University of Illinois Spring 2020

CS 446/ECE 449: Machine Learning
Problem Sets

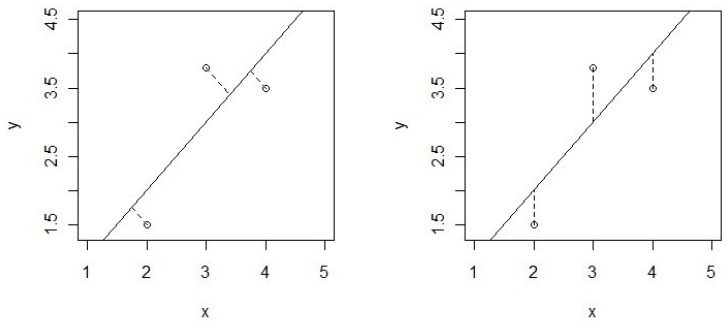
1. **[13 points]** Regression
Suppose we are given a set of observations $\{(x^{(i)}, y^{(i)})\}$ where $x, y \in \mathbb{R}$ and $i \in \{1, 2, \ldots, N\}$. Consider the following program:

$$\underset{w_1, w_2}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y^{(i)} - w_1 \cdot x^{(i)} - w_2 \right)^2. \tag{1}$$

(a) What is the minimum number of observations required for a unique solution?

Solution: 2 distinct observations (1 point)

(b) Which of the following plots shows the correct residual that is minimized with the above program? Circle the correct answer.



Solution: The one on the right. (1 point)

(c) Suppose we now want to fit a quadratic model to the observed data. Modify the program given in Eq. 1 accordingly. Derive the closed form solution for this case. You may assume that you have a sufficient number of data samples. Use matrix vector notation, i.e., $\mathbf{w}$, $\mathbf{X}$, and $\mathbf{Y}$ and define them carefully.

Solution: (5 point)

$$\underset{w_1, w_2, w_3}{\operatorname{argmin}} \sum_{i}^{N} (y^{(i)} - w_1 \cdot (x^{(i)})^2 - w_2 \cdot x^{(i)} - w_3)^2$$

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}, \quad \mathbf{X}^T = \begin{bmatrix} {x^{(1)}}^2 & x^{(1)} & 1 \\ {x^{(2)}}^2 & x^{(2)} & 1 \\ & \vdots & \\ {x^{(N)}}^2 & x^{(N)} & 1 \end{bmatrix}$$

$$\mathbf{w}^* = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{Y}$$

(d) Briefly describe the problem(s) that we encounter if we were to fit a high degree polynomial to a data that is known to be linear.

Solution: (1 points) Overfitting, poor generalization.

(e) The program above (Eq. 1) assumes $x \in \mathbb{R}$. State the program for $\mathbf{x} \in \mathbb{R}^D$ and specify the dimensions of $\mathbf{w}$.

Solution: (2 points)

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - \left\langle \begin{bmatrix} \mathbf{x}^{(i)^T} & 1 \end{bmatrix}, \mathbf{w} \right\rangle \right)^2, \qquad \mathbf{w} \in \mathbb{R}^{D+1}$$

(f) If $\dim(\mathbf{x}) = D > N$, how could the program be modified such that a unique solution can be obtained?

Solution: (1 point)

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - \left\langle \begin{bmatrix} \mathbf{x}^{(i)^T} & 1 \end{bmatrix}, \mathbf{w} \right\rangle \right)^2 + \lambda \left\| \mathbf{w} \right\|_2^2$$

(g) Is there a closed form solution to the new program? If so derive it.

Solution: (2 points)

$$\mathbf{w}^* = (\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I})^{-1}\mathbf{X}\mathbf{Y}$$

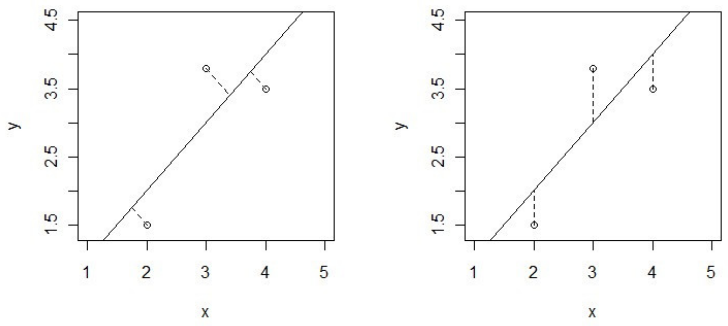
2. **[7 points]** Regression
Suppose we are given a set of observations $\{(x^{(i)}, y^{(i)})\}$ where $x, y \in \mathbb{R}$ and $i \in \{1, 2, \ldots, N\}$. Consider the following program:

$$\underset{w_1, w_2}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y^{(i)} - w_1 \cdot x^{(i)} - w_2 \right)^2. \tag{2}$$

(a) (1 point) What is the minimum number of distinct observations required for a unique solution?

Solution: 2 distinct observations (1 point)

(b) (1 point) Which of the following two plots shows the correct residual that is minimized with the above program? Circle the correct answer.



Solution: The one on the right. (1 point)

(c) (2 points) The program above (Eq. 2) assumes $x \in \mathbb{R}$. State the program for $\mathbf{x} \in \mathbb{R}^d$ and specify the dimensions of $\mathbf{w}$. Assume the feature vector $\mathbf{x} \in \mathbb{R}^d$ includes the 1 for the bias.

Solution: (2 points)

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - \left\langle \mathbf{x}^{(i)^T}, \mathbf{w} \right\rangle \right)^2, \qquad \mathbf{w} \in \mathbb{R}^d$$

(d) (1 point) If $\dim(\mathbf{x}) = d > N$, how could the program be regularized such that a unique solution can be obtained? (Hint: use ℓ_2 regularization)

Solution: (1 point)

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - \left\langle \mathbf{x}^{(i)^T}, \mathbf{w} \right\rangle \right)^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

(e) (2 points) **Derive** a closed form solution w^* for the new regularized program. Use the following notation: $\mathbf{X} \in \mathbb{R}^{d \times N}$ is the input data matrix, $\mathbf{Y} \in \mathbb{R}^N$ is the vector of labels, $\mathbf{I}$ is the identity matrix. Your final expression should be in terms of these provided quantities.

Solution: (2 points)

$$\mathbf{w}^* = (\mathbf{X}\mathbf{X}^T + \frac{\lambda}{2}\mathbf{I})^{-1}\mathbf{X}\mathbf{Y}$$

3. **[15 points]** Softmax Regression

We are given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{|\mathcal{D}|}$ with feature vectors $\mathbf{x}^{(i)} \in \mathbb{R}^d$ and their corresponding labels $y^{(i)} \in \{1, \cdots, K\}$. Here, K denotes the number of classes. The distribution over $y^{(i)}$ is given via

$$p(y^{(i)} = k | \mathbf{x}^{(i)}) = \mu_k(\mathbf{x}^{(i)}) = \frac{e^{\mathbf{w}_k^T \mathbf{x}^{(i)}}}{\sum_{j=1}^K e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}}. \tag{3}$$

Our goal is to learn the weight parameters $\{\mathbf{w}_j\}_{j=1}^K$, with $\mathbf{w}_j \in \mathbb{R}^d$ $(\forall j)$.

(a) Show that the negative conditional log-likelihood $\ell(\mathbf{w}_1, \cdots, \mathbf{w}_K)$ is given by the expression,

$$-\log p(\mathcal{D}) = -\log p(\{y^{(i)}\}_{i=1}^{|\mathcal{D}|} | \{\mathbf{x}^{(i)}\}_{i=1}^{|\mathcal{D}|}) = -\sum_{i=1}^{|\mathcal{D}|}\sum_{k=1}^{K} \mathbb{1}_{\{y^{(i)}=k\}} \mathbf{w}_k^T \mathbf{x}^{(i)} + \sum_{i=1}^{|\mathcal{D}|} \log\Big(\sum_{j=1}^{K} e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}\Big).$$

Here, $\mathbb{1}_{\{y^{(i)}=k\}}$ is an indicator variable, *i.e.*, $\mathbb{1}_{\{y^{(i)}=k\}}$ is equal to 1 if $(y^{(i)} = k)$ and equal to 0 otherwise. *Show intermediate steps and state any used assumptions.*

Solution: (3 points) Under the assumption that the data points are independant and identically distributed, we obtain;

$$\begin{aligned}
-\log p(\mathcal{D}) &= -\log \prod_{i=1}^{|\mathcal{D}|} p(y^{(i)}|\mathbf{x}^{(i)}) = -\log \prod_{i=1}^{|\mathcal{D}|}\prod_{k=1}^{K} \Big(\frac{e^{\mathbf{w}_k^T \mathbf{x}^{(i)}}}{\sum_{j=1}^{K} e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}}\Big)^{\mathbb{1}_{\{y^{(i)}=k\}}} \\
&= -\sum_{i=1}^{|\mathcal{D}|}\sum_{k=1}^{K} \mathbb{1}_{\{y^{(i)}=k\}} \Big(\mathbf{w}_k^T \mathbf{x}^{(i)} - \log\Big(\sum_{j=1}^{K} e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}\Big)\Big) \\
&= -\sum_{i=1}^{|\mathcal{D}|}\sum_{k=1}^{K} \mathbb{1}_{\{y^{(i)}=k\}} \mathbf{w}_k^T \mathbf{x}^{(i)} + \sum_{i=1}^{|\mathcal{D}|} \log\Big(\sum_{j=1}^{K} e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}\Big)
\end{aligned}$$

(b) We want to minimize the negative log-likelihood. To combat overfitting, we add a regularizer to the objective function. The regularized objective is $\ell_r(\mathbf{w}_1, \cdots, \mathbf{w}_K) = \frac{1}{|\mathcal{D}|}\ell(\mathbf{w}_1, \cdots, \mathbf{w}_K) + \lambda \sum_{k=1}^{K} \|\mathbf{w}_k\|^2$. Justify that λ should be a strictly positive scalar, *i.e.*, $\lambda > 0$.

Solution: (2 points) A negative λ would lead to the opposite effect of regularization. The optimal solution for $\mathbf{w}_1, \cdots, \mathbf{w}_K$ is $+\inf$.

(c) Show that the gradient of the regularized loss ℓ_r is

$$\nabla_{\mathbf{w}_k} \ell_r(\mathbf{w}_1, \cdots, \mathbf{w}_K) = 2\lambda \mathbf{w}_k + \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} (\mu_k(\mathbf{x}^{(i)}) - \mathbb{1}_{\{y^{(i)}=k\}})\mathbf{x}^{(i)}.$$

Solution: (4 point)

$$\begin{aligned}
\nabla_{\mathbf{w}_k} \ell_r(\mathbf{w}_1, \cdots, \mathbf{w}_K) &= 2\lambda \mathbf{w}_k - \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \mathbb{1}_{\{y^{(i)}=k\}} \mathbf{x}^{(i)} + \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \frac{e^{\mathbf{w}_k^T \mathbf{x}^{(i)}}}{\sum_{j=1}^{K} e^{\mathbf{w}_j^T \mathbf{x}^{(i)}}} \mathbf{x}^{(i)} \\
&= 2\lambda \mathbf{w}_k + \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} (\mu_k(\mathbf{x}^{(i)}) - \mathbb{1}_{\{y^{(i)}=k\}})\mathbf{x}^{(i)}
\end{aligned}$$

(d) State the gradient update (formula) for **gradient descent** for the regularized loss ℓ_r. Use a learning rate of α.

Solution: (2 points)

$$\mathbf{w}_k^{(t+1)} = \mathbf{w}_k^{(t)} - \alpha\Big(2\lambda\mathbf{w}_k^{(t)} + \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}(\mu_k(\mathbf{x}^{(i)}) - \mathbb{1}_{\{y^{(i)}=k\}})\mathbf{x}^{(i)}\Big) \tag{4}$$

with α being the learning rate.

(e) State the gradient update for **stochastic gradient descent** for the regularized loss ℓ_r. Use a batch size equal to 1 and a learning rate of α.

Solution: (2 points)

$$\mathbf{w}_k^{(t+1)} = \mathbf{w}_k^{(t)} - \alpha\Big(2\lambda\mathbf{w}_k^{(t)} + (\mu_k(\mathbf{x}^{(i)}) - \mathbb{1}_{\{y^{(i)}=k\}})\mathbf{x}^{(i)}\Big) \tag{5}$$

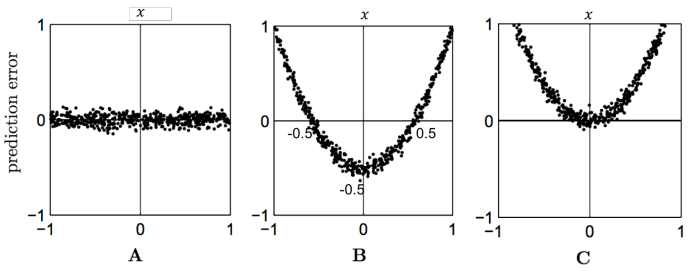
with α being the learning rate.

(f) Consider the case where every instance can be assigned more than one label, for example a movie can be assigned the labels *action* and *comedy* simultaneously. Would you choose to use the classifier from Eq. (3) or K 1-vs.-all logistic regression classifiers?

Solution: (2 points) Use K 1-vs.-all logistic regressions. The classes are not mutually exclusive.

4. **[15 points]** Regression

(a) [3 points] Each plot below claims to represent prediction errors as a function of x for a trained regression model based on some dataset. Some of these plots could potentially be prediction errors for linear ($y = w_1x + w_0$) or quadratic ($y = w_2x^2 + w_1x + w_0$) regression models, while others could not. The regression models are trained with the least squares loss (L2 Loss). Please indicate compatible models and plots.



Solution:

	A	B	C
linear regression	(X)	(X)	( )
quadratic regression	(X)	( )	( )

(b) [4 points] Consider a simple one dimensional logistic regression model:

$$P(y = 1|x, w_1, w_2) = g(w_2 + w_1x), \tag{6}$$

where $g(z) = (1 + \exp(-z))^{-1}$ is the logistic function. Fig. 1 shows two possible conditional distributions $P(y = 1|x, w_1, w_2)$, viewed as a function of x, that we can get by changing the parameters w_1 and w_2. Indicate the number of classification errors for each conditional on the samples A, B and C with coordinates $x_A = -1, x_B = 0$ and $x_C = 1$

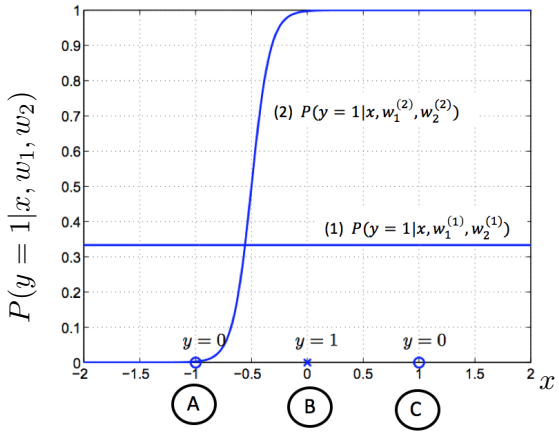


Figure 1: Two possible logistic regression solutions for the three labeled points.

and labels $y_A = 0, y_B = 1$ and $y_C = 0$. Consider a threshold of 0.5, *i.e.*, y is assigned the label 1 if $P(y = 1|x, w_1, w_2) \geq 0.5$.

Solution: Conditional (1) makes (**1**) classification error(s) on sample(s) (**B**). Conditional (2) makes (**1**) classification error(s) on sample(s) (**C**).

(c) [3 points] One of the conditionals in Fig. 1 corresponds to the **maximum likelihood** setting of the parameters w_1 and w_2 based on the labeled data in the figure. Which one is the maximum likelihood solution (1 or 2)?
Hint: Justify by providing $P(y_A, y_B, y_C|x, w_1, w_2)$ for the two conditionals.

Solution: Conditional (1): $P(y_A, y_B, y_C|x, w_1, w_2) = P(y_A|x, w_1, w_2) \cdot P(y_B|x, w_1, w_2) \cdot P(y_C|x, w_1, w_2) = \frac{2}{3} \times \frac{1}{3} \times \frac{2}{3} = \frac{4}{27}$
Conditional (2): $P(y_A, y_B, y_C|x, w_1, w_2) = P(y_A|x, w_1, w_2) \cdot P(y_B|x, w_1, w_2) \cdot P(y_C|x, w_1, w_2) = 1 \times 1 \times 0 = 0$.
Hence, Conditional (1) is obtained using the maximum likelihood estimation.

(d) [2 points] If we additionally tell you that we added a regularization penalty $\frac{1}{2}w_1^2$ to the log-likelihood estimation criterion, would this new information affect your choice of the solution in part (c)? Answer by Yes/No. Justify.
Hint: Check the values of $w_1^{(1)}$ and $w_1^{(2)}$. See Fig. 1 for their definition.

Solution: No. For Conditional (1), we have $w_1^{(1)} = 0$. Hence, the regularization term is also equal to 0. As a result the log likelihood loss with regularization is the same as in part(c). For Conditional (2), $w_1^{(1)} > 0$. Hence, $-\log(P(y_A, y_B, y_C|x, w_1, w_2)) + \frac{1}{2}w_1^2$ is still $+\inf$.

(e) [3 points] In the following, consider an 8-dimensional logistic regression with parameters $w = \{w_i\}_{i=0}^7$ fitted on a dataset $\{(x_k, y_k)\}_{k=1}^N$ and the three objective functions:

$$L_1: \quad \min_w \quad -\sum_{k=1}^{N} \log P(y_k|x_k, w), \tag{7}$$

$$L_2: \quad \min_w \quad -\sum_{k=1}^{N} \log P(y_k|x_k, w) + \lambda \sum_{j=0}^{7} w_j^2, \tag{8}$$

$$L_3: \quad \min_w \quad -\sum_{k=1}^{N} \log P(y_k|x_k, w) + \lambda \sum_{j=0}^{7} |w_j|, \tag{9}$$

where $\lambda \in \mathbb{R}$ and $\lambda > 0$. The following table contains the weights learned for all three objective functions (not in any particular order).

	Column A	Column B	Column C
w_0	0.61	0.23	0.01
w_1	0.84	0.12	0.00
w_2	1.4	0.12	0.00
w_3	0.32	0.09	0.28
w_4	0.74	0.34	0.01
w_5	0.93	0.52	0.00
w_6	0.58	0.16	0.13
w_7	0.24	0.14	0.00

Beside each objective write the appropriate column label (A, B, or C).

Solution: Objective $L_1 \rightarrow$ Column (A)
Objective $L_2 \rightarrow$ Column (B)
Objective $L_3 \rightarrow$ Column (C)

5. **[9 points]** Regression

(a) Consider the following dataset $\mathcal{D}$ in the one-dimensional space.

i	$x^{(i)}$	$y^{(i)}$
1	0	-1
2	1	2
3	1	0

Table 1: Data for $\mathcal{D}$

For a set of observations $\mathcal{D} = \{(y^{(i)}, x^{(i)})\}$, where $y^{(i)}, x^{(i)} \in \mathbb{R}$ and $i \in \{1, 2, \ldots, |\mathcal{D}|\}$, we optimize the following program.

$$\underset{w_1, w_2}{\operatorname{argmin}} \sum_{(y^{(i)}, x^{(i)}) \in \mathcal{D}} (y^{(i)} - w_1 \cdot x^{(i)} - w_2)^2 \tag{10}$$

Find the optimal w_1^*, w_2^* given the aforementioned dataset $\mathcal{D}$ and justify your answer. **Compute the scalars w_1^* and w_2^*.**

Solution: Plot it out. From geometry, the line goes through (0,-1) and (0,1). $w_1 = 2$(1 points), $w_2 = -1$. (1 points)
Total Points: 2

(b) What is the minimum number of observations that are required to obtain a unique solution for the program in Eq. (10)?

Solution: Two observations. (1 points)

(c) Consider another dataset $\mathcal{D}_1$, where $x^{(i)}, y^{(i)} \in \mathbb{R}$

i	$x^{(i)}$	$y^{(i)}$
1	0	0
2	1	1
3	2	4
4	3	9
5	4	16

Table 2: Data for $\mathcal{D}_1$

Clearly $\mathcal{D}_1$ can not be fit exactly with a linear model. In class, we discussed a simple approach of building a nonlinear model while still using our linear regression tools. How would you use the linear regression tools to obtain a nonlinear model which better fits $\mathcal{D}_1$, *i.e.*, what feature transform would you use? Provide your reasons and write down the resulting program that you would optimize using a notation which follows Eq. (10), *i.e.*, make all the trainable parameters explicit. **Do NOT plug the datapoints from $\mathcal{D}_1$ into your program and solve for its parameters. Just provide the program.**

Solution: Observe: the data behaves as x^2 (1 point). Therefore, square the features to get a better fit. We obtain the following program (1 point):

$$\underset{w_1,w_2}{\operatorname{argmin}} \sum_{i}^{N} (y^{(i)} - w_1 \cdot (x^{(i)})^2 - w_2)^2$$

Total Points: 2

(d) Write down a program equivalent to the one derived in part (c) using matrix-vector notation. Carefully define the matrices and vectors which you use, their dimensions and their entries. Show how you fill the matrices and vectors with the data. Derive the closed form solution for this program using the symbols which you introduced. **Do NOT compute the solution numerically.**

Solution: Derivation (1 points)

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}, \quad \mathbf{X}^T = \begin{bmatrix} {x^{(1)}}^2 & 1 \\ {x^{(2)}}^2 & 1 \\ & \vdots \\ {x^{(N)}}^2 & 1 \end{bmatrix}$$

(1 points)

$$\mathbf{w}^* = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{Y}$$

(1 points)
Total Points: 3

(e) Briefly describe the problem(s) that we will encounter if we were to fit a very high degree polynomial to the dataset $\mathcal{D}_1$?

Solution: Overfitting and poor generalization. (1 points)

6. [**14 points**] Linear and Logistic Regression

(a) Using vector $y \in \mathbb{R}^N$, matrix $X \in \mathbb{R}^{N \times d}$, and vector $w \in \mathbb{R}^d$, linear regression can be formulated as

$$\min_w \frac{1}{2}\|y - Xw\|_2^2 \tag{11}$$

What does N and d refer to and what is the optimal argument w^* to the program given in Eq. 11.

Solution:
N: number of samples; d: dimensions of the features; Solution: $w^* = (X^\top X)^{-1} X^\top y$

(b) What is an issue when using the program given in Eq. 11 for classification rather than regression?

Solution: We penalize samples that are very easy to classify

(c) Assume we are given a dataset $\mathcal{D} = \{(x_i, y_i)\}$ containing sample pairs composed of datapoints $x_i \in \mathbb{R}^d$ and class labels $y_i \in \{0, 1\}$. Further, assume we are given a probabilistic model $p(y_i|x_i)$. Under the assumption that all samples are independent and identically distributed (i.i.d.), what is the probability/likelihood of the dataset under the model $p(y_i|x_i)$.

Solution:
$p(\mathcal{D}) = \prod_{(x_i,y_i)\in\mathcal{D}} p(y_i|x_i)$

(d) Assume our probabilistic model depends on some parameters $w \in \mathbb{R}^d$ and is given by

$$p(y_i|x_i) = \frac{1}{1 + \exp(-y_i w^\top x_i)}$$

What is the negative log-likelihood of the i.i.d. dataset $\mathcal{D}$ under this model, and how do we want to choose the parameters w of the model?

Solution:

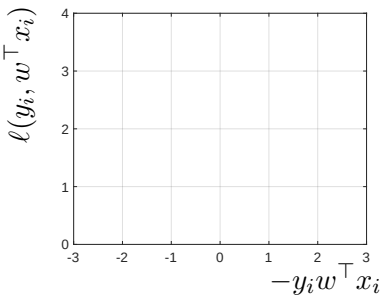
$$\min_w \underbrace{\sum_{(x_i,y_i)\in\mathcal{D}} \log\left(1 + \exp(-y_i w^\top x_i)\right)}_{\text{neg. log-lik}}$$

(e) Without further assumptions and restrictions on the program you derived, is it possible to analytically compute the solution to the program? Yes or No? Justify your answer.

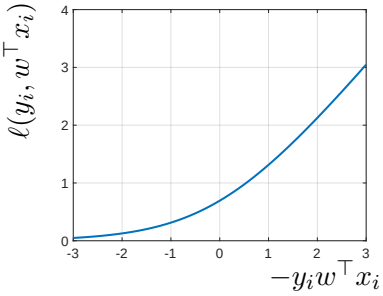
Solution: No, setting the gradient to zero is not solvable analytically

(f) Determine R and ℓ by comparing your program to the following formulation and complete the given illustration:

$$\min_{w\in\mathbb{R}^d} R(w) + \sum_{(x_i,y_i)\in\mathcal{D}} \ell(y_i, w^\top x_i)$$



Solution:
$R(w) = 0$; $\ell(y_i, w^\top x_i) = \log(1 + \exp(-y_i w^\top x_i))$



7. **[12 points]** Binary Classifiers

(a) Is it possible to use a linear regression model for binary classification? If so, how do we map the regression output $\mathbf{w}^\top \mathbf{x}$ to the class labels $y \in \{-1, 1\}$?

Solution:
(2 points) Yes,

$$y = \text{sign}(\mathbf{w}^\top \mathbf{x})$$

(b) Assume $y \in \{-1, 1\}$. Consider the following program for logistic regression:

$$\min_{\mathbf{w}} f(\mathbf{w}) := \sum_i \log\left(1 + \exp(-y^{(i)} \mathbf{w}^T \phi(x^{(i)}))\right).$$

Assume the data is not linearly separable. Is there an analytical solution? If so, derive it. If not, sketch an algorithm that solves the program iteratively. (Make sure to include any important mathematical expressions.)

Solution:
(6 points) No, use gradient descent.
Obtain gradient w.r.t. $\mathbf{w}$,

$$\nabla_{\mathbf{w}} f(\mathbf{w}) = \sum_i \frac{-y^{(i)} \phi(x^{(i)}) \exp(-y^{(i)} \mathbf{w}^T \phi(x^{(i)}))}{1 + \exp(-y^{(i)} \mathbf{w}^T \phi(x^{(i)}))}$$

Initialize $t = 0$, $\mathbf{w}_t$ and stepsize α
While not converged,

i. $\mathbf{g}_t = \nabla_{\mathbf{w}} f(\mathbf{w}_t)$

ii. $\mathbf{w}_{t+1} := \mathbf{w}_t - \alpha \mathbf{g}_t$

iii. $t := t + 1$

(c) The above program for binary classification, makes assumptions on the samples/data points. What are those assumptions.

Solution: (2 point) Independently drawn from an identical distribution

(d) What is the probability model assumed for logistic regression and linear regression? Give names rather than equations.

Solution: (2 points) Logistic/Binomial vs Gaussian probability model

8. **[8 points]** Binary Classifiers

Based on a data set, $D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), ..., (\mathbf{x_N}, y_N)\}$, where $\mathbf{x_i} \in \mathbb{R}^d$, $y_i \in \{0, 1\}$, and samples are i.i.d., we want to train a logistic regression model. We define our probabilistic model to have the form:

$$\hat{y}_i = \mathrm{g}(\mathbf{w}^T \mathbf{x_i}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x_i}}},$$

where $\hat{y}_i$ is the probability given data $\mathbf{x_i}$ and model parameters $\mathbf{w} \in \mathbb{R}^d$. Given this notation we define the probability of predicting y_i via

$$P[Y = y_i | X = \mathbf{x_i}] = (\hat{y}_i)^{y_i} \cdot (1 - \hat{y}_i)^{(1-y_i)}.$$

We want to find the model parameters $\mathbf{w}$, such that the likelihood of the data set D is maximized, which is formulated as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left(-\sum_{i=1}^{N} (y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)) \right).$$

(a) (6 points) Let the program above be referred to as:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(y, \mathbf{x}, \mathbf{w})$$

Is $L(y, \mathbf{x}, \mathbf{w})$ convex with respect to $\mathbf{w}$? Prove it is convex or non-convex without using knowledge of convexity for any function. (Hint: use the Hessian.)

Solution:
(6 points)
Yes. It is convex.

Recall that $\hat{y}_i = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x_i}}} = \mathrm{g}(\mathbf{w}^T\mathbf{x_i})$ where $\mathrm{g}(a) = \frac{1}{1+e^{-a}}$ and $\mathrm{g}'(a) = \mathrm{g}(a)(1-\mathrm{g}(a))$.

In order to prove the convexity of $L = \sum_{i=1}^{N}[y_i \cdot (-\log(\hat{y}_i)) + (1-y_i)\cdot(-\log(1-\hat{y}_i))]$, we only need to prove the convexity of $(-\log(\hat{y}_i))$ and $(-\log(1-\hat{y}_i))$:

$$\nabla_{\mathbf{w}}(-\log(\hat{y}_i)) = (\hat{y}_i - 1)\mathbf{x_i}$$

$$\nabla^2_{\mathbf{w}}(-\log(\hat{y}_i)) = \hat{y}_i(1-\hat{y}_i)\mathbf{x_i}\mathbf{x_i}^T$$

$$\nabla_{\mathbf{w}}(-\log(1-\hat{y}_i)) = \hat{y}_i\mathbf{x_i}$$

$$\nabla^2_{\mathbf{w}}(-\log(1-\hat{y}_i)) = \hat{y}_i(1-\hat{y}_i)\mathbf{x_i}\mathbf{x_i}^T$$

Prove their Hessian are positive semi-definite: $\forall \mathbf{z}$,

$$\mathbf{z}[\hat{y}_i(1-\hat{y}_i)\mathbf{x_i}\mathbf{x_i}^T]\mathbf{z}^T = \hat{y}_i(1-\hat{y}_i)(\mathbf{z}^T\mathbf{x_i})(\mathbf{x_i}^T\mathbf{z}) = \hat{y}_i(1-\hat{y}_i)(\mathbf{x_i}^T\mathbf{z})^T(\mathbf{x_i}^T\mathbf{z}) = \hat{y}_i(1-\hat{y}_i)(\mathbf{x_i}^T\mathbf{z})^2 \geq 0$$

Therefore L is convex with respect to $\mathbf{w}$.

(b) (2 points) Can we find a closed form analytic solution for $\mathbf{w}$? How to train the model $\mathbf{w}$ based on the data set D? State your approach and write down the equation.

Solution:
(2 points)
We can train $\mathbf{w}$ by gradient descent:

$$\mathbf{w_{t+1}} := \mathbf{w_t} - \alpha\sum_{i=1}^{N}[-y_i\mathbf{x_i} + \mathbf{x_i}(\frac{1}{1+e^{-\mathbf{w_t}^T\mathbf{x_i}}})]$$

9. **[10 points]** Binary Classifiers

(a) Assume $y \in \{-1, 1\}$. Consider the following program for linear regression:

$$\min_{\mathbf{w}} f(\mathbf{w}) := \frac{1}{2}\sum_i \left(1 - y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)}\right)^2$$

Is the objective function $f(\mathbf{w})$ convex in $\mathbf{w}$ assuming everything else given and fixed? (Yes or No)

Solution:
(1 points)
Yes.

(b) Assume $y \in \{-1, 1\}$. Consider the following program for logistic regression:

$$\min_{\mathbf{w}} f(\mathbf{w}) := \sum_i \log\left(1 + \exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})\right)$$

Is the objective function $f(\mathbf{w})$ convex in $\mathbf{w}$ assuming everything else given and fixed? (Yes or No)

Solution:
(2 points)
Yes.

(c) We want to use gradient descent to address the above **logistic regression** program. What is the gradient $\nabla_{\mathbf{w}} f(\mathbf{w})$? Use the symbols and notation which was used in the cost function.

Solution:
(5 points)

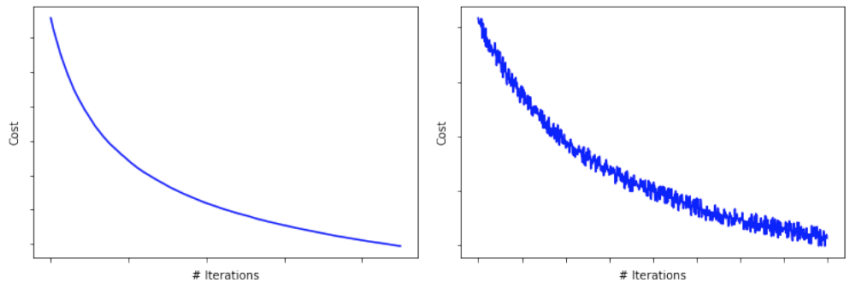
$$\nabla_{\mathbf{w}} f(\mathbf{w}) = \sum_i \frac{-y^{(i)}\mathbf{x}^{(i)} \exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}{1 + \exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}$$

(d) What is the probability model assumed for logistic regression and linear regression? Give names rather than equations.

Solution:
(2 points)
Logistic/Binomial vs Gaussian probability model

10. **[17 points]** Optimization

(a) [2 points] Fig. 2 shows the cost curves of a deepnet training with two different optimization algorithms, *i.e.*, **gradient descent** and **stochastic gradient descent**. Which of the graphs corresponds to which optimization algorithm? Explain.



(a) Graph A (b) Graph B

Figure 2: Training cost as a function of the number of iterations.

Solution:
Graph A: Gradient descent. The cost goes down at every single iteration (smooth curve).
Graph B: Stochastic gradient descent. The cost does not decrease at every iteration since we are just training on a mini-batch (noisier)

(b) [1 points] Fig. 3 below shows the cost curve during a deepnet training. Which of the following options could have caused the sudden drop in the cost: (1) learning rate-decay (decreasing the learning rate), (2) increasing the batch-size. Explain.

Solution: Learning rate decay. The curve did not become smoother. Hence, the drop is not due to an increase of the batch size.

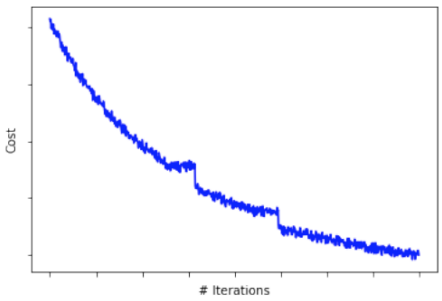


Figure 3: Training cost as a function of the number of iterations.

(c) [3 points] Consider the following program:

$$\min_{w} \frac{1}{2} \max_{i \in \{1,\cdots,k\}} (a_i^T w + b_i) + \frac{1}{2} \|w\|^2 , \tag{12}$$

where $w, a_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$ for $1 \leq i \leq k$. To rewrite the optimization problem as a quadratic programming program, we substitute the maximum term with an auxiliary slack variable $\xi \in \mathbb{R}$. Fill in the missing constraint in the program below:

$$\min_{w,\xi} \xi + \frac{1}{2} \|w\|^2 \tag{13}$$
$$\text{s.t.} \quad \underset{\text{(missing constraint)}}{\ldots\ldots\ldots\ldots}$$

Solution:

$$\min_{w,\xi} \xi + \frac{1}{2} \|w\|^2 \tag{14}$$
$$\text{s.t } \tfrac{1}{2}(a_i^T w + b_i) \leq \xi (\forall i) \tag{15}$$

(d) [2 points] Consider the following program:

$$\min_{w,\xi} \xi + \frac{1}{2} \|w\|^2$$
$$\text{s.t. } A^T w + b \leq \xi \mathbf{1} \tag{16}$$

Write down the Lagrangian $L(w, \xi, u)$ of the program in Eq. (16), where $u \in \mathbb{R}^k$ denotes the Lagrange multiplier, $u \geq 0$, $A \in \mathbb{R}^{d \times k}$, $b \in \mathbb{R}^k$ and $\mathbf{1} \in \mathbb{R}^k$ is a vector of all ones.

Solution:

$$L(w, \xi, u) = \xi + \frac{1}{2} \|w\|_2^2 + u^T (A^T w + b - \xi \mathbf{1}) \tag{17}$$

(e) [3 points] Write down the dual program of the program in Eq. (16) as a function of $L(w, \xi, u)$, the **min** and **max** operators, as well as w, ξ and u. Make sure to specify all the constraints.

Solution:

$$\max_{u \geq 0} \min_{w,\xi} L(w, \xi, u) \tag{18}$$

(f) [2 points] Show that the solution to the program $\min_w L(w, \xi, u)$ satisfies the constraint $w = -Au$.
Hint: Compute $\frac{\partial L(w,\xi,u)}{\partial w}$.

Solution:

$$\frac{\partial L(w, \xi, u)}{\partial w} = 0 \longleftrightarrow w + Au = 0. \tag{19}$$

(g) [2 points] Show that solving $\min_\xi L(\xi, u)$ results in the constraint $\|u\|_1 = 1$.

Solution: Plugging $-uA$ into $L(w, \xi, u)$ results in:

$$L(\xi, u) = \xi - \frac{1}{2} u^T A^T A u + u^T b - \xi u^T \mathbf{1} \tag{20}$$

Solving $\min_\xi$:

$$\frac{\partial L(w, \xi, u)}{\partial \xi} = 0 \longleftrightarrow u^T \mathbf{1} = 1 \longleftrightarrow \|u\|_1 = 1 \tag{21}$$

(h) [2 points] Using parts (e)-(g), show that the dual to Eq. (16) is:

$$\begin{aligned} \max_u \quad & -\tfrac{1}{2} u^T A^T A u + u^T b \\ \text{s.t.} \quad & u \geq 0, \|u\|_1 = 1 \end{aligned} \tag{22}$$

Solution:

$$L(\xi, u) = \xi - \frac{1}{2} u^T A^T A u + u^T b - \xi u^T \mathbf{1} \tag{23}$$

We remove the term related to ξ and replace it by the constraint from part (g). We obtain the quadratic program:

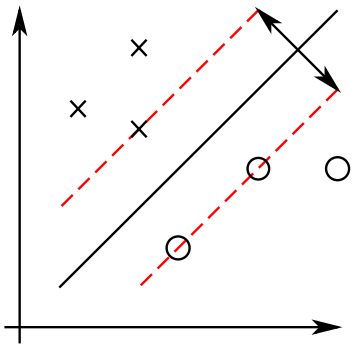
$$\max_u \quad -\tfrac{1}{2} u^T A^T A u + u^T b \tag{24}$$

$$\text{s.t} \quad u \geq 0, \|u\|_1 = 1 \tag{25}$$

11. **[13 points]** Support Vector Machines

(a) Give a high level explanation of the binary support vector machine in a few words. Use a diagram to help illustrate your explanation.

Solution:
(1 point) Binary SVM finds separating hyperplane which maximizes margin between data points of two classes
(2 points)



(b) Consider the following unconstrained program for a binary SVM

$$\min_{\mathbf{w}} \|\mathbf{w}\|_2^2 + \sum_i \max\{0, 1 - y^{(i)}\mathbf{w}^T\phi(x^{(i)})\} \tag{26}$$

One possible approach to tackle this problem is via gradient descent. However, a term in the cost function is not differentiable. Which is the offending term and how can we overcome this?

Solution:
(1 point)$\max\{0, x\}$ is not differentiable.
(1 point)Set its derivative to 0 for $x < 0$ and 1 for $x \geq 0$.

(c) Another approach to addressing the program given in Eq. 26 is the use of the dual problem. Rewrite the program given in Eq. 26 as an equivalent constrained program, next obtain the dual objective. Carefully define the Lagrangian and the KKT conditions as well as their solution. Note that we are looking for the dual to the program in Eq. 26 and not a more general form.

Solution:
(1 point) Constrained program

$$\min_{\mathbf{w},\xi\geq 0} \|\mathbf{w}\|_2^2 + \sum \xi \qquad \text{s.t.} \qquad y\mathbf{w}^T\phi(x) \geq 1 - \xi$$

(4 points) Lagrangian, KKT

$$L(\cdot) = \|\mathbf{w}\|_2^2 - \mathbf{w}^T \sum_i \alpha^{(i)} y^{(i)} \phi(x^{(i)}) + \sum_i \xi^{(i)}(1 - \alpha^{(i)}) + \sum_u \alpha^{(i)}$$

$$\frac{\partial L}{\partial \mathbf{w}}: \qquad 2\mathbf{w} = \sum_i \alpha^{(i)} y^{(i)} \phi(x^{(i)})$$

$$\min_{\xi^{(i)}\geq 0} \xi^{(i)}(1 - \alpha^{(i)}) \implies \alpha^{(i)} \leq 1$$

(1 point)

$$\max_{0\leq\alpha\leq 1} -\frac{1}{4}\left\|\sum_i a^{(i)} y^{(i)} \phi(x^{(i)})\right\|_2^2 + \sum_i \alpha^{(i)}$$

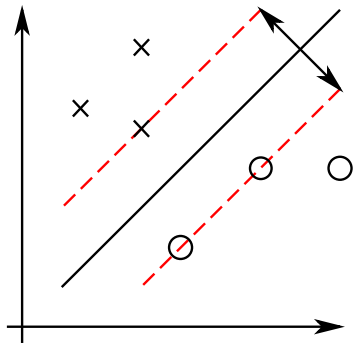
(d) What observation can you make about the dual and why is this useful?

Solution: (2 points) Objective is quadratic and QP solvers can be used to solve the optimization problem efficiently/solver exists.

12. **[15 points]** Binary and Multi-class Support Vector Machine for Object Detection

(a) Explain the intuition behind a binary support vector machine in a few words and draw an illustration that underlines your explanation.

Solution:
Binary SVM finds separating hyperplane which maximizes margin between datapoints of two classes



(b) The program for a binary SVM is as follows when given a dataset $\mathcal{D} = \{(x_i, y_i)\}$ containing pairs of input data $x_i \in \mathbb{R}^d$ and corresponding label $y_i \in \{0, 1\}$:

$$\min_{w \in \mathbb{R}^d, \xi_i \geq 0} \frac{C}{2}\|w\|_2^2 + \sum_{(x_i,y_i)\in\mathcal{D}} \xi_i \qquad \text{s.t.} \qquad y_i w^\top x_i \geq 1 - \xi_i \quad \forall (x_i, y_i) \in \mathcal{D} \tag{27}$$

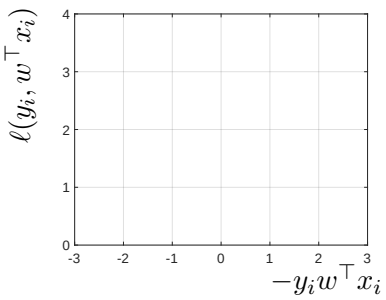
Transform this program into an unconstrained program (not using Lagrangian).

Solution: $\min_w \frac{C}{2}\|w\|_2^2 + \sum_{(x_i,y_i)\in\mathcal{D}} \max\{0, 1 - y_i w^\top x_i\}$

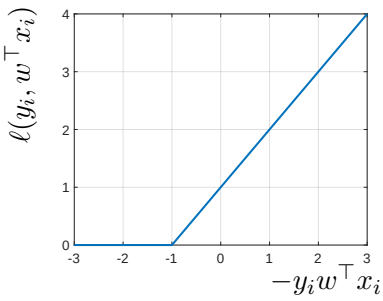
(c) Relate your program to the task of empirical risk minimization which is given in the following:

$$\min_{w \in \mathbb{R}^d} R(w) + \sum_{(x_i,y_i)\in\mathcal{D}} \ell(y_i, w^\top x_i)$$

and complete the enclosed plot while paying attention to the given axis labels.



Solution: $R(w) = \frac{C}{2}\|w\|_2^2$ and $\ell(y_i, w^\top x_i) = \max\{0, 1 - y_i w^\top x_i\}$



(d) Provide intuition and the program formulation for the extension of the task given in Eq. 27 to multiple classes, i.e., for the case of $y_i \in \{0, 1, \ldots, K-1\}$. Keep your formulation similar to the form given in Eq. 27.

Solution:
The groundtruth label should score higher than any other label, i.e., $w_{y_i}^\top x_i \geq w_{\hat{y}_i}^\top x_i$ $\forall (x_i, y_i) \in \mathcal{D}, \hat{y}_i$

$$\min_{w, \xi_i \geq 0} \frac{C}{2}\|w\|_2^2 + \sum_{(x_i, y_i) \in \mathcal{D}} \xi_i \qquad \text{s.t.} \quad w_{y_i}^\top x_i - w_{\hat{y}_i}^\top x_i \geq 1 - \xi_i \quad \forall (x_i, y_i) \in \mathcal{D}, \hat{y}_i$$

13. **[11 points]** L2 SVM

We are given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{|\mathcal{D}|}$ with feature vectors $\mathbf{x}^{(i)} \in \mathbb{R}^d$ and their corresponding labels $y^{(i)} \in \{-1, 1\}$. The following is the primal formulation of L2 SVM, a variant of the standard SVM obtained by **squaring** the hinge loss,

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{2}\textstyle\sum_{i=1}^{|\mathcal{D}|}(\xi^{(i)})^2 \\ s.t. \quad & y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \xi^{(i)}, \ i \in \{1, \cdots, |\mathcal{D}|\} \\ & \xi^{(i)} \geq 0, i \in \{1, \cdots, |\mathcal{D}|\} \end{aligned} \tag{28}$$

Here, $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are learnable weights and $\boldsymbol{\xi} = (\xi^{(1)}, \ldots, \xi^{(|\mathcal{D}|)}) \in \mathbb{R}^{|\mathcal{D}|}$ are slack variables. We will first show that removing the last set of constraints $\boldsymbol{\xi} \geq 0$ does not change the optimal solution of the problem, *i.e.*, we will show that for the optimal solution $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$, the inequality $\boldsymbol{\xi}^* \geq 0$ always holds.

(a) Assume that the dataset consists of 3 samples, that $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ is the optimal solution to the problem without the last set of constraints, *i.e.*, $\boldsymbol{\xi} \geq 0$, and that $(\xi^{(3)})^* = 0$. Write down the resulting expression of the loss as a function of $\mathbf{w}^*$, $(\xi^{(1)})^*$ and $(\xi^{(2)})^*$.

Solution:
(1 points) $\frac{1}{2}(\mathbf{w}^*)^T\mathbf{w}^* + \frac{C}{2}(((\xi^{(1)})^*)^2 + ((\xi^{(2)})^*)^2)$

(b) Assume that the dataset consists of 3 samples, that $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ is the optimal solution to the problem without the last set of constraints, *i.e.*, $\boldsymbol{\xi} \geq 0$, and that $(\xi^{(3)})^* < 0$. Write-down the expression of the resulting loss and compare it with the one from **part a**. Which of the losses has a larger value?

Solution: (2 points) $\frac{1}{2}(\mathbf{w}^*)^T\mathbf{w}^* + \frac{C}{2}(((\xi^{(1)})^*)^2 + ((\xi^{(2)})^*)^2 + ((\xi^{(3)})^*)^2) > \frac{1}{2}(\mathbf{w}^*)^T\mathbf{w}^* + \frac{C}{2}(((\xi^{(1)})^*)^2 + (\xi^{(2)})^*)^2$

(c) Consider the general case where $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ is the optimal solution to the problem without the constraints $\boldsymbol{\xi} \geq 0$. Suppose, there exists some $(\xi^{(j)})^* < 0$. Show that $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ with $\hat{\mathbf{w}} = \mathbf{w}^*$, $\hat{b} = b^*$, $\hat{\xi}^{(i)} = (\xi^{(i)})^*$ $(\forall i \neq j)$ and $\hat{\xi}^{(j)} = 0$, is a feasible solution.

Solution: (2 point) To prove that $\hat{\xi}^{(j)}$ is a feasible solution, we need to prove that it satisfies the constraints. For $(\xi^{(j)})^* < 0$, we get $y^{(j)}((\mathbf{w}^*)^T\mathbf{x}^{(j)} + b^*) \geq 1 - (\xi^{(j)})^* > 1$. When evaluating the same constraint for $\hat{\xi}^{(j)} = 0$, we get $y^{(j)}((\mathbf{w}^*)^T\mathbf{x}^{(j)} + b^*) \geq 1$. Hence, $y^{(j)}((\mathbf{w}^*)^T\mathbf{x}^{(j)} + b^*) > 1$ is not violated. This implies that $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ is a feasible solution.

(d) Compare the losses obtained for $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ and $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$. Conclude that the optimal solution does not change when removing the constraints $\boldsymbol{\xi} \geq 0$.

Solution: (2 point) $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ is feasable and leads to a smaller loss than $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$. This is a contradiction to the assumption that $(\xi^{(j)})^*$ is optimal. Hence, for the optimal solution $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$, the inequality $\xi^{(i)} \geq 0$ $(\forall i)$ always holds.

(e) After removing the constraints $\boldsymbol{\xi} \geq 0$, we get a simpler program

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{2}\textstyle\sum_{i=1}^{|\mathcal{D}|}(\xi^{(i)})^2 \\ s.t. \quad & y^{(i)}(\mathbf{w}^T\boldsymbol{x}^{(i)} + b) \geq 1 - \xi^{(i)},\ i \in \{1, \cdots |\mathcal{D}|\}. \end{aligned} \tag{29}$$

Give the Lagrangian of the program above as a function of $\mathbf{w}$, b, $\xi^{(i)}$, $y^{(i)}$, $\boldsymbol{x}^{(i)}$, C, $\mathcal{D}$ and Lagrange multipliers $\alpha^{(i)}$. What's the range of the Lagrange multipliers $\alpha^{(i)}$?

Solution: (2 points)
$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{2}\sum_{i=1}^{|\mathcal{D}|}(\xi^{(i)})^2 - \sum_{i=1}^{|\mathcal{D}|}\alpha^{(i)}(y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) - 1 + \xi^{(i)})$, with $\alpha^{(i)} \geq 0$ $(\forall i)$.

(f) Show that the dual of the program in Eq. (29) is

$$\begin{aligned} \max_{\boldsymbol{\alpha} \geq 0} \quad & -\tfrac{1}{2}\boldsymbol{\alpha}^T(\mathbf{Q} + \mathbf{I}\tfrac{1}{C})\boldsymbol{\alpha} + \mathbf{1}^T\boldsymbol{\alpha} \\ s.t. \quad & \mathbf{y}^T\boldsymbol{\alpha} = 0 \end{aligned}$$

with $Q_{ij} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j$, $\boldsymbol{\alpha} \in \mathbb{R}^{|\mathcal{D}|}$ is a vector of Lagrange multipliers and $\mathbf{y} \in \mathbb{R}^{|\mathcal{D}|}$ a vector of labels.

Solution: (4 points) Taking partial derivatives of the Lagrangian wrt $\mathbf{w}$, b and $\xi^{(i)}$,

$$\nabla_{\boldsymbol{w}} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = 0 \Longleftrightarrow \mathbf{w} = \sum_{i=1}^{|\mathcal{D}|} \alpha^{(i)} y^{(i)} \mathbf{x}^{(i)} \tag{30}$$

$$\partial_b L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = 0 \Longleftrightarrow \sum_{i=1}^{|\mathcal{D}|} \alpha^{(i)} y^{(i)} = 0 \tag{31}$$

$$\partial_{\xi^{(i)}} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = 0 \Longleftrightarrow \xi^{(i)} = \alpha^{(i)}/C \tag{32}$$

Plugging these back to the Lagrangian, rearranging terms and keeping constraints on the Lagrange multipliers, we obtain the dual.

$$\begin{aligned} \max_{\boldsymbol{\alpha} \geq 0} \quad & -\tfrac{1}{2} \boldsymbol{\alpha}^T (\mathbf{Q} + \tfrac{1}{C} \mathbf{I}) \boldsymbol{\alpha} + \mathbf{1}^T \boldsymbol{\alpha} \\ s.t. \quad & \mathbf{y}^T \boldsymbol{\alpha} = 0 \end{aligned}$$

14. **[15 points]** Support Vector Machines for Function Estimation.
Suppose we are given a dataset $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), ...\}$, where $\mathcal{D} \subset \mathbb{R}^d \times \mathbb{R}$. We want to find a model $f(\mathbf{x}^{(i)}) = \mathbf{w}^\intercal \mathbf{x}^{(i)}$ such that the deviation from the corresponding $y^{(i)}$ is at most ϵ, where $\epsilon > 0$, while maintaining a small $\|\mathbf{w}\|_2^2$.

We can write this goal as the following optimization problem:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 && (33) \\ \text{subject to} \quad & \mathbf{w}^\intercal \mathbf{x}^{(i)} - y^{(i)} \leq \epsilon, \ i = 1, \ldots, |\mathcal{D}| && (34) \\ & y^{(i)} - \mathbf{w}^\intercal \mathbf{x}^{(i)} \leq \epsilon, \ i = 1, \ldots, |\mathcal{D}|. && (35) \end{aligned}$$

(a) (2 points) Consider the following toy-dataset to build intuition, where both $x^{(i)}$ and $y^{(i)}$ are in $\mathbb{R}$. Find the optimal w^* for the optimization problem given in Eq. (33)-(35), where $\epsilon = 2$. Explain your reasoning. **(Only use the toy-dataset for this subproblem.)**

i	$x^{(i)}$	$y^{(i)}$
1	0	2
2	2	0

Table 3: Toy-dataset $\mathcal{D}$

Solution: Without a bias term, the line has to pass through (0,0). To satisfy the constraint for point (2,0), and minimize $||w||_2^2$, $w^* = 0$.
(2 Points, for correct reasoning and correct w^*.)

(b) (1 point) The above optimization problem given in Eq. (33) – Eq. (35) may not always

be feasible. Consider the following optimization problem:

$$\begin{aligned} \underset{\mathbf{w},\xi,\hat{\xi}}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 + \sum_i^{|\mathcal{D}|}(\xi^{(i)} + \hat{\xi}^{(i)}) \\ \text{subject to} \quad & \mathbf{w}^\intercal \mathbf{x}^{(i)} - y^{(i)} \leq \epsilon + \xi^{(i)},\ i = 1, \ldots, |\mathcal{D}| \\ & y^{(i)} - \mathbf{w}^\intercal \mathbf{x}^{(i)} \leq \epsilon + \hat{\xi}^{(i)},\ i = 1, \ldots, |\mathcal{D}| \\ & \xi^{(i)}, \hat{\xi}^{(i)} \geq 0,\ i = 1, \ldots, |\mathcal{D}|. \end{aligned} \tag{36}$$

What are $\xi^{(i)}$ and $\hat{\xi}^{(i)}$ called?

Solution: Slack variables (1 point)

(c) (2 points) Explain why the optimization problem given in Eq. (36) handles the infeasible constraints, and why $\xi^{(i)}$ and $\hat{\xi}^{(i)}$ has to be greater than 0.

Solution: Relax the constraints by adding a positive term to ϵ, then penalizes the added amount in the loss function.
If ξ and $\hat{\xi}$ can be less than 0, the cost function is unbounded.
(2 Points, one point for each answer).

(d) (1 point) Let the optimization problem given in Eq. (36) be the primal problem.
What is the Lagrange function $\mathcal{L}$ for this primal problem? For the Lagrange multipliers use the symbols $\alpha^{(i)}, \hat{\alpha}^{(i)}, \mu^{(i)}, \hat{\mu}^{(i)}$, specifically (don't leave out any Lagrange multipliers, i.e., treat all constraints explicitly):

- Use $\alpha^{(i)}$ for constraints involving $\mathbf{w}$ and $\xi^{(i)}$.
- Use $\hat{\alpha}^{(i)}$ for constraints involving $\mathbf{w}$ and $\hat{\xi}^{(i)}$.
- Use $\mu^{(i)}$ for constraints involving only $\xi^{(i)}$.
- Use $\hat{\mu}^{(i)}$ for constraints involving only $\hat{\xi}^{(i)}$.

Solution: Introduce Lagrange Multiplier and follow definition.

$$\begin{aligned} \mathcal{L} = & \frac{1}{2}\|\mathbf{w}\|_2^2 + \sum_i^{|\mathcal{D}|}(\xi^{(i)} + \hat{\xi}^{(i)}) \\ & - \sum_i^{|\mathcal{D}|}(\mu^{(i)}\xi^{(i)} + \hat{\mu}^{(i)}\hat{\xi}^{(i)}) \\ & - \sum_i^{|\mathcal{D}|}\alpha^{(i)}(\mathbf{w}^\intercal \mathbf{x}^{(i)} - y^{(i)} - (\epsilon + \xi^{(i)})) \\ & - \sum_i^{|\mathcal{D}|}\hat{\alpha}^{(i)}(y^{(i)} - \mathbf{w}^\intercal \mathbf{x}^{(i)} - (\epsilon + \hat{\xi}^{(i)})) \end{aligned}$$

(1 Points, correct or not correct.)

(e) (7 points) Derive the dual optimization problem, and simplify (*i.e.*, the final optimization problem should only include $\alpha^{(i)}$, $\hat{\alpha}^{(i)}$, $\mathbf{x}^{(i)}$, $y^{(i)}$, ϵ). (You must show all your work.)

Solution: Take derivative with respect to the $\mathbf{w}, \xi^{(i)}, \hat{\xi}^{(i)}$.

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{i}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)})x^{(i)} = 0$$

$$\frac{\partial \mathcal{L}}{\partial \xi^{(i)}} = 1 - \alpha^{(i)} - \mu^{(i)}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{\xi}^{(i)}} = 1 - \hat{\alpha}^{(i)} - \hat{\mu}^{(i)}$$

Note that we also have $\hat{\mu}^{(i)} \geq 0$, $\mu^{(i)} \geq 0$, $\hat{\alpha}^{(i)} \geq 0$, $\alpha^{(i)} \geq 0$
Substitute back into the Lagrangian

$$\begin{aligned}\underset{\alpha,\hat{\alpha}}{\text{maximize}} \quad & \frac{-1}{2}\sum_{i,j=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)})(\alpha^{(j)} - \hat{\alpha}^{(j)})\mathbf{x}^{(i)\intercal}\mathbf{x}^{(j)} \\ & - \epsilon\sum_{i=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)}) + \sum_{i=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)})y^{(i)}\end{aligned}$$

subject to

$\alpha^{(i)}, \hat{\alpha}^{(i)} \in [0, 1]$,

(7 Points, three correct gradients, correct dual function, two correct constraints, simplified formulation which only involves $\alpha, \hat{\alpha}$.)

(f) (2 points) Explain how this dual optimization problem can be extended to non-linear features. Rewrite the optimization problem to involve kernel $\mathcal{K}$.

Solution: The dual optimization problem involves $\mathbf{x}^{(i),\intercal}\mathbf{x}^{(j)}$, we can replace $\mathbf{x}$ with a non-linear transform $\phi(\mathbf{x})$, where $\mathcal{K}(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^\intercal\phi(\mathbf{z})$

$$\begin{aligned}\underset{\alpha,\hat{\alpha}}{\text{maximize}} \quad & \frac{-1}{2}\sum_{i,j=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)})(\alpha^{(j)} - \hat{\alpha}^{(j)})\mathcal{K}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \\ & - \epsilon\sum_{i=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)}) + \sum_{i=1}^{|\mathcal{D}|}(\alpha^{(i)} - \hat{\alpha}^{(i)})y^{(i)}\end{aligned}$$

subject to

$\alpha^{(i)}, \hat{\alpha}^{(i)} \in [0, 1]$,

(2 points, correct explanation and correct optimization formulation using kernel.)

15. **[10 points]** Support Vector Machine

(a) Recall, a hard-margin support vector machine in the primal form optimizes the following program

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{s.t. } y^{(i)}(\mathbf{w}^\intercal\mathbf{x}^{(i)} + b) \geq 1 \ \ , \forall(\mathbf{x}^{(i)}, y^{(i)}) \in \mathcal{D} \tag{37}$$

What is the Lagrangian, $L(\mathbf{w}, b, \alpha)$, of the constrained optimization problem in Eq. (37)?

Solution:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||_2^2 + \sum_i^{|\mathcal{D}|} \alpha^{(i)}(1 - y^{(i)}\mathbf{w}^\intercal \mathbf{x}^{(i)} + b)$$

(1 point)
Total of 1 points

(b) Consider the Lagrangian

$$L(\mathbf{w}, \alpha) := \frac{1}{2}\|\mathbf{w}\|_2^2 + \sum_{(x^{(i)}, y^{(i)}) \in \mathcal{D}} \alpha^{(i)}(1 - y^{(i)}\mathbf{w}^\intercal \mathbf{x}^{(i)}) \tag{38}$$

where $\alpha^{(i)}$ are elements of α. *Note: This Lagrangian is not the same as the solution in the previous part.*

Derive the dual program for the Lagrangian given in Eq. (38). Provide all its constraints if any.

Solution: Take the gradient with respect to $\mathbf{w}$ and set it to 0. (1 point)

$$\sum_{i=1}^{N} \alpha^{(i)} - \frac{1}{2}||\sum_i^N \alpha^{(i)} y^{(i)} x^{(i)}||_2^2$$

(1 point) s.t.

$$\alpha^{(i)} \geq 0$$

(1 point)
Total of 3 points

(c) Recall that a kernel SVM optimizes the following program

$$\max_{\alpha} \sum_{i=1}^{|\mathcal{D}|} \alpha^{(i)} - \frac{1}{2} \sum_{i,j=1}^{|\mathcal{D}|} \alpha^{(i)}\alpha^{(j)} y^{(i)} y^{(j)} \kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \tag{39}$$

s.t. $\alpha^{(i)} \geq 0$ and $\sum_{i}^{|\mathcal{D}|} \alpha^{(i)} y^{(i)} = 0$

We have chosen the kernel to be

$$\kappa(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\intercal \mathbf{z})^2 + 1$$

Consider the following dataset $\mathcal{D}_2$ in the one-dimensional space; $x^{(i)}, y^{(i)} \in \mathbb{R}$.

i	$x^{(i)}$	$y^{(i)}$
1	$\frac{1}{2}$	+1
2	-1	+1
3	$\sqrt{3}$	-1
4	4	-1

What are the optimal primal parameters, $\mathbf{w}^*, b^*$ when optimizing the program in Eq. (39) on the dataset $\mathcal{D}_2$. Note: b is NOT included in the margin or the features (treat it explicitly).

Hint: First, construct a feature vector $\phi(\mathbf{x}) \in \mathbb{R}^2$ such that $\kappa(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^\intercal \phi(\mathbf{z})$ for the given one dimensional dataset. Then use this feature vector to transform the data $\mathcal{D}_2$ into feature space and plot the result. Read of the bias term b and the optimal weight vector $\mathbf{w}^*$.

Solution: Observe that x is one dimensional, then kernel can be written as $\phi^\intercal(x)\phi(z)$, where $\phi(x) = [x^2, 1]^\intercal$. (1 point)
Let $\mathbf{w} = [w_1, w_2]$.
From geometry b has to be the midpoint (1 point), $w_2 = 0$, $b* = 2$ and $w_1 = -c$ for some $c > 0$.
Plug in support vector example (2), $w_1 = -1$.
$\mathbf{w}^* = [-1, 0]$ (1 point) and $b^* = 2$ (1 point)
Total of 4 points

(d) (Continuing from previous part) Which of the points in $\mathcal{D}_2$ are support vectors? What are $\alpha^{(1)}$ and $\alpha^{(2)}$?

Hint: To find $\alpha^{(2)}$ make use of the relationship between the primal solution and the dual variables, *i.e.*, $\mathbf{w}^* = \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \phi(\mathbf{x}^{(i)})$. Assume $\mathbf{w}^* = [-4 \quad 0]^T$ if you couldn't solve part (c).

Solution: Support vectors are example 2 and 3. (1 point)
From the dual formulation, we know that

$$\mathbf{w}^* = \sum_i^N \alpha^{(i)} y^{(i)} \mathbf{z}^{(i)},$$

where N is the size of the dataset and $\mathbf{z} = \phi(x)$.
Plug in the support vectors example (2) and (3) and solve.
$\alpha^{(1)} = 0$ as not a support vector and $\alpha^{(2)} = \frac{1}{2}$. (1 point)
Total of 2 points
If you couldn't solve part (c) using $w^* = [-4, 0]^T$ $\alpha^{(2)} = 2$. (1 point)

16. **[7 points]** Multiclass Classification
Consider the objective function of a multiclass SVM given by

$$\min_{w, \xi^{(i)} \geq 0} \frac{1}{2}\|w\|_2^2 + \sum_{i=1}^{n} \xi^{(i)}$$

$$\text{s.t.} \quad w_{y^{(i)}}^\top \phi(x^{(i)}) - w_{\hat{y}}^\top \phi(x^{(i)}) \geq 1 - \xi^{(i)} \quad \forall i \in \{1, \ldots, n\}; \hat{y} \in \{0, \ldots, K-1\}$$

where $w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{K-1} \end{bmatrix}$.

(a) What's the optimal value of $\xi^{(i)}$, given $\phi(x^{(i)}), y^{(i)}$, and w?

Solution:
(2 points)

$$\hat{\xi}^{(i)} = \max_{\hat{y}} \{1 - w_{y^{(i)}}^\top \phi(x^{(i)}) + w_{\hat{y}}^\top \phi(x^{(i)})\}$$

(b) Rewrite the objective function in unconstrained form, using the optimal value of $\xi^{(i)}$.

Solution:
(1 point)

$$\min_w \frac{1}{2}\|w\|_2^2 + \sum_{i=1}^{n} \max_{\hat{y}} \{1 - w_{y^{(i)}}^\top \phi(x^{(i)}) + w_{\hat{y}}^\top \phi(x^{(i)})\}$$

(c) Briefly explain using English language the reason for using $w_{y^{(i)}}^\top \phi(x^{(i)}) - w_{\hat{y}}^\top \phi(x^{(i)}) \geq 1 - \xi^{(i)}$ in a multiclass SVM formulation, *i.e.*, what does this constraint encourage?

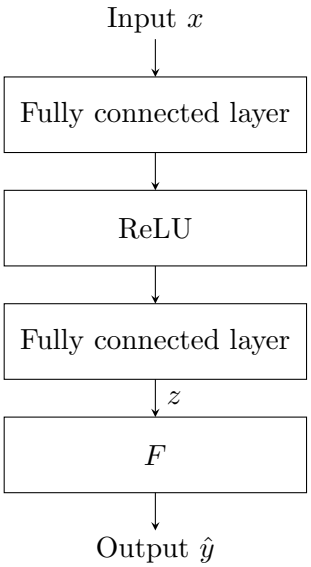
Solution:
(2 points) It encourages the difference between the score of true label $y^{(i)}$ and any class $\hat{y}$ on $x^{(i)}$ to be larger with a margin.

(d) Suppose we want to train a set of one-vs-rest classifiers and a set of one-vs-one classifiers on a dataset of 5,000 samples and 10 classes, each class having 500 samples. Suppose the running time of the underlying binary classifier we use is n^2 in nanoseconds, where n is the size of the training dataset. Which one is faster, training of the one-vs-rest classifiers or training of the one-vs-one classifiers? Explain your reason.

Solution:
(2 points) OVR classifier takes $9 \times 5000^2 = 2.25 \times 10^8$ nanoseconds or $10 \times 5000^2 = 2.5 \times 10^8$ nanoseconds, while OVO classifier takes $45 \times 1000^2 = 4.5 \times 10^7$ nanoseconds. So OVO classifier is faster.

17. **[6 points]** Multiclass Classification via Neural Networks

Suppose we use a multi-layer neural network to classify any input image into one of the following three classes: *apple*, *pear* & *orange*. The neural network architecture is summarized in the following figure:



The output $\hat{y} = F(z)$ is a three-dimensional vector $(\Pr(apple|x), \Pr(pear|x), \Pr(orange|x))$, where $\Pr(c|x)$ denotes the probability of x being in class c.

(a) (1 point) Which function should be used as activation function F, for multiclass classification? (a) logistic (b) softmax (c) ReLU (d) sigmoid. Suppose the input to F is $z = (z_1, z_2, z_3)$, write down the expression of $F(z)$. (Hint: $F(z)$ should sum to 1.)

Solution:
(1 point) (b) softmax.

$$F(z) = \left(\frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}}, \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}}, \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}} \right)$$

(b) (2 points) Suppose we have an alternative activation function G:

$$G(z) = \left(\frac{z_1}{z_1 + z_2 + z_3}, \frac{z_2}{z_1 + z_2 + z_3}, \frac{z_3}{z_1 + z_2 + z_3} \right)$$

which normalizes vector z to sum to 1. Consider the following two inputs $z^{(1)} = (0.01, 0.01, 0.02)$ and $z^{(2)} = (1.01, 1.01, 1.02)$. Use the given inputs to answer whether F and G are *translation invariant*, i.e. the value of the function does **not** change when we

add a constant to all its inputs z_i. Use this fact to give an advantage of using F over G. (Hint: you do not need to exactly evaluate the expressions.)

Solution:
(2 points) It is easy to verify that F is translation invariant by the property of exponential function. $G(z^{(1)}) = (0.25, 0.25, 0.5)$ and $G(z^{(2)}) \approx (1/3, 1/3, 1/3)$ so G is not translation invariant. The fact that softmax function being translation invariant is desirable since it always gives near $1/K$ probability when the K components of z are very close to each other, no matter how large they are.

(c) (2 points) Suppose for an input image, the second fully-connected layer outputs $z = (1, 10^{-5}, 10^{-5})$, which means it is very confident that the image is *apple*, while the true label $y =$ *orange*. Considering this input, give another advantage of using F over G, by evaluating (1) the cross entropy between the true label and classifier prediction $\mathrm{CE}(y, F(z))$, $\mathrm{CE}(y, G(z))$ and (2) their derivatives w.r.t. z_3, where $z = (z_1, z_2, z_3)$.

Solution:
(2 points)

$$\mathrm{CE}(y, F(z)) = -\ln\left(\frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}\right) = -z_3 + \ln(e^{z_1} + e^{z_2} + e^{z_3})$$

$$\mathrm{CE}(y, G(z)) = -\ln\left(\frac{z_3}{z_1 + z_2 + z_3}\right) = -\ln z_3 + \ln(z_1 + z_2 + z_3)$$

$$\frac{\partial \mathrm{CE}(y, F(z))}{\partial z_3} = -1 + \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

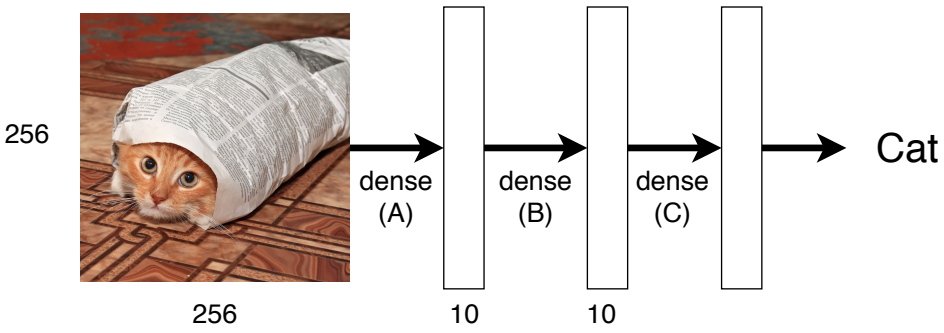
$$\frac{\partial \mathrm{CE}(y, G(z))}{\partial z_3} = -\frac{1}{z_3} + \frac{1}{z_1 + z_2 + z_3}$$

At $z = (1, 10^{-5}, 10^{-5})$, the gradient of $\mathrm{CE}(y, G(z))$ will explode, causing unstable optimization.

(d) (1 point) As an alternative to a multiclass neural network, we can use one-vs-rest multiclass classification, which fits a single-output neural network for each class. Suppose we use the same number of hidden units in the two approaches. Which approach will have faster prediction (output $\hat{y}$ given x) speed? Explain your reason.

Solution:
(1 point) Multiclass neural network. Since only the last layer has 3x parameters comparing to single-output NN, the forward computation time of multiclass NN is less than 3x of single-output NN.

18. **[5 points]** Modeling of Deep Neural Networks
We are interested in building a deep net for image classification. The input is a 256×256 RGB image, and there are 7 different possible output classes. The model architecture is shown below:



(a) How many trainable parameters are in layer "dense (A)"?

Solution: 3 * 256 * 256 * 10 + 10 = 1966090 (1 point)
Missed 3 (-0.5 point)

(b) How many trainable parameters are in layer "dense (B)"?

Solution: 10 * 10 + 10= 110 (1 point)

(c) How many trainable parameters are in layer "dense (C)"?

Solution: 10 * 7 + 7= 77 (1 point)

(d) We redesigned the network by replacing dense (A) and dense (B) layers with a convolution layer of four 3×3 filters, stride 1 and no padding. What are the dimensions for the output of this convolution layer?

Solution: $254 \times 254 \times 4$ (2 point)
((Input size - filter size) / stride) + 1

19. **[9 points]** Representation of Deep Neural Networks
We will use the XOR dataset, $\mathcal{D}$, shown in Table 4. Each example $\mathbf{x}^{(i)} \in \mathbb{R}^2$ and the label $y \in \{0, 1\}$. We are interested in designing classification models for this dataset.

i	$\mathbf{x}_0^{(i)}$	$\mathbf{x}_1^{(i)}$	y
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Table 4: The XOR Dataset $\mathcal{D}$.

(a) Consider a model with the following parameterization:

$$p(y^{(i)}|\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\intercal \mathbf{x}^{(i)} - b)}, \tag{40}$$

where $\mathbf{w} \in \mathbb{R}^2$ and $b \in \mathbb{R}$.

What is the highest accuracy for this model on the XOR dataset? **Note:** To compute accuracy, we use a threshold of 0.5, *i.e.*, the final prediction of the model is $\delta[p(y^{(i)}|\mathbf{x}) > 0.5]$, where δ denotes the indicator function.

Solution: 75% (1 point)

(b) Consider another model with the parametrization shown below:

$$\tilde{y}^{(i)} = \frac{1}{1 + \exp(-a_2^{(i)})} \tag{41}$$

$$a_2^{(i)} = \theta^\intercal \max(\mathbf{a_1^{(i)}}, 0) + b \tag{42}$$

$$\mathbf{a_1^{(i)}} = \mathbf{W}\mathbf{x^{(i)}} + \mathbf{c} \tag{43}$$

where $\theta \in \mathbb{R}^2$, $b \in \mathbb{R}$, $\mathbf{W} \in \mathbb{R}^{2\times 2}$ and $\mathbf{c} \in \mathbb{R}^2$.

Find a θ and b that achieve 100 % accuracy on the XOR dataset, given $\mathbf{W} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$, $\mathbf{c} = [0, -1]^\intercal$. *To show your work, write out* $\mathbf{a_1^{(i)}}$ *and* $a_2^{(i)}$ *for the four datapoints in the XOR dataset and your choice of* θ*, and* b*.*

Note: To compute accuracy, we use a threshold of 0.5, *i.e.*, the final prediction of the model is $\delta[\tilde{y}^{(i)} > 0.5]$, where δ denotes the indicator function.

Solution: Compute $\mathbf{a_1}$

i	$\mathbf{a_1}_0^{(i)}$	$\mathbf{a_1}_1^{(i)}$
0	0	-1
1	1	0
2	1	0
3	2	1

After max:

i	$\mathbf{a_1}_0^{(i)}$	$\mathbf{a_1}_1^{(i)}$
0	0	0
1	1	0
2	1	0
3	2	1

Let $\theta = [1, -2]^\intercal$ and $b = -0.1$ (2 points)

i	$\mathbf{a_2}_0^{(i)}$
0	-0.1
1	0.9
2	0.9
3	-0.1

(1 point)

Sigmoid at 0 corresponds to probability of 0.5.

(c) To learn the parameters of the model in (b), the learning problem is formulated as the following program:

$$\min_{\theta, b, \mathbf{W}, \mathbf{c}} \mathcal{L} := \sum_{(\mathbf{x}^{(i)}, y^{(i)}) \in \mathcal{D}} ||y^{(i)} - \tilde{y}^{(i)}||_2^2. \tag{44}$$

Write out $\frac{\partial \mathcal{L}}{\partial \tilde{y}^{(i)}}$. You may use the terms $y^{(i)}$ and $\tilde{y}^{(i)}$.

Solution: $-2(y^{(i)} - \tilde{y}^{(i)})$ (1 point)

(d) Write out $\frac{\partial \tilde{y}^{(i)}}{\partial a_2^{(i)}}$. You may use the terms $a_2^{(i)}$.

Solution:

$$\frac{e^{-a_2^{(i)}}}{(e^{-a_2^{(i)}} + 1)^2} \tag{45}$$

(1 point)

(e) Write out $\frac{\partial \mathcal{L}}{\partial \mathbf{c}}$. You may use the terms $\frac{\partial \mathcal{L}}{\partial \tilde{y}^{(i)}}$, $\frac{\partial \tilde{y}^{(i)}}{\partial a_2^{(i)}}$, $\mathbf{W}$, $\mathbf{x}^{(i)}$, $\mathbf{c}$, $\theta, b, \mathbf{a}_1^{(i)}$ and $\delta[\cdot]$ as the indicator function.

Solution:

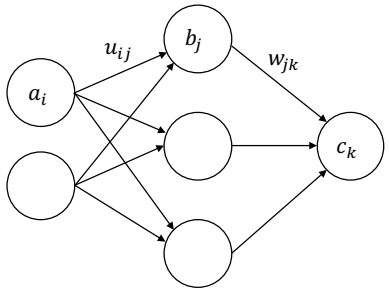
$$\sum_{(\mathbf{x}^{(i)},y^{(i)})\in\mathcal{D}} \frac{\partial\mathcal{L}}{\partial\tilde{y}^{(i)}} \cdot \frac{\partial\tilde{y}^{(i)}}{\partial a_2^{(i)}} \cdot \frac{\partial a_2^{(i)}}{\partial \mathbf{a}_1^{(i)}} \tag{46}$$

where, $\frac{\partial a_2^{(i)}}{\partial \mathbf{a}_1^{(i)}} = \theta \cdot \delta[\mathbf{a}_1^{(i)} > 0]$ (1 point)

(3 points) One for the summation, one for x, one for the chained terms.

20. **[8 points]** Backpropagation

Consider the deep net in the figure below consisting of an input layer, an output layer, and a hidden layer. The feed-forward computations performed by the deep net are as follows: every input a_i is multiplied by a set of fully-connected weights u_{ij} connecting the input layer to the hidden layer. The resulting weighted signals are then summed and combined with a bias e_j. This results in the activation signal $z_j = e_j + \sum_i a_i u_{ij}$. The hidden layer applies activation function g on z_j resulting in the signal b_j. In a similar fashion, the hidden layer activation signals b_j are multiplied by the weights connecting the hidden layer to the output layer w_{jk}, a bias f_k is added and the resulting signal is transformed by the output activation function g to form the network output c_k. The loss between the desired target t_k and the output c_k is given by the MSE: $E = \frac{1}{2}\sum_k (c_k - t_k)^2$, where t_k denotes the ground truth signal corresponding to c_k. Training a neural network involves determining the set of parameters $\theta = \{U, W, e, f\}$ that minimize E. This problem can be solved using gradient descent, which requires determining $\frac{\partial E}{\partial \theta}$ for all θ in the model.



(a) For $g(x) = \sigma(x) = \frac{1}{1+e^{-x}}$, compute the derivative $g'(x)$ of $g(x)$ as a function of $\sigma(x)$.

Solution:
(1 point) $g'(x) = \sigma(x)(1 - \sigma(x))$

(b) Compute $\frac{\partial E}{\partial w_{jk}}$. Use c_k, t_k, g', f_k, b_j, w_{jk}. Note the use of g' to simplify the expression.

Solution:
(2 points) $\frac{\partial E}{\partial w_{jk}} = (c_k - t_k) g'\left(f_k + \sum_j w_{jk} b_j\right) b_j$

(c) Compute $\frac{\partial E}{\partial f_k}$. Use c_k, t_k, g', f_k, b_j, w_{jk}. Note the use of g' to simplify the expression.

Solution:
(1 point) $\frac{\partial E}{\partial f_k} = (c_k - t_k) g'\left(f_k + \sum_j w_{jk} b_j\right)$

(d) Compute $\frac{\partial E}{\partial u_{ij}}$. Use c_k, t_k, g', f_k, b_j, w_{jk}, a_i. Note the use of g' to simplify the expression.

Solution:
(3 points) $\frac{\partial E}{\partial u_{ij}} = \sum_{k\in K} (c_k - t_k) g'\left(f_k + \sum_j w_{jk} b_j\right) w_{jk} g'(b_j) a_i$

(e) Compute $\frac{\partial E}{\partial e_j}$. Use c_k, t_k, g', f_k, b_j, w_{jk}, a_i. Note the use of g' to simplify the expression.

Solution:
(1 point) $\frac{\partial E}{\partial e_j} = \sum_{k \in K} (c_k - t_k) g' \left(f_k + \sum_j w_{jk} b_j \right) w_{jk} g'(b_j)$

21. **[13 points]** Deep Nets

(a) (2 points) You are working on an automated check-out kiosk for a supermarket, and are building a classifier for apples, bananas and oranges. Suppose your classifier obtains a training set error of 0.5%, and a test set error of 7%. Name two promising things to try to improve your classifier?

Solution: (1) Add regularization, (2) Get more training data

(b) (3 points) Suppose gradient descent in a deep network is taking excessively long to find a value of the parameters that achieves a small value for the cost function. Name three techniques that could help improve the convergence.

Solution: (1) Use better optimizers (Adam, Mini-Bach gradient descent), (2) Try better random initialization for the weights, (3) Try tuning the learning rate α.

(c) (1 point) Does it make sense to initialize all weights in a deep network to 0?

Solution: No. Symmetry breaking: If all weights are equal, their gradients will be the same as well, and the gradients will possibly vanish. Hence all neurons will learn the same feature.

(d) (1 point) State one advantage of linear rectified activation compared to logistic sigmoid activation.

Solution: Stable gradient, sparse activation, easy computation, etc..

(e) (2 points) State two roles of pooling layers in a CNN?

Solution: (a) they allow flexibility in the location of certain features in the image, therefore allowing non-rigid or rotated or scaled objects to be recognized, (b) they allow unions of features to be computed, e.g. blue eyes or green eyes (c) they reduce the output image size.

(f) (1 point) A convolutional neural network has 4 consecutive 3x3 convolutional layers with stride 1 and no pooling. How large is the support of (the set of image pixels which activate/influence) a neuron in the 3rd non-image layer of this network?

Solution: With a stride of 1, and a 3×3 filter, and no pooling, this means the outer ring of the image gets chopped off each time this is applied. Hence, this reduces the dimension from $n \times n$ to $((n-2) \times (n-2))$. We get, working backwards: $1 \times 1 \to 3 \times 3 \to 5 \times 5 \to 7 \times 7$ Thus, the support is $7 \times 7 = 49$ pixels.

(g) (1 point) What is the output image size resulting from applying three $5 \times 5 \times 3$ filters to a $32 \times 32 \times 3$ input image (no padding, stride is 1)?

Solution: The output size is $28 \times 28 \times 3$.

(h) (2 points) What are the padding and stride sizes that produce an output size of $32 \times 32 \times 3$ given a filter size of $5 \times 5 \times 3$ and input dimensions of $32 \times 32 \times 3$?

Solution: Stride size: 1, Padding size: 2.

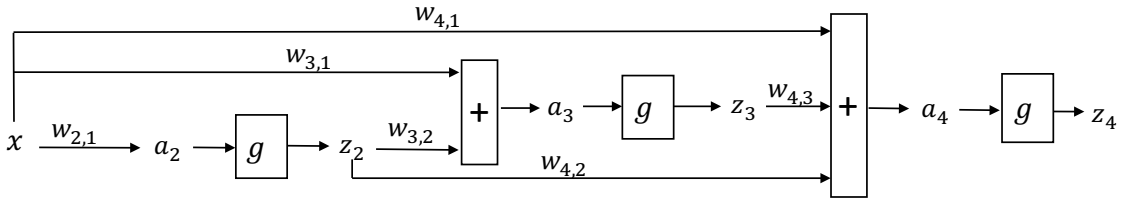
22. **[13 points]** Backpropagation

Consider the neural network given in the figure below. The network has a scalar input variable $x \in \mathbb{R}$ and a scalar target $t \in \mathbb{R}$ and is defined as follows:

$$z_j = \begin{cases} x, & \text{if } j = 1 \\ g(a_j) & \text{if } j \in \{2,3,4\} \text{ with } a_j = \sum_{i=1}^{j-1} w_{j,i} z_i \end{cases} \tag{47}$$

Suppose that the network is trained to minimize the L2 loss per sample, *i.e.*, $E = \frac{1}{2}(z_4 - t)^2$. The error gradient can be written as:

$$\frac{\partial E}{\partial w_{j,i}} = \delta_j z_i \tag{48}$$



(a) [2 pts] For $g(x) = \sigma(x) = \frac{1}{1+e^{-x}}$, compute the derivative $g'(x)$ of $g(x)$ as a function of $\sigma(x)$.

Solution: $g'(x) = \sigma(x)(1 - \sigma(x))$

(b) [2 pts] Compute δ_4 as a function of z_4, t and $g'(a_4)$.

Solution: $\delta_4 = (z_4 - t)g'(a_4)$

(c) [2 pts] Compute δ_3 as a function of δ_4, $w_{4,3}$ and $g'(a_3)$.

Solution: $\delta_3 = \delta_4 w_{4,3} g'(a_3)$

(d) [3 pts] Compute δ_2 as a function of δ_3, δ_4, $w_{3,2}$, $w_{4,2}$ and $g'(a_2)$.

Solution: $\delta_3 w_{3,2} g'(a_2) + \delta_4 w_{4,2} g'(a_2)$

(e) [4 pts] Write down a recursive formula for computing δ_j for $j \in \{2, \cdots, M-1\}$, as a function of δ_k, $w_{k,j}$ and $g'(a_j)$ for $k \in \{j+1, \cdots, M\}$.

Solution: $\delta_j = \sum_{k=j+1}^{M} \delta_k w_{k,j} g'(a_j)$

23. **[15 points]** Inference in Discrete Markov Random Fields

(a) Inference in Markov random fields amounts to finding the highest scoring configuration for a set of variables. Suppose we have two variables $x_1 \in \{0,1\}$ and $x_2 \in \{0,1\}$ and their local evidence functions $\theta_1(x_1)$ and $\theta_2(x_2)$ as well as a pairwise function $\theta_{1,2}(x_1, x_2)$. Using this setup, inference solves $\arg\max_{x_1,x_2} \theta_1(x_1) + \theta_2(x_2) + \theta_{1,2}(x_1, x_2)$. Using

$$\theta_1(x_1) = \begin{cases} 1 & \text{if } x_1 = 0 \\ 2 & \text{otherwise} \end{cases} \quad \theta_2(x_2) = \begin{cases} 1 & \text{if } x_2 = 0 \\ 2 & \text{otherwise} \end{cases} \quad \theta_{1,2}(x_1, x_2) = \begin{cases} -1 & \text{otherwise} \\ 2 & \text{if } x_1 = 0 \ \& \ x_2 = 1 \end{cases}$$

what is the integer linear programming formulation of the inference task. Make cost function and constraints explicit for the given problem, i.e., no general formulation.

Solution:

$$\max_{b} \begin{bmatrix} 1 \\ 2 \\ 1 \\ 2 \\ -1 \\ -1 \\ 2 \\ -1 \end{bmatrix}^{\top} \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{1,2}(0,0) \\ b_{1,2}(1,0) \\ b_{1,2}(0,1) \\ b_{1,2}(1,1) \end{bmatrix}$$

$$\text{s.t.} \quad \begin{cases} b_1(0), b_1(1), b_2(0), b_2(1), b_{1,2}(0,0), b_{1,2}(1,0), b_{1,2}(0,1), b_{1,2}(1,1) \in \{0,1\} \\ b_1(0) + b_1(1) = 1, b_2(0) + b_2(1) = 1 \\ b_{1,2}(0,0) + b_{1,2}(1,0) + b_{1,2}(0,1) + b_{1,2}(1,1) = 1 \\ b_1(0) = b_{1,2}(0,0) + b_{1,2}(0,1) \\ b_1(1) = b_{1,2}(1,0) + b_{1,2}(1,1) \\ b_2(0) = b_{1,2}(0,0) + b_{1,2}(1,0) \\ b_2(1) = b_{1,2}(0,1) + b_{1,2}(1,1) \end{cases}$$

(b) What is the solution (value and argument) to the program in part (a).

Solution:
argument: $b_1(0) = 1; b_1(1) = 0; b_2(0) = 0; b_2(1) = 1; b_{1,2}(0,0) = 0; b_{1,2}(1,0) = 0; b_{1,2}(0,1) = 1; b_{1,2}(1,1) = 0$; value: 5

(c) Why do we typically not use the integer linear program for reasonably sized MRFs and what other methods do you know to approximately solve MRFs. Name at least three other methods.

Solution:
too slow; alternatives: Linear programming relaxation, sampling, message passing

24. **[10 points]** Inference in Discrete Markov Random Fields

(a) Inference in Markov random fields amounts to finding the highest scoring configuration for a set of variables. Suppose we have two variables $x_1 \in \{0,1\}$ and $x_2 \in \{0,1\}$ and their local evidence functions $\theta_1(x_1)$ and $\theta_2(x_2)$ as well as pairwise function $\theta_{1,2}(x_1, x_2)$. Using this setup, inference solves $\arg\max_{x_1,x_2} \theta_1(x_1) + \theta_2(x_2) + \theta_{1,2}(x_1, x_2)$. Using

$$\theta_1(x_1) = \begin{cases} -1 & \text{if } x_1 = 0 \\ 1 & \text{otherwise} \end{cases} \qquad \theta_2(x_2) = \begin{cases} -1 & \text{if } x_2 = 0 \\ 1 & \text{otherwise} \end{cases}$$

$$\theta_{1,2}(x_1, x_2) = \begin{cases} 2 & \text{if } x_1 = 1 \;\&\; x_2 = 0 \\ 1 & \text{if } x_1 = 0 \;\&\; x_2 = 1 \\ -1 & \text{otherwise} \end{cases}$$

what is the integer linear programming (ILP) formulation of the inference task? Make cost function and constraints explicit for the given problem, i.e., do not use a general formulation.

Solution:

$$\max_b \left(\sum_{i=1}^{2} b_i(1) - b_i(0) \right) - b_{1,2}(1,1) + 2b_{1,2}(1,0) + b_{1,2}(0,1) - b_{1,2}(0,0)$$

$$\text{s.t.} \quad \begin{cases} b_1(0), b_1(1), b_2(0), b_2(1) \in \{0,1\} \\ b_{1,2}(0,0), b_{1,2}(1,0), b_{1,2}(0,1), b_{1,2}(1,1) \in \{0,1\} \\ b_1(0) + b_1(1) = 1, b_2(0) + b_2(1) = 1 \\ b_{1,2}(0,0) + b_{1,2}(1,0) + b_{1,2}(0,1) + b_{1,2}(1,1) = 1 \\ b_1(0) = b_{1,2}(0,0) + b_{1,2}(0,1) \\ b_1(1) = b_{1,2}(1,0) + b_{1,2}(1,1) \\ b_2(0) = b_{1,2}(0,0) + b_{1,2}(1,0) \\ b_2(1) = b_{1,2}(0,1) + b_{1,2}(1,1) \end{cases}$$

(5 pts)

(b) If the two variables instead took on values $x_1, x_2 \in \{0, 1, 2, 3\}$, how many constraints would the integer linear program have?

Solution: 24 domain constraints + 3 intra-region marginalization constraints + 8 inter-region marginalization constraints = 35 total constraints (1 pt)

(c) Let's say we wanted to use a different method to solve this inference problem. Can we use a dynamic programming method? Why or why not?

Solution:
Yes, we can - the graph represented by the problem is a tree. (2 pts)

(d) Name two other inference methods that may be more efficient than ILP, and name one advantage and one disadvantage for each.

Solution: Some possible answers:
Linear programming relaxation of the ILP - is no longer NP Hard and we have good solvers, but still may be inefficient for larger problems.
Message Passing: Efficient due to analytically computable sub-problems, but it takes special care to find global optimum (2 pts) Graph Cut: Have fast solvers, but requires potentials to have specific properties to work.

25. **[20 points]** Structured Prediction
We define a distribution $p(x_1, x_2)$ over two discrete random variables $x_1 \in \{0, 1\}$ and $x_2 \in \{0, 1\}$ as:

$$p(x_1, x_2) = \frac{1}{Z} \exp(-E(x_1, x_2)), \tag{49}$$

where $E : \{0, 1\}^2 \to \mathbb{R}$.

(a) [3 points] What is Z called? Give the expression of Z as a function of $E(x_1, x_2)$.

Solution: Z: partition function/normalization constant.

$$Z = \exp^{-E(0,0)} + \exp^{-E(0,1)} + \exp^{-E(1,0)} + \exp^{-E(1,1)} \tag{50}$$

(b) [1 point] We are interested in maximizing the likelihood $p(x_1, x_2)$ with respect to x_1 and x_2. Justify that this is equivalent to solving the program:

$$\min_{x_1 \in \{0,1\}, x_2 \in \{0,1\}} E(x_1, x_2). \tag{51}$$

Solution: The exp function is monotone increasing. Z does not depend on x_1 and x_2. Hence, maximizing $p(x_1, x_2)$ is equivalent to maximizing $-E(x_1, x_2)$.

(c) [8 points] $E(x_1, x_2)$ is an energy function modeled as:

$$E(x_1, x_2) = -(\theta_1(x_1) + \theta_2(x_2) + \theta_{1,2}(x_1, x_2)). \tag{52}$$

The potentials $\theta_1(x_1)$, $\theta_2(x_2)$ and $\theta_{1,2}(x_1, x_2)$ are given as follows:

$$\theta_1(x_1) = \begin{cases} 1 & \text{if } x_1 = 0 \\ 2 & \text{otherwise.} \end{cases} \tag{53}$$

$$\theta_2(x_2) = \begin{cases} 1 & \text{if } x_2 = 0 \\ 2 & \text{otherwise.} \end{cases} \tag{54}$$

$$\theta_{1,2}(x_1, x_2) = \begin{cases} -2 & \text{otherwise} \\ 1 & \text{if } x_1 = 0 \text{ and } x_2 = 1\ . \end{cases} \tag{55}$$

What is the Integer Linear Programming formulation of the inference task. Make cost function and constraints explicit for the given problem, *i.e.*, no general formulation.

Solution:

$$\max_b \left(\begin{bmatrix} 1 \\ 2 \\ 1 \\ 2 \\ -2 \\ -2 \\ 1 \\ -2 \end{bmatrix} \cdot \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{1,2}(0,0) \\ b_{1,2}(1,0) \\ b_{1,2}(0,1) \\ b_{1,2}(1,1) \end{bmatrix} \right) \tag{56}$$

$$\text{s.t.} \begin{cases} b_1(0), b_1(1), b_2(0), b_2(1), b_{1,2}(0,0), b_{1,2}(1,0), b_{1,2}(0,1), b_{1,2}(1,1) \in \{0,1\} \\ b_1(0) + b_1(1) = 1, b_2(0) + b_2(1) = 1 \\ b_{1,2}(0,0) + b_{1,2}(1,0) + b_{1,2}(0,1) + b_{1,2}(1,1) = 1 \\ b_1(0) = b_{1,2}(0,0) + b_{1,2}(0,1) \\ b_1(1) = b_{1,2}(1,0) + b_{1,2}(1,1) \\ b_2(0) = b_{1,2}(0,0) + b_{1,2}(1,0) \\ b_2(1) = b_{1,2}(0,1) + b_{1,2}(1,1) \end{cases} \tag{57}$$

(d) [3 points] What is the solution (value and argument) to the program in Eq. (51). Give the values of x_1^*, x_2^* and $E(x_1^*, x_2^*)$.

Solution: $x_1^* = 0$, $x_2^* = 1$, $E(x_1^*, x_2^*) = -4$

(e) [1 point] Name a disadvantage of using an Integer Linear Programming solver.

Solution: NP-complete. Slow for larger problems.

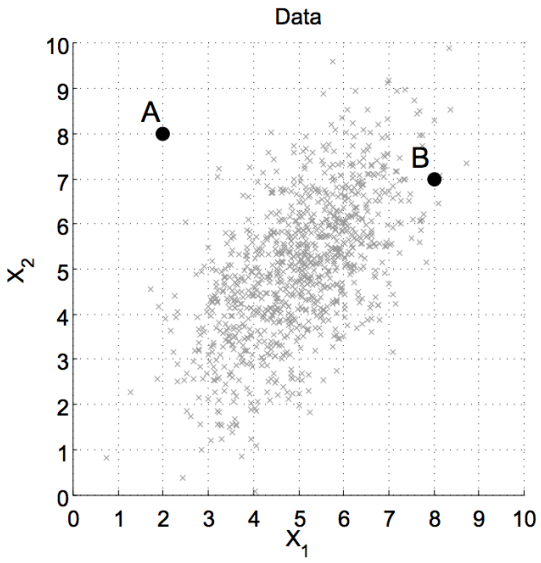
(f) [2 points] Is $E(x_1, x_2)$ sub-modular? Justify.

Solution: the potential $-\theta_{1,2}(x_1, x_2)$ is not sub-modular. $-(\theta_{1,2}(0,0)+\theta_{1,2}(1,1)) = 4$, $-(\theta_{1,2}(1,0) + \theta_{1,2}(0,1)) = 1$

(g) [2 points] Draw the Markov Random Field associated with the energy $E(x_1, x_2)$. Can we use a dynamic programming solver for the program above? Justify.

Solution: Yes, we can. The graph represented by the problem is a tree.
Graph: $\theta_1(x_1)$–$\theta_{1,2}(x_1, x_2)$–$\theta_2(x_2)$

26. **[7 points]** Principle Component Analysis (PCA)
Plotted in the figure below is a two dimensional data set drawn from a multivariate Normal distribution.



(a) [2 points] What is the mean of this distribution? Estimate the answer visually and round to the nearest integer.

Solution:

$$\mathbb{E}[X_1] = \mu_1 = 5$$
$$\mathbb{E}[X_2] = \mu_2 = 5$$

(b) [1 points] Circle the right answer. What would the off-diagonal co-variance $cov(X_1, X_2)$ be?

Solution:

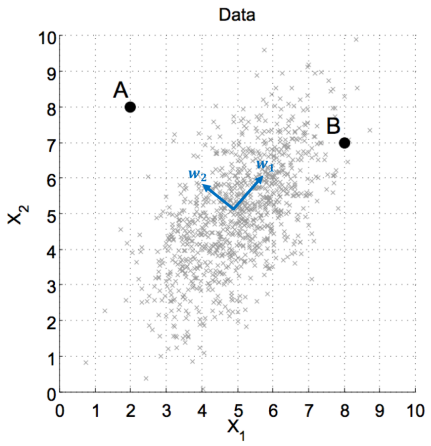
(a) negative
(b) positive **(X)**
(c) approximately zero

(c) [2 points] Let w_1 and w_2 be the directions of the first and second principal component. These directions define a change of basis,

$$Z_1 = (X - \mu)^T w_1, \quad (58)$$
$$Z_2 = (X - \mu)^T w_2. \quad (59)$$

Sketch and label w_1 and w_2 on the following figure. The arrows should originate from the mean of the distribution. You do not need to solve the SVD, instead visually estimate the directions. Present w_1 and w_2 as unit norm vectors.

Solution:



(d) [1 points] Circle the right answer. The co-variance $cov(Z_1, Z_2)$ is:

Solution:
(a) negative
(b) positive
(c) approximately zero (**X**)

(e) [1 points] Which point (A or B) would have the higher reconstruction error after projecting onto the first principal component direction w_1?

Solution:
(1) Point A (**X**)
(2) Point B

27. **[11 points]** k-means

(a) (3 points) Fill in the blanks in the following statement using the options provided in the boxes.

k-means is a/an **(a.)** _____ $\boxed{supervised, unsupervised, semi-supervised}$ learning algorithm used for **(b.)** _____ $\boxed{classification, clustering, regression}$. It aims to partition n observations into **(c.)**_____ $\boxed{n, d, k}$ clusters in which each observation belongs to the cluster with the nearest center that is the **(d.)** _____ $\boxed{median, mode, mean}$. The k-means objective function is related to the **(e.)** _____

$\boxed{Hamming, Euclidean, Manhattan}$ distance and the goal is to **(f.)** _____ $\boxed{maximize, minimize, orthogonalize}$ the objective function.

Choose the correct option for each blank from the box following it. Write down your answers below.

Solution:
(3 points)
a. unsupervised
b. clustering
c. k
d. mean
e. euclidean
f. minimize

(b) (4 points) Briefly describe the k-means algorithm. We want you to just write down the 4-5 lines. What is the run time, for n samples in d dimensions?

Solution:
(4 points)
Choose initial clusters $(C_1, \ldots, C_k)$.
Repeat until convergence:
(Recenter.) Set $\mu_j := \text{mean}(C_j), \forall j \in (1, \ldots, k)$.
(Reassign). Update $C_j := x_i : \mu(x_i) = \mu_j, \forall j \in (1, \ldots, k)$
(break ties arbitrarily)
Runtime: $(O)(nkd)$

(c) (4 points) In class, you saw that the k-means objective was defined using the squared L2-norm, i.e. $\sum_{i=1}^{n} \min_j ||x_i - \mu_j||_2^2$. What are optimal cluster centers with this objective function?
Now if we define a new objective function using the L1-norm instead, i.e. $\sum_{i=1}^{n} \min_j ||x_i - \mu_j||_1$, what would be the expression for the optimal cluster centers? **Derive** your answer assuming $x_i - \mu_j \neq 0, \forall i, j$, i.e. the cluster centers don't overlap with the given input datapoints.

Solution:
(4 points)
The optimal solution centers comprise of the mean of the points nearest to that center.
If the objective function is defined using the L1-norm, then, following the derivation of k-means as done in the class, let's take the gradient w.r.t μ_1, and
define $\mu(x_i) \in (\mu_1, \ldots, \mu_k)$ closest to x_i

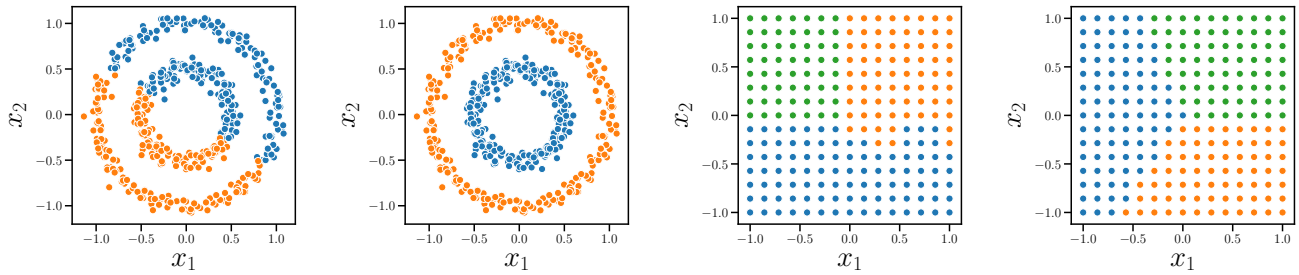
$$\nabla_{\mu_1} \sum_{i=1}^{n} ||x_i - \mu_j||_1 = \nabla_{\mu_1} \Big(\sum_{\substack{i \in (1,\ldots,n) \\ \mu(x_i) = \mu_1}} ||x_i - \mu_1||_1 + \sum_{\substack{i \in (1,\ldots,n) \\ \mu(x_i) \neq \mu_1}} ||x_i - \mu(x_i)||_1 \Big)$$

$$= \sum_{\substack{i \in (1,\ldots,n) \\ \mu(x_i) = \mu_1 \\ x_i < \mu 1}} 1 + \sum_{\substack{i \in (1,\ldots,n) \\ \mu(x_i) = \mu_1 \\ x_i > \mu_1}} -1$$

This would be equal to zero only if there are equal number of points on either side of μ_1, hence μ_1 would be the median. Note: since no data point ever lies over the centers, we could piecewise differentiate the objective function.

28. **[15 points]** K-means and mixture models

(a) [4 points] Below are four cluster visualizations:



(a) (b) (c) (d)

Please indicate which visualizations are generated from the K-means algorithm in the visualized feature space, $x = (x_1, x_2)$, and why.

Hint: Our K-means algorithm optimizes the following:

$$\min_{\mu} \min_{r} \sum_{i} \sum_{k} \frac{1}{2} r_{ik} \left\| x^{(i)} - \mu_k \right\|_2^2 \quad \text{s.t.} \quad \begin{cases} r_{ik} \in \{0,1\} & \forall i, k \\ \sum_{k=1}^{K} r_{ik} = 1 & \forall i \end{cases} .$$

Solution:
(a) and (d), K-means uses a squared loss per cluster.
(b) and (c)'s decision boundaries cannot be generated with a squared-loss.

(b) [8 points] Recall, a mixture model is given by

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \cdot p(\mathbf{x}|z=k), \tag{60}$$

where $z \in \{1, 2, \ldots, K\}$.
Suppose the vector $\mathbf{x}$ is partitioned into two parts $\mathbf{x} := [\mathbf{x_a}, \mathbf{x_b}]$.

Show that $p(\mathbf{x}_b|\mathbf{x}_a)$ is also a mixture distribution, *i.e.*, the conditional density has the form

$$p(\mathbf{x}_b|\mathbf{x}_a) = \sum_{k=1}^{K} \hat{\pi}_k \cdot p(\mathbf{x}_b|\mathbf{x}_a, z=k).$$

To this end, provide an expression for $\hat{\pi}_k$ in terms of $p(\mathbf{x}), p(\mathbf{x_a}|z)$ and $p(z=k)$.
Hint: $p(\mathbf{x_b}, z|\mathbf{x_a})$ may be useful.

Solution:
(Question Credit: From Bishop's book question 9.10)

$$p(\mathbf{x}_b|\mathbf{x}_a) = \sum_{k=1}^{K} p(\mathbf{x}_b, z=k|\mathbf{x}_a) \tag{61}$$

$$= \sum_{k=1}^{k} p(\mathbf{x}_b, |z=k, \mathbf{x}_a) \cdot p(z=k|\mathbf{x}_a) \tag{62}$$

$$= \sum_{k=1}^{k} p(\mathbf{x}_b, |z=k, \mathbf{x}_a) \cdot \underbrace{p(\mathbf{x}_a|z=k)p(z=k)/p(\mathbf{x})}_{\hat{\pi}_k} \tag{63}$$

(c) [3 points] How are K-means and Gaussian Mixture Model related? (There are three conditions.) Explain them shortly.

Solution:

i. Same variance for all Gaussian mixtures.

ii. Change to uniform distribution for the latent variable. $\pi_k = 1/K \ \forall k$

iii. Zero-temperature (hard-version) of the Gaussian Mixture Model.

29. **[16 points]** Gaussian Mixture Models & K-Means
Consider a Gaussian mixture model with K components ($k \in \{1, \ldots, K\}$), each having mean μ_k, variance σ_k^2, and mixture weight π_k. Further, we are given a dataset $\mathcal{D} = \{x_i\}$, where $x_i \in \mathbb{R}$. We also use z_{ik} to denote the latent variables.

(a) What is the log-likelihood of the data according to the Gaussian Mixture Model? (use μ_k, σ_k, π_k, K, x_i and $\mathcal{D}$). Don't use any abbreviations.

Solution:
(1 point)

$$\sum_{x_i \in \mathcal{D}} \log \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(x_i - \mu_k)^2}{2\sigma_k^2}\right) \tag{64}$$

(b) Assume $K = 1$, find the maximum likelihood estimate for the parameters $(\mu_1, \sigma_1^2, \pi_1)$.

Solution:
(3 points)

$$\mu_1 = \frac{1}{|D|} \sum_{x_i \in \mathcal{D}} x_i \tag{65}$$

$$\sigma_1^2 = \frac{1}{|D|} \sum_{x_i \in \mathcal{D}} (x_i - \mu_1)^2 \tag{66}$$

$$\pi_1 = 1 \tag{67}$$

(c) What is the probability distribution on the latent variables, *i.e.*, what is the distribution $p(z_{i,1}, z_{i,2}, \ldots, z_{i,K})$ underlying Gaussian mixture models. Also give its name.

Solution:
(2 points) Multinomial distribution; $p(z_{i,1}, z_{i,2}, \ldots, z_{i,K}) = \prod_{k=1}^{K} \pi_k^{z_{ik}}$

(d) For general K, what is the posterior probability $p(z_{ik} = 1|x^{(i)})$? To simplify, wherever possible, use $\mathcal{N}(x_i|\mu_k, \sigma_k)$, a Gaussian distribution over $x_i \in \mathbb{R}$ having mean μ_k and variance σ_k^2.

Solution:
(1 point)

$$p(z_{ik} = 1|x_i) = \frac{\pi_k \mathcal{N}(x_i|\mu_k, \sigma_k)}{\sum_{\hat{k}=1}^{K} \pi_{\hat{k}} \mathcal{N}(x_i|\mu_{\hat{k}}, \sigma_{\hat{k}})} \tag{68}$$

(e) How are kMeans and Gaussian Mixture Model related? (There are three conditions)

Solution:
(3 points)
1) Same variance for all Gaussian mixtures.
2) Change to uniform distribution for the latent variable. $\pi_k = 1 \ \forall k$
3) Zero-temperature (hard-version) of the Gaussian Mixture Model.

(f) Show that the objective for kMeans and Gaussian Mixture Model are equivalent under the conditions you provided in the previous part (e).

Solution:
(6 points)

Assume σ's are fixed and equal for all clusters, then the objective for GMM is

$$\min_{\mu} -\sum_{x_i \in \mathcal{D}} \log \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp -\frac{(x_i - \mu_k)^2}{2\sigma_k^2} \tag{69}$$

Equivalent to

$$\min_{\mu} -\sum_{x_i \in \mathcal{D}} \log \sum_{k=1}^{K} \exp -(x_i - \mu_k)^2 \tag{70}$$

Apply log-sum approximation and taking ϵ to 0.

$$\min_{\mu} -\sum_{x_i \in \mathcal{D}} \epsilon \log \sum_{k=1}^{K} \exp\left(-(x_i - \mu_k)^2/\epsilon\right) \tag{71}$$

Apply L'Hospital's rule, let $F_k = (x - \mu_k)^2$

$$\lim_{\epsilon \to 0} \sum_{k=1}^{K} \frac{\exp(-F_k/\epsilon)}{\sum_{\hat{k}} \exp(-F_{\hat{k}}/\epsilon)} \cdot F_k \tag{72}$$

Move the limit inside and obtain

$$\sum_{k=1}^{K} \mathbf{1}[k = \arg\max_k F_k] F_k \tag{73}$$

Finally, the k-means objective

$$\sum_{x_i \in \mathcal{D}} \sum_{k=1}^{k} \mathbf{1}[k = \arg\max_k F_k](x_i - \mu_k)^2 \tag{74}$$

30. **[10 points]** Generative models
In class, you have studied various generative models including GMMs, VAEs, and GANs. In this problem we will analyze the links between them.

(a) (1 point) GMMs parameters are learned by maximizing the log-likelihood of the data. Suppose we are given N samples x_i ($i = 1, 2, \ldots, N$) from a random variable $X \sim P$ and we want to fit a Gaussian mixture model with K components. Write the log-likelihood of the data in terms of the mixture parameters θ=(μ_k, σ_k, π_k for $k = 1, 2, \ldots, K$).

Solution:

$$\text{log likelihood} = \sum_{i=1}^{N} \log \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp(-\frac{(x_i - \mu_k)^2}{2\sigma_k^2}) \tag{75}$$

(b) (1 point) Other generative models minimize a distance. E.g. GANs may minimize the KL divergence between a ground truth distribution P and a generated distribution G. Write the KL divergence between P and G.

Solution:

$$KL(P||G) = E_{X\sim P}[\log \frac{P(X)}{G(X)}] \tag{76}$$

(c) (5 points) Let the distribution represented by the GMM be M. We find a solution $\hat{\theta}$ for the parameters as

$$\hat{\theta} = \text{argmax}_\theta \sum_{i=1}^{N} \log M(x_i) \tag{77}$$

Show that as $N \to \infty$, this formulation is equivalent to minimizing the divergence $KL(P||M)$, where P is the ground truth distribution.

Solution:

$$\lim_{N\to\infty} \text{argmax}_\theta \sum_i \log M(x_i) = \lim_{N\to\infty} \text{argmax}_\theta \frac{1}{N} \sum_i \log M(x_i) \tag{78}$$

$$= \text{argmax}_\theta E_{x\sim P}[\log M(x)] \tag{79}$$

(average over infinite samples)

$$= \text{argmax}_\theta E_{x\sim P}[\log M(x)] - E_{x\sim P}[\log P(x)] \tag{80}$$

(subtracting a constant that does not depend on θ)

$$= \text{argmax}_\theta E_{x\sim P}[\log \frac{M(x)}{P(x)}] \tag{81}$$

$$= \text{argmax}_\theta - KL(P||M) \tag{82}$$

(d) (3 points) For GMMs, we know the model of the distribution. Therefore, we can sample from it easily. GANs take a different approach where they transform a known distribution into the data distribution. Suppose we have a uniform distribution $U[0, 1]$. We want to map this distribution using a function f to another distribution whose pdf is p and cdf is C. Find f in terms of p and C.

Solution: For $x \sim U$, we want

$$P(f(x) \leq y) = C(y) \tag{83}$$

Alternatively,

$$P(x \leq f^{-1}(y)) = C(y) \tag{84}$$

For the uniform distribution, we know that

$$P(x \leq y) = y \tag{85}$$

Therefore, $f = C^{-1}$.

31. **[7 points]** Variational Autoencoder (VAE)

(a) Show that the KL divergence between two discrete distributions $p(x)$ and $q(x)$ defined on the domain $x \in X$ is non-negative.

Solution:
(3 points)

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln p(x)/q(x)$$

Via Jensen's inequality:

$$\sum_{x \in X} p(x) \ln p(x)/q(x) > -\ln(\sum_{x \in X} q(x)) = -ln(1) = 0$$

(b) What is the reparameterization trick? Why do we need it in VAEs?

Solution:
(2 points) Sampling from $\mathcal{N}(\mu, \sigma^2)$ is equivalent to sample obtained from $\mu + \sigma \cdot \epsilon$, where ϵ is drawn from $\mathcal{N}(0, 1)$. Use this to backpropagate to the encoder.

(c) Given a code snippet of the `_sample_z` function from mp3, which builds the graph to perform the reparametrization trick. Circle the line that is incorrect and explain why it is wrong. You may assume all the shapes and syntax are correct.

```
def _sample_z(self, z_mean, z_log_var):
    """
    Sample z using reparametrization trick.

    Args:
        z_mean (tf.Tensor): The latent mean,
            tensor of dimension (None, 2)
        z_log_var (tf.Tensor): The latent log variance,
            tensor of dimension (None, 2)
    Returns:
        z (tf.Tensor): Random z sampled of dimension (None, 2)
    """
    eps = np.random.randn(self.z_shape[0], self.z_shape[1])
    z = z_mean + (tf.exp(z_log_var / 2) * eps)
    return z
```

Solution:
(2 points) Incorrect, as a numpy eps samples once during the graph construction phase. Does not generate a new random sample each time.

32. **[15 points]** Variational Auto-encoders for Image Generation

(a) What is the expression for the Kullback-Leibler divergence $D_{KL}(p, q)$ between two 1-dimensional distributions p and q defined over the domain of real numbers x?

Solution:
$D_{KL}(p, q) = \int_{x=-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

(b) What is the expression for a 1-dimensional Gaussian distribution defined over the domain of real numbers x which has mean μ and standard deviation σ, often abbreviated $\mathcal{N}(x \mid \mu, \sigma)$.

Solution:

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

(c) Explain the manifold assumption underlying variational auto-encoders.

Solution:
We assume $p(x|z)$ to be a simple distribution, where z is a low-dimensional manifold.

(d) Show the following identity:

$$\int_z q(z|x) \log \frac{p(x,z)}{q(z|x)} dz + \int_z q(z|x) \log \frac{q(z|x)}{p(z|x)} dz = \log p(x)$$

Solution:

$$\begin{aligned}
\log p(x) &= \int_z q(z|x) \log p(x) dz \\
&= \int_z q(z|x) \log \frac{p(x,z)}{p(z|x)} dz \\
&= \int_z q(z|x) \log \left(\frac{p(x,z)}{q(z|x)} \cdot \frac{q(z|x)}{p(z|x)} \right) dz \\
&= \int_z q(z|x) \log \frac{p(x,z)}{q(z|x)} dz + \int_z q(z|x) \log \frac{q(z|x)}{p(z|x)} dz
\end{aligned}$$

(e) Given your derivation in the previous part, provide a lower bound on the log-probability $\log p(x)$ and explain.

Solution:
The second term on the RHS is a KL-divergence which is non-negative. Hence we obtain the lower bound

$$\mathcal{L} = \int_z q(z|x) \log \frac{p(x,z)}{q(z|x)} dz$$

33. **[14 points]** Variational Autoencoders (VAEs)

Suppose we are given a dataset with N data points $\{x_i\}_{i=1}^N$ where each of the data points is a D-dimensional vector. We use VAEs to learn the distribution of the data. Let z denote the unobserved latent variable. We refer to the approximated posterior $q_\phi(z|x)$ as the encoder and to the conditional distribution $p_\theta(x|z)$ as the decoder. Use the above notations to answer the following questions.

(a) (2 points) The empirical lower bound (ELBO) of $p_\theta(x_i)$ is

$$\mathcal{L}(\theta, \phi, x_i) = -D_{KL}(q_\phi(z|x_i)||p(z)) + E_{q_\phi(z|x_i)}\Big[\log p_\theta(x_i|z)\Big]$$

Write down the minimization program that VAEs solve in terms of $p_\theta(\cdot)$ and $q_\phi(\cdot)$ given the dataset $\mathbf{x} = \{x_i\}_{i=1}^N$.

Solution: (2 points)

$$\min_{\theta,\phi} \sum_i \left\{ D_{KL}(q_\phi(z|x_i)||p(z)) + E_{q_\phi(z|x_i)}\Big[-\log p_\theta(x_i|z)\Big] \right\}$$

(1 point) Write down the reconstruction error in the loss function of VAEs in your

previous answer, again given the dataset $\mathbf{x} = \{x_i\}_{i=1}^N$.

Solution: (1 point)

$$\sum_i E_{q_\phi(z|x_i)}\Big[-\log p_\theta(x_i|z)\Big]$$

(1 point) Write down the formula to compute the reconstruction error empirically by drawing M samples from the distribution $q_\phi(z|x_i)$. Denote these M samples as $z_{i,m}$, where $m = 1, 2, ..., M$.

Solution: (1 point)

$$\sum_i \frac{1}{M} \sum_{m=1}^{M} \Big[-\log p_\theta(x_i|z = z_{i,m})\Big]$$

(b) (3 points) Let $f(z_{i,m}) \in \mathbf{R}^D$ be the reconstructed sample with respect to $z_{i,m}$, which is the output of the decoder. What is the empirical reconstruction error if we assume $p_\theta(x_i|z)$ to be a Gaussian distribution $\mathcal{N}(f(z), \sigma^2\mathbf{I})$, where σ is a constant and $\mathbf{I}$ is the D-by-D identity matrix (simplify as much as possible)?

Solution:
(3 points)

$$\sum_i \frac{1}{M} \sum_{m=1}^{M} \Big[-\log \frac{1}{\sqrt{2\pi}\sigma^D} + \frac{||x_i - f(z_{i,m})||_2^2}{2\sigma^2}\Big]$$

(c) (4 points) Now consider all the data points x_i to be binary, i.e., $\forall i$, $x_i \in \{0, 1\}^D$. If we want to have the empirical reconstruction error to be the cross entropy loss, what should we assume $p_\theta(x_i|z)$ to be? What is the name of the distribution? Let the output of the decoder be $g = f(z) \in [0, 1]^D$, where the values are all between 0 and 1. If you need, use $x_i^{(d)}$ to denote the d-th element in the vector x_i.

Solution: (3 points)
Assume

$$\log p_\theta(x_i|z) = \sum_{d=1}^{D} x_i^{(d)} \log g^{(d)} + (1 - x_i^{(d)}) \log(1 - g^{(d)})$$

(1 point)
Multivariate Bernoulli distribution.

(d) (3 points) The following shows a code snippet of the `_sample_z` function from mp10, which builds the graph to perform the reparametrization trick. Is this implementation correct? If not, circle each of the place that you think is incorrect and explain your reason. You may assume all the shapes and syntax are correct.

```
def _sample_z(self, z_mean, z_log_var):
    """
    Sample z using the reparametrization trick.
    Args:
        z_mean (tf.Tensor): The latent mean,
            tensor of dimension (None, 2)
        z_log_var (tf.Tensor): The latent log variance,
            tensor of dimension (None, 2)
    Returns:
```

```
        z (tf.Tensor): Random z sampled of dimension (None, 2)
    """
    eps = np.random.randn(self.z_shape[0], self.z_shape[1])
    z = z_mean * tf.sqrt(z_log_var) * eps
    return z
```

Solution: (3 points)

`eps = np.random.randn(self.z_shape[0],self.z_shape[1])`

is incorrect. The numpy eps is only sampled once during the graph construction phase. Should call the tensorflow random number generator instead such that a new random sample is generated each time running the computational graph.

`z = z_mean * tf.sqrt(z_log_var) * eps`

is incorrect. Should be

`z = z_mean + tf.exp(z_log_var/2.0)*eps`

.

34. **[20 points]** Variational Autoencoders (VAEs)
We use VAEs to learn the distribution of the data, x. Let z denote the unobserved latent variable. We refer to the approximated posterior $q_\phi(z|x)$ as the encoder and to the conditional distribution $p_\theta(x|z)$ as the decoder. Use these names to answer the following questions.

    (a) [4 points] We are interested in modeling data, $x \in \{0,1\}^G$. Hence, we choose $p_\theta(x|z)$ to follow G independent Bernoulli distributions. Recall, a Bernoulli distribution has a probability density function of

$$P(k) = \begin{cases} 1-p & \text{if } k = 0 \\ p & \text{if } k = 1 \end{cases}.$$

    Write down the explicit form for $p_\theta(x|z)$. Use $\hat{y}_j$ to denote the $j^{\text{th}} \in [1, G]$ dimension of the decoder's output.

Solution: $p_\theta(x|z) = \prod_{j=1}^{G} \hat{y}_j^x \cdot (1-\hat{y}_j)^{1-x}$

    (b) [4 points] We further assume that $z \in \mathbb{R}^2$ and that $q_\phi(z|x)$ follows a multi-variate Gaussian distribution with an identity covariance matrix. What is the output dimension of the encoder and why?

Solution: The output dimension is 2, we only need to model the mean, μ.

(c) [8 points] Recall, the evidence lower bound (ELBO) of the log likelihood, $\log p_\theta(x)$, is

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)). \tag{86}$$

We can also write the ELBO as

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z) + \log p(z) - \log(q_\phi(z|x)]. \tag{87}$$

Practically, will training a VAE using the formulation in Eq. (86) be the same as the one in Eq. (87)? If not, why use one formulation over another?

Solution: (1) In practice the two formulation will not be the same. This is because the training algorithm for VAE requires sampling z. In the first formulation, D_{KL} uses an analytic form and do not estimate the KL-term from samples.
(2) One might choose to use the second formulation when the analytic form of D_{KL} do not exist or is difficult to compute.

(d) [4 points] Observe that the ELBO in Eq. (86) works for any q_ϕ distribution. Is it a good idea to choose $q_\phi(z|x) := \mathcal{N}(\mathbf{0}, \mathbf{I})$? In other words, why is an encoder necessary?

Solution: No. Ideally, q_ϕ should approximate $p_\theta(z|x)$ for the ELBO to be tight. Otherwise, samples from $q(z)$ may result in $p_\theta(x|z) \approx 0$, which may require many samples to get a gradient signal.

35. **[8 points]** Generative Adversarial Network (GAN)

(a) What is the key difference between VAE and GAN?

Solution:
(1 point) VAE makes an explicit assumption on $p(X)$, GAN is not a probabilistic model.

(b) What is the cost function for classical GANs? Use $D_w(x)$ as the discriminator and $G_\theta(x)$ as the generator.

Solution:
(1 point)

$$\max_\theta \min_w -\sum_x \log D_w(x) - \sum_z \log(1 - D_w(G_\theta(z)))$$

(c) Assume arbitrary capacity for both discriminator and generator. In this case we refer to the discriminator using $D(x)$, and denote the distribution on the data domain induced by the generator via $p_G(x)$. State an equivalent problem to the one asked for in part (a), by using $p_G(x)$.

Solution:
(1 point)

$$\max_{p_G} \min_D -\int_x p_{\text{data}}(x) \log D(x) dx - \int_x p_G(x) \log(1 - D(x)) dx \tag{88}$$

(d) Assume arbitrary capacity, derive the optimal discriminator $D^*(x)$ in terms of $p_{data}(x)$ and $p_G(x)$.
You may need the Euler-Lagrange equation:

$$\frac{\partial L(x, D, \dot{D})}{\partial D} - \frac{d}{dx}\frac{\partial L(x, D, \dot{D})}{\partial \dot{D}} = 0$$

where $\dot{D} = \partial D/\partial x$.

Solution:
(2 points) Use the Euler-Lagrange formalism which says that the stationary point of $S(D) = \int_x L(x, D, \dot{D})dx$ can be obtained from

$$\frac{\partial L(x, D, \dot{D})}{\partial D} - \frac{d}{dx}\frac{\partial L(x, D, \dot{D})}{\partial \dot{D}} = 0$$

to demonstrate that a stationary point of GANs (use Eq. 89) is obtained for $p_D = p_G$. Note that $\dot{D} = \partial D/\partial x$.

$$\frac{\partial L(x, D, \dot{D})}{\partial D} = -\frac{p_{\text{data}}}{D} + \frac{p_G}{1 - D} = 0$$

Consequently:

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

(e) Assume arbitrary capacity and an optimal discriminator $D^*(x)$, show that the optimal generator, $G^*(x)$, generates the distribution $p^*_G = p_{data}$, where $p_{data}(x)$ is the data distribution
You may need the Jensen-Shannon divergence:

$$\text{JSD}(p_{\text{data}}, p_G) = \frac{1}{2}D_{KL}(p_{\text{data}}, M) + \frac{1}{2}D_{KL}(p_G, M) \quad \text{with} \quad M = \frac{1}{2}(p_{\text{data}} + p_G)$$

Solution:
(2 points)

$$\text{JSD}(p_{\text{data}}, p_G) = \frac{1}{2}D_{KL}(p_{\text{data}}, M) + \frac{1}{2}D_{KL}(p_G, M) \quad \text{with} \quad M = \frac{1}{2}(p_{\text{data}} + p_G)$$

Therefore:

$$\begin{aligned} & -\int_x p_{\text{data}}(x) \log D^*(x) + p_G(x) \log(1 - D^*(x))dx \\ = \; & -\int_x p_{\text{data}}(x) \log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)} + p_G(x) \log \frac{p_G(x)}{p_{\text{data}}(x) + p_G(x)}dx \\ = \; & -2\,\text{JSD}(p_{\text{data}}, p_G) + \log(4) \end{aligned}$$

Hence:

$$p_{\text{data}} = p_G$$

(f) What is the optimal discriminator $D^*(x)$, assuming arbitrary capacity and optimal generator?

Solution:
(1 point)
$D^*(x) = 0.5$

36. **[9 points]** Generative Adversarial Nets (GANs) for Image Generation

(a) Explain the intuition underlying generative adversarial nets (GANs).

Solution:
Two-player game with generator producing artificial samples and discriminator trying to tell apart artificial samples from real data.

(b) What is the relation between the cost function for GANs, i.e.,

$$\max_{\theta} \min_{w} -\sum_{x} \log D_w(x) - \sum_{z} \log(1 - D_w(G_\theta(z)))$$

and the following program:

$$\max_{p_G} \min_{D} -\int_x p_{\text{data}}(x) \log D(x) dx - \int_x p_G(x) \log(1 - D(x)) dx \tag{89}$$

Solution:
We assume an arbitrary capacity discriminator $D(x)$ (function over the domain x) and an arbitrary capacity generator given by the probability distribution $p_G(x)$. Instead of using samples we integrate over the entire data domain. The distribution generated by the generator $G_\theta(z)$ is referred to using $p_G(x)$.

(c) Use the Euler-Lagrange formalism which says that the stationary point of $S(D) = \int_x L(x, D, \dot{D}) dx$ can be obtained from

$$\frac{\partial L(x, D, \dot{D})}{\partial D} - \frac{d}{dx} \frac{\partial L(x, D, \dot{D})}{\partial \dot{D}} = 0$$

to demonstrate that a stationary point of GANs (use Eq. 89) is obtained for $p_D = p_G$. Note that $\dot{D} = \partial D / \partial x$. Hints: solve for the optimal discriminator first. You may need the Jensen-Shannon divergence:

$$\text{JSD}(p_{\text{data}}, p_G) = \frac{1}{2} D_{KL}(p_{\text{data}}, M) + \frac{1}{2} D_{KL}(p_G, M) \quad \text{with} \quad M = \frac{1}{2}(p_{\text{data}} + p_G)$$

Solution:

$$\frac{\partial L(x, D, \dot{D})}{\partial D} = -\frac{p_{\text{data}}}{D} + \frac{p_G}{1 - D} = 0$$

Consequently:

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

Therefore:

$$\begin{aligned} & -\int_x p_{\text{data}}(x) \log D^*(x) + p_G(x) \log(1 - D^*(x)) dx \\ = & -\int_x p_{\text{data}}(x) \log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)} + p_G(x) \log \frac{p_G(x)}{p_{\text{data}}(x) + p_G(x)} dx \\ = & -2 \,\text{JSD}(p_{\text{data}}, p_G) + \log(4) \end{aligned}$$

Hence:

$$p_{\text{data}} = p_G$$

37. **[20 points]** Generative Adversarial Nets (GANs)

(a) [2 points] What is the key difference between a VAE and a GAN?

Solution:
VAE makes an explicit assumption on $p(X)$, GAN is not a probabilistic model.

(b) [2 points] Below are images sampled from a VAE and a GAN trained on the CelebA dataset.

(I) 

(II) 

Which row of samples are generated from a GAN and why?

Solution:
(I) is from GAN. Images in (II) are very smooth, typical for VAE.

(c) [4 points] The loss function for a classical GAN is

$$\max_{\theta} \min_{w} -\sum_{x \in \mathcal{D}} \log D_w(x) - \sum_{z \sim \mathcal{N}(0,1)} \log(1 - D_w(G_\theta(z))), \tag{90}$$

where D_w denotes the discriminator, G_θ denotes the generator, and $\mathcal{D} = \{(x)\}$ denotes the dataset.

What is an issue when there is a poor generator and a good discriminator? What is a common heuristic to counter this issue?

Solution:
A good discriminator and poor generator results in almost no gradient.
Instead solve for:

$$\min_{\theta} -\sum_{z} \log(D_w(G_\theta(z))$$

(d) [8 points] Please write down the mini-batch gradient descent training algorithm for optimizing the program given in Eq. (90).

Solution:

Algorithm 1: Minibatch gradient descent training

initialization;
for *number of training iterations* **do**

- Sample minibatch, $\mathcal{Z}$, of z from $\mathcal{N}(0, 1)$.
- Sample minibatch, $\mathcal{X}$, of x from data $\mathcal{D}$ uniformly.
- Update D_w:

$$w \mathrel{+}= \nabla_w \left(\sum_{(x \in \mathcal{X})} \log(D_w(x)) + \sum_{z \in \mathcal{Z}} \log(1 - D_w(G_\theta(z))) \right)$$

- Update G_θ:

$$\theta \mathrel{+}= -\nabla_\theta \left(\sum_{z \in \mathcal{Z}} \log(1 - D_w(G_\theta(z))) \right)$$

end

(e) [4 points] To theoretically analyze the cost function in Eq. (90), we consider the following program:

$$\max_{p_G} \min_{D} -\int_x p_{\text{data}}(x) \log D(x) dx - \int_x p_G(x) \log(1 - D(x)) dx. \tag{91}$$

Write down the meaning and relation to Eq. (90) for each of the following symbols: p_{data}, p_G, D, and G.

Solution:

- p_{data} = The distribution of x where the samples are uniform selected from $\mathcal{D}$
- p_G = The distribution of x induced from $G(z)$, where $z \sim \mathcal{N}(0, 1)$.
- D is the discriminator with arbitrary capacity.
- G is the generator with arbitrary capacity.

38. **[10 points]** Markov Decision Processes (blue color: rewards; red color: decision probabilities)

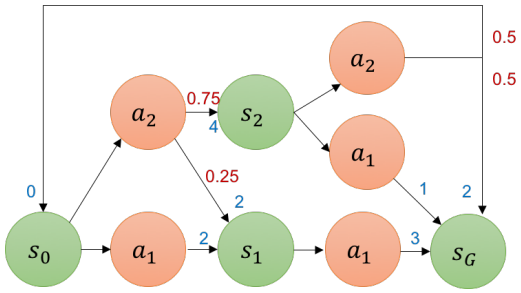


Figure 4: MDP for this problem. Start state is s_0.

(a) Explain the difference between a deterministic and a stochastic Markov decision process. Is the given MDP stochastic or deterministic. Explain your answer.

Solution:
(3/2 points)

deterministic: performing an action always results in a deterministic transition; stochastic: performing an action may lead the agent to one of a few possible states; the given MDP is stochastic

(b) What are three mechanisms to find the optimal policy π^* for a given MDP?

Solution:
(3/2 points)

Exhaustive search, policy iteration, value iteration

(c) For the policy $\pi(s_0) = a_1$, $\pi(s_1) = a_1$, what is the policy graph and the resulting value function $V^\pi(s_1)$ and $V^\pi(s_0)$?

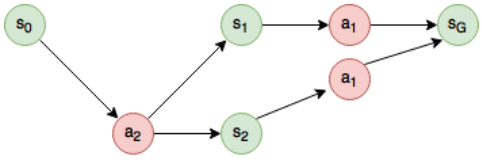
Solution:
(3/2 points)

Policy graph: 

$V^\pi(s_1) = 3 \quad V^\pi(s_0) = 5$

(d) For the policy $\pi(s_0) = a_2$, $\pi(s_2) = a_1$, what is the policy graph and the resulting value function $V^\pi(s_2)$, $V^\pi(s_1)$ and $V^\pi(s_0)$?

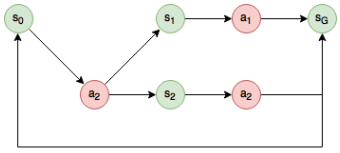
Solution:
(4/2 points)

Policy graph: 

$V^\pi(s_1) = 3 \quad V^\pi(s_2) = 1 \quad V^\pi(s_0) = 5$

(e) For the policy $\pi(s_0) = a_2$, $\pi(s_2) = a_2$, what is the policy graph and the resulting value function $V^\pi(s_2)$, $V^\pi(s_1)$ and $V^\pi(s_0)$?

Solution:
(4/2 points)

Policy graph: 

$V^\pi(s_1) = 3 \quad V^\pi(s_2) = 5 \quad V^\pi(s_0) = 8$

(f) What is the optimal policy for the MDP given in Fig. 5? Briefly explain your answer.

Solution:
(3/2 points)

$\pi^*(s_0) = a_2$, $\pi^*(s_2) = a_2$; Policy with highest $V^\pi(s_0)$ from part (c), (d), and (e).

39. **[13.5 points]** Markov Decision Processes (transition probabilities are given in boxes)

(a) (1 point) Explain the difference between a deterministic and a stochastic Markov decision process. Is the given MDP stochastic or deterministic. Explain your answer.

Solution: deterministic: performing an action always results in a deterministic transition; stochastic: performing an action may lead the agent to one of a few possible states; the given MDP is stochastic

(b) (1.5 points) What are three mechanisms to find the optimal policy π^* for a given MDP?

Solution: Exhaustive search, policy iteration, value iteration

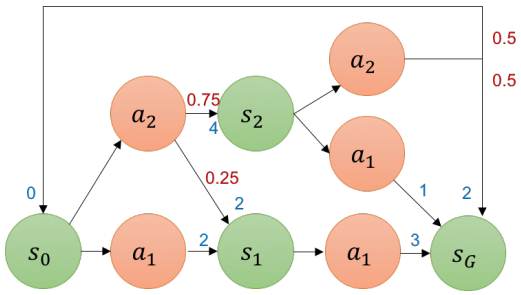
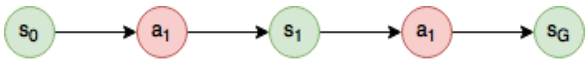


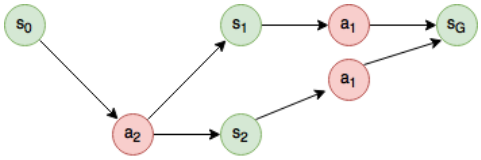
Figure 5: MDP for this problem. Start state is s_0.

(c) (3 points) For the policy $\pi(s_0) = a_1$, $\pi(s_1) = a_1$, what is the policy graph and the resulting value function $V^\pi(s_1)$ and $V^\pi(s_0)$?

Solution: Policy graph: 

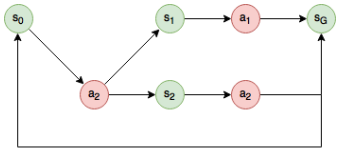
$V^\pi(s_1) = 3 \quad V^\pi(s_0) = 5$

(d) (3 points) For the policy $\pi(s_0) = a_2$, $\pi(s_2) = a_1$, what is the policy graph and the resulting value function $V^\pi(s_2)$, $V^\pi(s_1)$ and $V^\pi(s_0)$?

Solution: Policy graph: 

$V^\pi(s_1) = 3 \quad V^\pi(s_2) = 1 \quad V^\pi(s_0) = 5$

(e) (3 points) For the policy $\pi(s_0) = a_2$, $\pi(s_2) = a_2$, what is the policy graph and the resulting value function $V^\pi(s_2)$, $V^\pi(s_1)$ and $V^\pi(s_0)$?

Solution: Policy graph: 

$V^\pi(s_1) = 3 \quad V^\pi(s_2) = 5 \quad V^\pi(s_0) = 8$

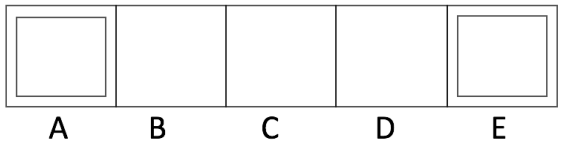
(f) (2 points) What is the optimal policy for the MDP given in Fig. 5? Briefly explain your answer.

Solution: $\pi^*(s_0) = a_2$, $\pi^*(s_2) = a_2$; Policy with highest $V^\pi(s_0)$ from part (c), (d), and (e).

40. **[19 points]** Markov Decision Process (MDP)

Consider the MDP defined on the grid below. In all cases double-rectangle states are exit states. From an exit state (A, E), the only action available is **Exit**, which results in the listed

immediate reward and ends the game (by moving into a terminal state T). From non-exit states, the agent can choose either **Left** or **Right** actions with equal probability, which move the agent in the corresponding direction. The only non-zero rewards come from exiting the grid ($R(A, \text{Exit}, T) = 1$ and $R(E, \text{Exit}, T) = 2$). Throughout this problem, assume that value iteration begins with initial values $V_0(s) = 0$ for all states $S = \{A, B, C, D, E, T\}$.



(a) [3.5 points] Fill in the blank in the MDP diagram below with the missing actions (on the edges), rewards (in squares) and states (in circles).

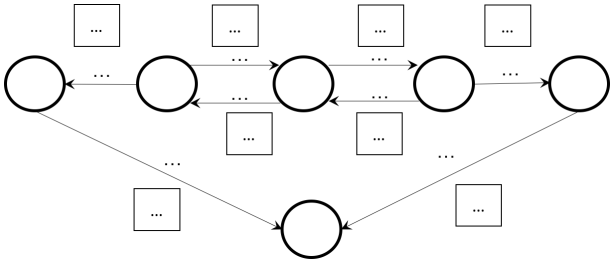
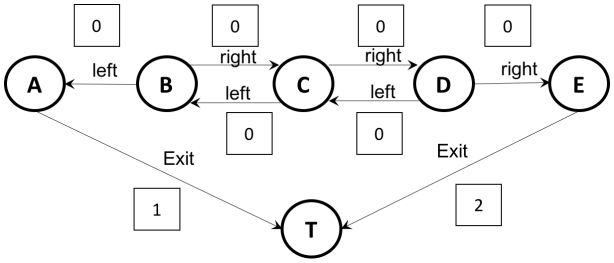


Figure 6: MDP representation

Solution:



(b) [1 points] Given the Bellman equation

$$V^*(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a)(\gamma V^*(s') + R(s, a, s')) \tag{92}$$

with S being the set of states, A_s the set of available actions from the state s, V the value function, R the reward, P the transition probability and γ a discount factor. Write down the update-rule for the **value iteration** algorithm at time t.

Solution:

$$V_{t+1}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a)(\gamma V_t(s') + R(s, a, s')) \tag{93}$$

(c) [1 points] What is the value of $P(s'|s, a)$ for the different states of the grid?

Solution: $P(s'|s, a)$=1.

(d) [3.5 points] Let the discount factor be $\gamma = 0.5$. Fill in the missing values for each state following the value iteration algorithm in the following table:

Solution:

t	A	B	C	D	E
0	0	0	0	0	0
1	1	0	0	0	2
2	1	0.5	0	1	2
3	1	0.5	0.5	1	2

(e) [2 points] Starting from state C, what is the optimal sequence of actions?

Solution: $\pi(C) = \text{Right}$, $\pi(D) = \text{Right}$

(f) [1 points] Instead of finding the optimal policy, assume we want to do **policy evaluation** for a policy π. Write down the iterative refinement formula for the Bellman equation.

Solution:

$$V_{t+1}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a)(\gamma V_t(s') + R(s,a,s')) \tag{94}$$

(g) [3.5 points] Evaluate the following policy: $\pi(B) = \text{Left}$, $\pi(C) = \text{Left}$ and $\pi(D) = \text{Right}$ for $\gamma = 0.5$.

Solution:

t	A	B	C	D	E
0	0	0	0	0	0
1	1	0	0	0	2
2	1	0.5	0	1	2
3	1	0.5	0.25	1	2

(h) [2 points] Recalling that the discount factor must be in range $0 \leq \gamma \leq 1$, for what range of values for γ is the optimal action $\pi^*(B) = \text{Right}$?

Solution: If we take **Left** action in state B: $V(B) = \gamma V(A) = \gamma R(A) = \gamma$.
If we take **Right** action in state B, the state E is the exit state. $V(B) = \gamma V(C) = \gamma^2 V(D) = \gamma^3 V(E) = \gamma^3 R(E) = 2\gamma^3$.
Hence,the following inequality should hold:

$$\gamma < 2\gamma^3 \tag{95}$$

which results in $\gamma < \sqrt{\frac{1}{2}}$. Hence, $\gamma < \sqrt{\frac{1}{2}} \leq 1$.

(i) [1.5 point] How is policy iteration different from value iteration?

Solution:
Policy iteration starts from a random policy and alternates between two steps until convergence: (1) policy evaluation and (2) policy improvement. Value iteration, estimates the value function, then deduces the policy.

41. **[13 points]** Q-Learning

(a) State the Bellman optimality principle as a function of the optimal Q-function $Q^*(s,a)$, the expected reward function $R(s,a,s')$ and the transition probability $P(s'|s,a)$, where s is the current state, s' is the next state and a is the action taken in state s.

Solution:
(1 point)

Bellman Equation:
$Q^*(s,a) = \sum_{s' \in S} P(s'|s,a)[R(s,a,s') + \max_{a' \in A_{s'}} Q^*(s',a')]$

(b) In case the transition probability $P(s'|s,a)$ and the expected reward $R(s,a,s')$ are unknown, a stochastic approach is used to approximate the optimal Q-function. After observing a transition of the form (s,a,r,s'), write down the update of the Q-function at the observed state-action pair (s,a) as a function of the learning rate α, the discount factor γ, $Q(s,a)$ and $Q(s',a')$.

Solution:
(2 points)

$Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a' \in A_{s'}} Q(s',a'))$

(c) What is the advantage of an epsilon-greedy strategy?

Solution:
(1 point)

Trade-off between exploration and exploitation. The best long-term strategy may involve short-term sacrifices resulting in not taking the best action at the beginning of the train to better explore the environment.

(d) What is the advantage of using a replay-memory?

Solution:
(1 point)

Learning from batches of consecutive samples is problematic, as the samples are correlated. This can lead to inefficient learning. For example, if maximizing action is to move left, training samples will be dominated by samples from left-hand size. Instead, a replay memory is used to store the transitions (s_t, a_t, r_t, s_{t+1}) as game episodes are played. The Q-network is then trained on random minibatches of transitions from the replay memory, instead of consecutive samples.

(e) Consider a system with two states S_1 and S_2 and two actions a_1 and a_2. You perform actions and observe the rewards and transitions listed below. Each step lists the current state, reward, action and resulting transition as: $S_i; R = r; a_k : S_i \to S_j$. Perform Q-learning using a learning rate of $\alpha = 0.5$ and a discount factor of $\gamma = 0.5$ for each step by applying the formula from part (b). The Q-table entries are initialized to zero. Fill in the tables below corresponding to the following four transitions. What is the optimal policy after having observed the four transitions?

i. S_1; $R = -10$; $a_1 : S_1 \to S_1$
ii. S_1; $R = -10$; $a_2 : S_1 \to S_2$
iii. S_2; $R = 18.5$; $a_1 : S_2 \to S_1$
iv. S_1; $R = -10$; $a_2 : S_1 \to S_2$

Q	S_1	S_2
a_1	.	.
a_2	.	.

Q	S_1	S_2
a_1	.	.
a_2	.	.

Q	S_1	S_2
a_1	.	.
a_2	.	.

Q	S_1	S_2
a_1	.	.
a_2	.	.

Solution:
(8 points/ 2 for each)

i. $Q(a_1, S_1) = -5$, $Q(a_2, S_1) = 0$, $Q(a_1, S_2) = 0$, $Q(a_2, S_2) = 0$

ii. $Q(a_1, S_1) = -5$, $Q(a_2, S_1) = -5$, $Q(a_1, S_2) = 0$, $Q(a_2, S_2) = 0$

iii. $Q(a_1, S_1) = -5$, $Q(a_2, S_1) = -5$, $Q(a_1, S_2) = 8$, $Q(a_2, S_2) = 0$

iv. $Q(a_1, S_1) = -5$, $Q(a_2, S_1) = -5.5$, $Q(a_1, S_2) = 8$, $Q(a_2, S_2) = 0$

The optimal policy at this point is: $\pi(S_1) = a_1$ and $\pi(S_2) = a_1$.

42. **[10 points]** Q-Learning

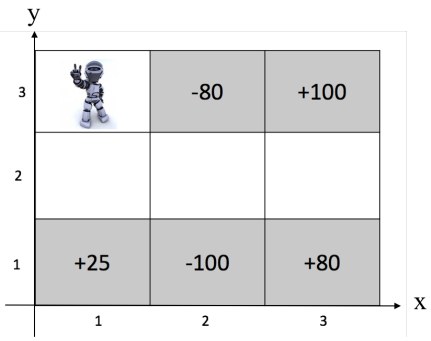


Figure 7: Find me the optimal policy. May the force be with us!

Consider the grid-world given in Figure 7 and our artificial agent who is trying to learn the optimal policy. If an action results in landing into one of the shaded states the corresponding reward is awarded during that transition. All shaded states are terminal states. The other states have the North, East, South, West actions available, which deterministically move the agent to the corresponding neighboring state (or have him stay in place if the action tries to move out of the grid). Assume a discount factor $\gamma = 0.5$ and the Q-learning rate of $\alpha = 0.5$ for all calculations. The agent starts in state $(x, y) = (1, 3)$.

(a) (2 points) Write down the expression of the optimum value $V^*(s)$ at state s as a function of the reward $r^a_{ss'}$, γ and the next state's optimum value $V^*(s')$ in this grid-world.

Solution: $V^*(s) = \max_a(r^a_{ss'} + \gamma V^*(s'))$

(b) (1 point) What is the optimal value V^* at state $(3, 2)$?

Solution: 100

(c) (1 point) What is the optimal value V^* at state $(2, 2)$?

Solution: $0 + \gamma * 100 = 50$

(d) (1 point) What is the optimal value V^* at state $(1, 3)$?

Solution: (1 point) Going to either of +25 or +100 has the same discounted reward of 12.5.

The agent starts from the top left corner and you are given the following three episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Episode1	Episode2	Episode3
$(1,3), S, (1,2), 0$	$(1,3), S, (1,2), 0$	$(1,3), S, (1,2), 0$
$(1,2), E, (2,2), 0$	$(1,2), E, (2,2), 0$	$(1,2), E, (2,2), 0$
$(2,2), S, (2,1), -100$	$(2,2), E, (3,2), 0$	$(2,2), E, (3,2), 0$
	$(3,2), N, (3,3), +100$	$(3,2), S, (3,1), +80$

(e) (2 points) Write down the Q-learning gradient update rule for $Q(s, a)$ as a function of α, γ, $Q(s', a')$ and $R(s, a, s')$.

Solution: $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(R(s,a,s') + \gamma \max_{a'} Q(s',a'))$

(f) (3 points) Assume the entire aforementioned experience replay to be given and all the Q values are initialized to zero. Perform a single step of gradient updates for the following Q-values: $Q((3,2),N)$, $Q((1,2),S)$ and $Q((2,2),E)$. The aforementioned Q-values are updated in the given order, i.e., when updating $Q((2,2),E)$, $Q((3,2),N)$ and $Q((1,2),S)$ are already updated and their updated values should be used.

Solution: (1) $Q((3,2),N) = 50$, (2) $Q((1,2),S) = 0$, (3) $Q((2,2),E) = 12.5$