CS181 2020 Midterm 1 Practice Questions

1. **Linear Regression**

Consider a one-dimensional regression problem with training data $\{x_i, y_i\}$. We seek to fit a linear model with no bias term:

$$\hat{y} = wx$$

a. Assume a squared loss $\frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ and solve for the optimal value of w^*.

b. What is the prediction for some new observation x, without mention of w?

c. Suppose that we have a generative model of the form $\hat{y} = wx + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and w is known. Given a new x, what is the expression for the probability of $\hat{y}$?
Note: The univariate Gaussian PDF is:

$$\mathcal{N}(a|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-(a-\mu)^2}{2\sigma^2}\right)$$

d. Now assume that w is random and that we have a prior on w with known variance s_0^2:

$$w \sim \mathcal{N}(0, s_0^2)$$

Write down the form of the *posterior* distribution over w. Take logs and drop terms that don't depend on the data and prior parameters, but you do not need to simplify further (i.e. you do not need to complete the square to make it look like a normal).

Solution:

2. **Regularization**

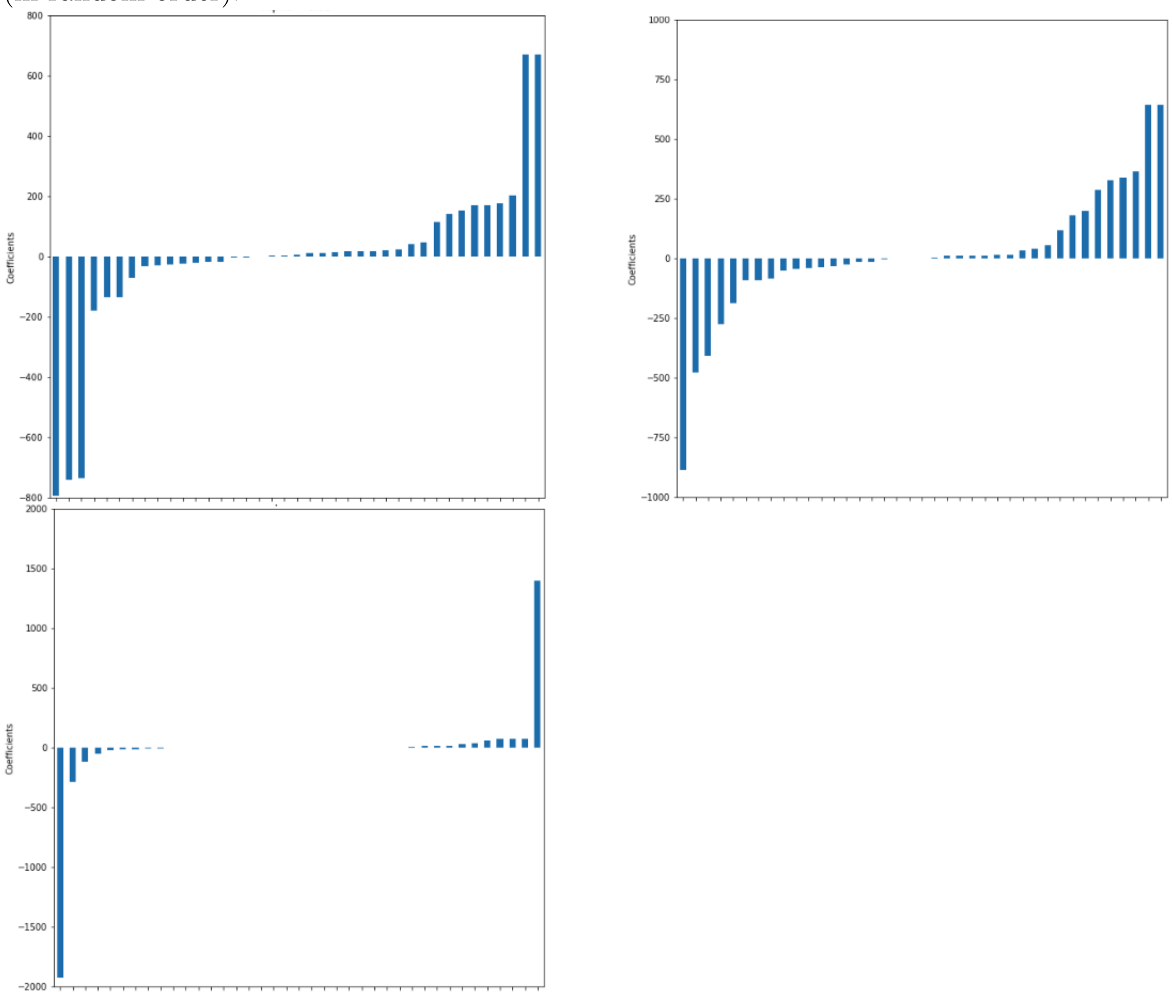
Suppose we wish to predict sales according to given characteristics of a sold item and its sales outlet using linear regression. Consider a linear regression model $y = w^T x$. We try three different loss functions on our data set:

(a) No regularization: $L(w) = \frac{1}{2}\sum_{n=1}^{N}(y_n - w^T x_n)^2$

(b) LASSO regression: $L(w) = \frac{1}{2}\sum_{n=1}^{N}(y_n - w^T x_n)^2 + \frac{\lambda}{2}||w||_1$

(c) Ridge regression: $L(w) = \frac{1}{2}\sum_{n=1}^{N}(y_n - w^T x_n)^2 + \frac{\lambda}{2}||w||_2^2$

Let $\lambda = 0.05$. We train our linear regression model for each loss function, which gives us different final weights, or variable coefficients. These coefficients are shown in the plots below (in random order):



Now answer the following questions:

a. Which plot corresponds to which loss function? Why?

b. How can we expect the plots to change as we increase λ?

Solution:

3. **Linear Basis Functions**

Linear basis functions $\phi(x)$ are often important in both regression and classification tasks.

$$h(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\phi}(x)$$

Without them, linear and logistic regression can only fit linear functions to the data. The following question asks you to determine if a class of basis function can linearly separate the data $\mathcal{D} = \{(x, y)\} = \{(-\pi, 1), (0, -1), (\pi, 1)\}$. If so, find a setting of $\boldsymbol{w}$ that correctly classifies the data-points (assuming a logistic regression setup).

a. $\boldsymbol{\phi}(x) = [1, x]^T$
b. $\boldsymbol{\phi}(x) = [1, x, x^2]^T$
c. $\boldsymbol{\phi}(x) = [1, x, x^4]^T$
d. $\boldsymbol{\phi}(x) = [1, \cos x]^T$

Solution:

4. **Probabilistic Linear Regression**

In class, we derived the optimal w^* to maximize the likelihood of training data given normally distributed noise. In this problem, you will explore an alternative distribution on the noise of labels y.

Assume 1-dimensional data x, and that

$$\epsilon \sim Lap(0, 1)$$
$$y|x, \epsilon = w^T x + \epsilon$$

where ϵ is a Laplace random variable. The probability density function for a $Lap(\mu, b)$ random variable is given by

$$p(x) = \frac{1}{2b} \exp(-\frac{|x - \mu|}{b})$$

You can also take as given that when you linearly transform any Laplace random variable by a constant, the distribution of the new transformed variable is still Laplace with a linearly transformed mean. For example, if some random variable $a \sim Lap(0, c)$, then for any constant b, $a + b \sim Lap(0 + b, c)$.

a. What is the distribution of random variable $(y|x)$?

b. Given data $\{(x_i, y_i)\}_{i=1}^N$, write down an expression for the likelihood of observing the data in terms of unknown parameter w.

c. Write down an expression for the negative log likelihood of the data.

d. Recall from section 2.6.2 of the CS 181 textbook that for probabilistic regression with normally distributed noise, minimizing our likelihood function was equivalent to minimizing L2 loss $L(y, \hat{y})$.

   Minimizing your expression from part (c) for Laplacian noise is equivalent to minimizing what kind of loss function $L(y, \hat{y})$?

e. Given that $\frac{d}{da}|a| = sign(a)$, where $sign(a) = 1$ when $a \geq 0$, $sign(a) = -1$ when $a < 0$, take the gradient of the negative log likelihood with respect to w. You can leave your expression in terms of the $sign()$ operator.

   Does this model class seem more or less sensitive to outliers than probabilistic regression with normally distributed noise? Why?

   Note: You won't be expected to solve for the optimal w^* in an expression with $sign()$ operators on the exam.

Solution:

5. **Bayesian Linear Regression**

Consider the following setup. Let $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \mathbb{R}$. Consider the model:

$$y_i \sim \mathcal{N}(\mathbf{w}^T\mathbf{x}, \sigma^2)$$

The likelihood will then be:

$$P(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}) = \prod_{i=1}^{|D|} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \mathbf{w}^T\mathbf{X}_i)^2}{2\sigma^2}\right)$$

Apply a conjugate Gaussian prior, specifically one where each weight is i.i.d.:

$$P(\mathbf{w}) = \mathcal{N}(0, \sigma_0^2\mathbf{I}) = \prod_{j=1}^{|\mathbf{w}|} \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(-\frac{w_j^2}{2\sigma_0^2}\right)$$

a. Find the MAP estimate for the weights as a simplified argmax or argmin expression in non-matrix form. You should NOT end up deriving the full posterior or finding a closed form solution for the MAP. (Hint: recall $\mathbf{w}_{MAP} = \arg\max_{\mathbf{w}} P(\mathbf{w}|D)$)

b. What does the expression that you derived in part 1 remind you of?

c. What happens to the posterior with wider (larger σ_0^2) or narrower (smaller σ_0^2) prior? In particular, how it will affect both the mean and the variance of the posterior. You may want to make a connection based on the results in part b.

d. The prior used here is Gaussian, which has a PDF of the form:

$$P(\mathbf{w}) = \prod_{j=1}^{|\mathbf{w}|} \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(-\frac{w_j^2}{2\sigma_0^2}\right) \propto \prod_j \exp(-w_j^2)$$

Another popular prior uses a modification of the Laplace distribution, which can be loosely thought of as a symmetric exponential distribution. The PDF of this distribution is:

$$P(\mathbf{w}) = \prod_{j=1}^{|\mathbf{w}|} \frac{\lambda}{2\sigma} \exp\left(\frac{-\lambda|w_j|}{\sigma}\right) \propto \prod_j \exp(-|w_j|)$$

How do you expect the result in part 1 to be different with a Laplacian prior instead of a Gaussian prior? How do you expect the connection in part 2 to change? Answer this conceptually without any math.

Solution:

6. **Multiclass Classification**

Suppose that we have a K-class classification scenario with training data $\{x_i, \mathbf{y}_i\}_{i=1}^n$, where the $\mathbf{y}_i$ are 1-hot column vectors.

We model this problem using a neural network with d units in a single hidden layer, expressed as a column vector $\boldsymbol{\phi}(\mathbf{x}; \mathbf{W}, \mathbf{w}_0) \in \mathbb{R}^d$, which we write as $\boldsymbol{\phi}$. We take a linear combination of these values and pass them to a softmax function to get a final set of K outputs. Let $\mathcal{C}_k$ represent a 1-hot vector with a 1 in the k^{th} index and let $\mathbf{v}_\ell \in \mathbb{R}^d$ be a column vector of weights:

$$p(\mathbf{y} = \mathcal{C}_k | \mathbf{x}; \{\mathbf{v}_\ell\}_{\ell=1}^K, \mathbf{W}, \mathbf{w}_0) = \frac{\exp(\mathbf{v}_k^\top \boldsymbol{\phi})}{\sum_{\ell'=1}^K \exp(\mathbf{v}_{\ell'}^\top \boldsymbol{\phi})}$$

a. Suppose we add the same global bias to each vector of weights in the final layer, i.e. replace $\mathbf{v}_k^\top \boldsymbol{\phi}$ with $\mathbf{v}_k^\top \boldsymbol{\phi} + v_0$ for some scalar v_0 with the same scalar for all k. Does this increase the expressivity of our model? Why or why not?

b. Write down and simplify the log likelihood of a particular observation $(\mathbf{x}_i, \mathbf{y}_i)$, including constants. Assume that we use a sigmoid activation function (don't need to simplify within the sigmoid, just the sums/logs/exps around it)

$$\boldsymbol{\phi}(\mathbf{x}; \mathbf{W}, \mathbf{w}_0) = \boldsymbol{\sigma}(\mathbf{W}\mathbf{x} + \mathbf{w}_0)$$

c. Consider the scenario of drawing items from a distribution and encoding them in binary for communication. An efficient scheme encodes common items with a short code and rare items with longer codes. The *cross-entropy* $\mathbb{E}_{p(x)}[-\ln q(x)]$ can be interpreted as the expected number of required bits to send a randomly chosen item $x \sim p(x)$ using a code optimized for $q(x)$. For classification, we can use the following as a loss:

$$\mathbb{E}_{p(y|x)}[-\ln q(y|x)]$$

where $p(\mathbf{y}|\mathbf{x})$ is 1 for the true class and is 0 otherwise and $q(\mathbf{y} = \mathcal{C}_k|\mathbf{x})$ is your model's prediction (output of the softmax layer for k^{th} class). Write down the expression for the cross-entropy by unpacking the expectation and writing it as a sum of terms and describe its relationship to the log loss in part (b).

Solution:

7. **Probabilistic Generative Classification**

Suppose that we use a Naive Bayes classifier to classify binary feature vectors $\mathbf{x} \in \{0, 1\}^D$ into two classes. The class conditional distributions will then be of the form

$$p(\mathbf{x} \,|\, y = C_k) = \prod_{j=1}^{D} \pi_{kj}^{x_j} \,(1 - \pi_{kj})^{(1-x_j)}$$

where $x_j \in \{0, 1\}$, and $\pi_{kj} = p(x_j = 1 \,|\, y = C_k)$. This is a Bernoulli Naive Bayes, different from Multinomial model in the notes in that all the features are binary instead of representing count data. Assume also that the class priors are $p(y = C_1) = p(y = C_2) = \frac{1}{2}$.

a. How is the quantity $\ln(p(y = C_1 \,|\, \mathbf{x})/p(y = C_2 \,|\, \mathbf{x}))$ used for classification of a new example $\mathbf{x}$?
b. If $D = 1$ (i.e., there is only one feature), use the equations above to write out $\ln \frac{p(y=C_1 \,|\, x)}{p(y=C_2 \,|\, x)}$ for a single binary feature x.
c. Now suppose we change our feature representation so that instead of using just a single feature, we use two redundant features. (i.e., two features that always have the same value). With this feature representation, instead of x we will use $\mathbf{x} = [x, x]^\top$. What is $\ln \frac{p(y=C_1 \,|\, \mathbf{x})}{p(y=C_2 \,|\, \mathbf{x})}$ in terms of the value for $\ln \frac{p(y=C_1 \,|\, x)}{p(y=C_2 \,|\, x)}$ you calculated in part (a.)?
d. Is this a bug or a feature?

Solution:

8. **Overfitting and Underfitting**

Harvard Insta-Ice Unit (HI2U) has built a robot that can deliver 24-hour shaved ice to student houses. To prevent collisions, they train three different approaches to classify camera images as containing nearby tourists or open space; if the robot identifies a tourist in its path, it is programmed to halt. The performances of the classifiers are:

	Training Accuracy	Testing Accuracy
Classifier A	75.3%	74.8%
Classifier B	80.3%	77.8%
Classifier C	90.2%	60.0%

where Classifier B has a more expressive model class than A, and classifier C has both a more expressive model class and more features than A. All the classifiers have closed-form solutions, so HI2U is pretty sure that the inference is not hindering performance.

a. If you had to choose one: might Classifier A be overfitting or underfitting? Explain your reasoning.

b. If you had to choose one: might Classifier C be overfitting or underfitting? Explain your reasoning.

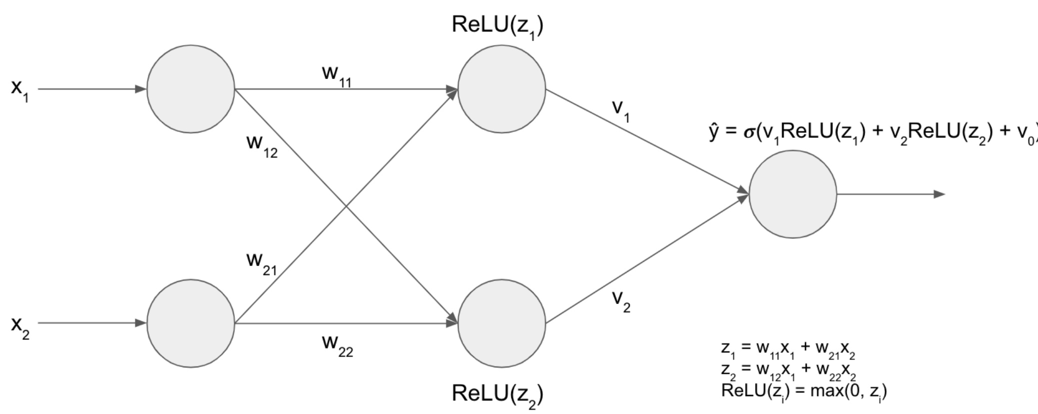
c. If you had to guess yes or no: might more training examples significantly boost the test-time performance of Classifier A? Classifier C? Explain your reasoning.

Hint: try to relate your reasoning to model bias and model variance.

Solution:

9. **Neural Networks**

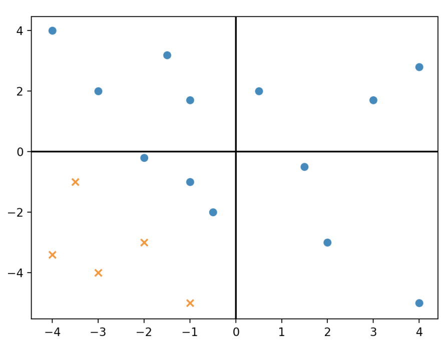
Consider the following 2-layer neural network, which takes in $x \in \mathbb{R}^2$ and has two ReLU hidden units and a final sigmoid activation. Notice there are no bias weights on the hidden units.



For a binary classification problem with true labels $y \in \{0, 1\}$, we will use the loss function $L = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}))$.

a. Suppose we update our neural network with stochastic gradient descent on a data point $x = [x_1 x_2]^T$.
   i. Calculate the gradient of the loss with respect to v_1.
   ii. Calculate the gradient of the loss with respect to w_{11}.

b. Consider the classification of data points below. Is it possible that this classification was generated by the set of weights $w_{11}, w_{12}, w_{21}, w_{22} = \{1, 0, 0, 1\}$? Why or why not? What if additional hidden layers were applied to further transform the data (still keeping the specified set of weights fixed)?



c. i. Why is it a bad idea in general to have ReLU as the activation function of the output layer?
   ii. Suppose we want to classify our outputs into 5 categories. Why might it be a bad idea to use the label set $\{1, 2, 3, 4, 5\}$? What could we use instead?

Solution:

10. **Neural Networks**

Consider the following non-linearity for use in a neural network:

$$f_{0/1}(z) = \left\{ \begin{array}{cc} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{array} \right\}$$

Let $\mathbf{x}$ be a binary feature vector of length 4: $\mathbf{x} \in \{0, 1\}^4$. Define neural network A as follows:

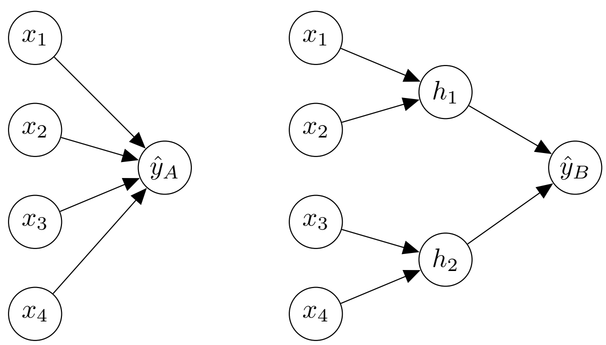
$$\hat{y_A} \leftarrow f_{0/1}(\mathbf{w}^\top \mathbf{x} + w_0)$$

with weight vector $\mathbf{w} \in \mathbb{R}^4$ and bias scalar $w_0 \in \mathbb{R}$.

Let $\mathbf{x}^L = [x_1, x_2]$ and $\mathbf{x}^R = [x_3, x_4]$. Define neural network B as follows:

$$\begin{aligned} h_1 &\leftarrow f_{0/1}(\mathbf{t}^\top \mathbf{x}^L + a) \\ h_2 &\leftarrow f_{0/1}(\mathbf{u}^\top \mathbf{x}^R + b) \\ \mathbf{h} &\leftarrow [h_1, h_2] \\ \hat{y_B} &\leftarrow f_{0/1}(\mathbf{v}^\top \mathbf{h} + c) \end{aligned}$$

with weight vectors $\mathbf{t}, \mathbf{u}, \mathbf{v} \in \mathbb{R}^2$ and bias scalars $a, b, c \in \mathbb{R}$. Basically, B can only look at the two halves of the input separately and has an extra layer to merge the transformations on the two halves of the input with another transformation:



a. i. Describe a logical formula on inputs that can be expressed by A but not by B and provide weights for $\mathbf{w}$ and w_0 that implement the formula in A (hint: think about things you may want to do with binary vectors, e.g. `AND`s, `OR`s)
   ii. Provide an argument for why B cannot express this formula (we don't expect a rigorous proof, but try to give a complete and convincing argument).
   iii. How might you change the architecture of B to fix this issue? What downside might this have?

b. What is the concern about training the networks as currently defined? What change would you make to the network to alleviate this concern?

c. State **two** ways in which a *validation set* can be used when training neural networks (one sentence for each is fine).

Solution: