$$\begin{aligned}
\frac{1}{1 + e^{-(\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0)}} &= 0.5 \\
1 &= 0.5(1 + e^{-(\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0)}) \\
1 - 0.5 &= 0.5(e^{-(\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0)}) \\
1 &= e^{-(\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0)} \\
ln(1) &= ln(e^{-(\mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0)}) \\
0 &= \mathbf{w}_k^\top \mathbf{x} + w_{k,0} x_0
\end{aligned}$$

General notes on input spaces:

- training examples are points
- hypotheses are half-spaces whose boundaries pass through origin
- the boundary is the decision boundary
- if training examples can be separated by linear decision rule, they are linearly separable

4. Optimization.

4.1. *Minimizing training error - 5pts.* Assume that you are minimizing a cost function which can be written as

$$\mathcal{J}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\mathbf{w}, \mathbf{x}_i, t_i),$$

where $N = 1,000,000$.

(a) Write the one-step update rules for gradient descent (GD), stochastic GD (SGD), and mini-batch SGD (mSGD) with batch size 100. You can denote the gradient of the loss with respect to $\mathbf{w}$ for each sample with $\mathbf{g}_i = \nabla \mathcal{L}(\mathbf{w}, \mathbf{x}_i, t_i)$, and your learning rate with η.
Solution

- GD:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \sum_{i=1}^{N} g_i$$

- SGD:

$$\text{Choose } i \sim \mathcal{U}[1, N], \mathbf{w} \leftarrow \mathbf{w} - \eta g_i$$

- mSGD:

$$\text{Choose a subset } M \subset \{1, \ldots, N\},\ \mathbf{w} \leftarrow \mathbf{w} - \eta \textstyle\sum_{i \in M}^{|M|} g_i$$

(b) Rank the computational cost of each iteration for GD, SGD, and mini-batch SGD (with batch size 100) from smallest the largest.
Solution
From smallest to largest: SGD < mSGD < GD.
SGD only requires processing 1 example; mSGD, 100; GD, 1,000,000.

5. Neural networks.

5.1. *NN-1.* Consider the following learning rule:

$$w_{ji}^{\texttt{new}} = w_{ji}^{\texttt{old}} - \eta \sum_{n} (y_j^{(n)} - t_j^{(n)}) x_i^{(n)}$$

(a) Define each of the five terms on the right-hand side of the learning rule.
Solution

- $w_{ji}^{\texttt{old}}$: The current (i.e. old) weight connecting the i-th hidden unit to the j-th output unit.
- η: The learning rate (or step size).
- $y_j^{(n)}$: Network's output for the j-th class for the n-th example (between 0 to 1).
- $t_j^{(n)}$: Target value for the j-th class for the n-th example (0 or 1).
- $x_i^{(n)}$: The n-th example's input for dimension i.

(b) Imagine that another term is added, producing this new learning rule:

$$w_{ji}^{\texttt{new}} = w_{ji}^{\texttt{old}} - \eta \sum_{n} (y_j^{(n)} - t_j^{(n)}) x_i^{(n)} - 2\alpha w_{ji}^{\texttt{old}}$$

What is the main aim of such a term? What effect does this term have on the network weights?
Solution
The added term is for *regularization*, resulting in *weight decay*, i.e. causing the (L2) norm of the network's weights to not be too large, in order to prevent overfitting to the training data.

5.2. *NN-3.* The "flexibility" of a neural network, it's ability to model different functions, is given by the number of hidden units. If we wanted to, we could simply use millions (i.e., a lot) of hidden units in order to model any kind of function we wanted. Why is this a bad idea in general? How could we avoid this problem?

Solution:
The problem is overfitting–i.e. the model will do well on the training data but not generalize to examples it hasn't seen before.
Some ways to avoid this problem: cross-validation to tune number of hidden units, regularization/weight decay, ensembles.

6. True or False questions. Circle either True or False. Each correct answer is worth 2 points. To discourage random guessing, 2 points will be deducted for a wrong answer.

1. (True or False) Assume that you are using cross validation to choose the penalty parameter λ in L^2 regularized linear regression. As the number of training samples increases, we expect that the value of λ chosen by cross validation becomes larger.
   Solution: False: As the number of training samples increase, the model will be overfitting the training data less. Therefore we expect that the model requires less regularization, i.e. the value of λ chosen by cross validation should become *smaller*.
2. (True or False) In the K-fold cross-validation procedure for selecting a model parameter λ out of m values, you fit your model $K \times m$ times.
   Solution: True
3. (True or False) Assume that you have a dataset composed of N observations: the target $\mathbf{t}$ and features $\mathbf{X}$. You want to fit a linear regression model and find the weights $\mathbf{w}$, but you also know that more data is always helpful. Instead of fitting a model with $\mathbf{t} \in \mathbb{R}^n$ and $\mathbf{X} \in \mathbb{R}^{n \times d}$, you concatenate the data and fit a model using $\begin{bmatrix} \mathbf{t} \\ \mathbf{t} \end{bmatrix} \in \mathbb{R}^{2n}$ and $\begin{bmatrix} \mathbf{X} \\ \mathbf{X} \end{bmatrix} \in \mathbb{R}^{2n \times d}$. Running linear regression on this new dataset will give the same weights as on the original dataset.
   Solution: True
4. (True or False) The decision boundaries resulting from linear regression with 1-of-K encoded targets are always the same those resulting from logistic regression.
   Solution: True
5. (True or False) We use stochastic gradient descent (SGD) with very small constant step size to minimize a loss function. Assuming that we can run SGD for a very long time, eventually it will converge to a minimum of the loss function.
   Solution: True.