

Name:

NetID:

**Reminder:** Please turn this practice packet in on either canvas or in-person at the exam for 5 points of credit on the exam (completion). This packet contains problems covering content from 02/26 to 03/31 (aka from the lecture titled “Perceptrons” through the lecture titled “Gradient Descent and Backpropagation”).

**Note:** Some questions have “Tom’s Thoughts” parts to them in parentheses. These “Tom’s Thoughts” are not necessary to answer for credit on the packet, but are intended to prompt your thinking on a deeper level about some of the concepts covered in this unit. They are things that Tom has seen in classes that follow this one that will be useful for you to remember and know!

**Note:** Questions that have been directly lifted from previous exams (and thus are incredibly indicative of what you may expect) have been marked with their semester and year

### **Perceptrons (02/26)**

1.) Please draw a diagram of a perceptron. Make sure to include inputs, weights, a bias term, a weighted sum, and an activation function.



2.) Given below is a Perceptron. Using the Step function between -1 and 1 as the activation function, classify the three points given below. Then update the Perceptron weights for each misclassified sample and record the new weights. (Fall 2024)

$$\omega = (1.2, 0.7, 0.2, 0.5), \eta = 0.2$$

Samples	$X_1$	$X_2$	$X_3$	Y
$S_1$	2	3	-1	1
$S_2$	1	1	1	1
$S_3$	-2	-4	3	-1

Classify:

$S_1$ :

$S_2$ :

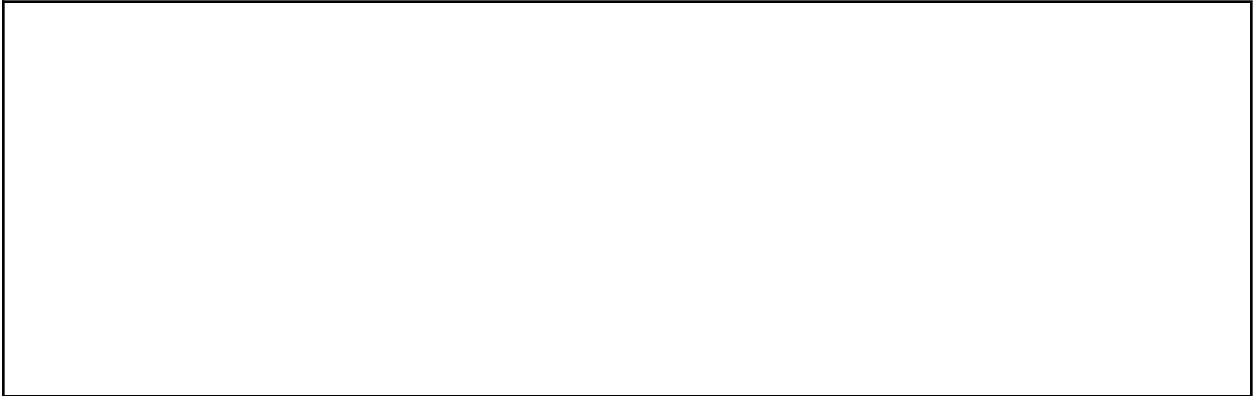
$S_3$ :

Weight Updates:

Final Weights:

$$\omega = (\omega_0: \quad, \omega_1: \quad, \omega_2: \quad, \omega_3: \quad)$$

3.) What is the perceptron learning rule? Give a brief “english” label for each component (I.E. “old/previous weight”)



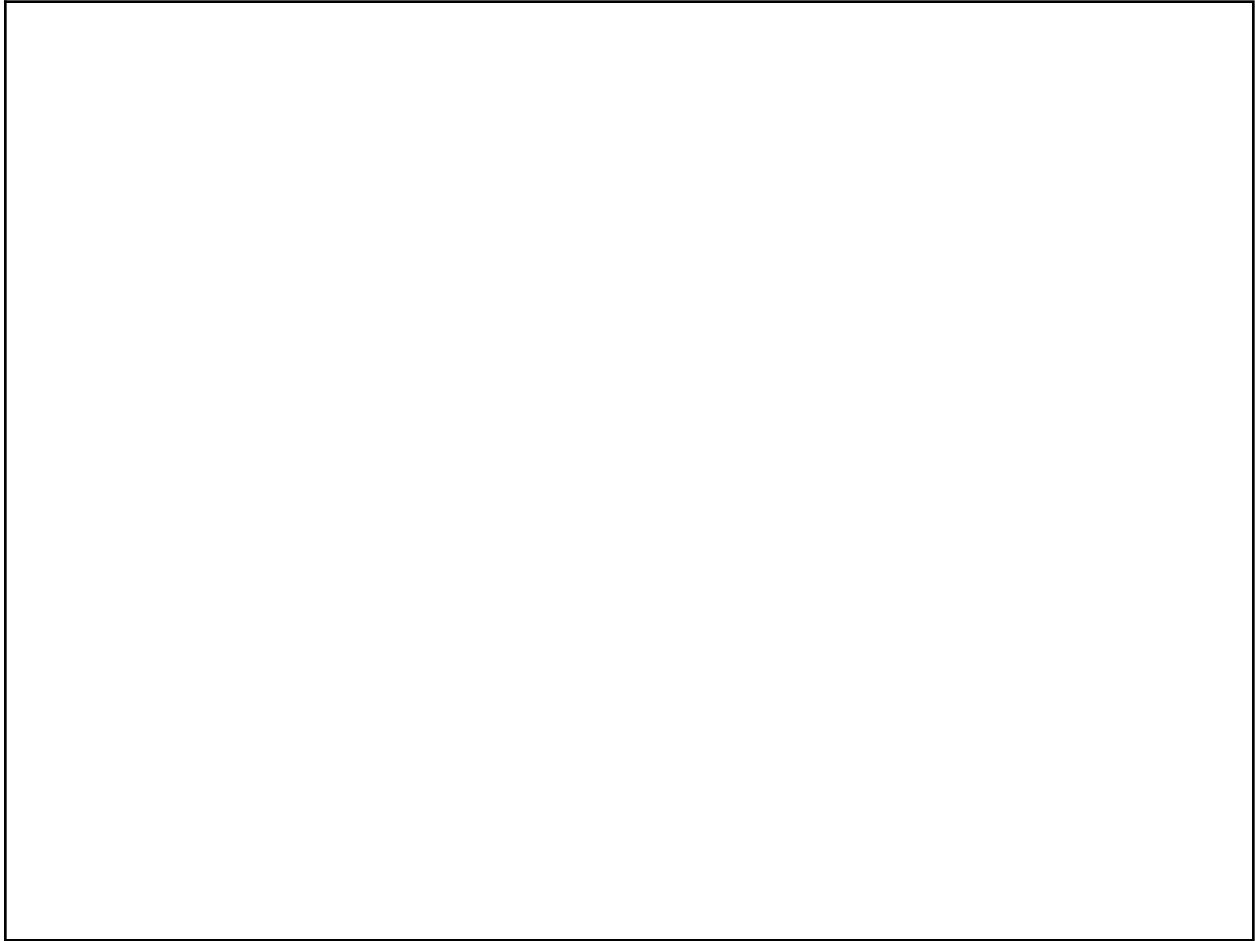
4.) If you replace the step activation function in a perceptron with a sigmoid function, what kind of model does it become? What could we use instead of the Perceptron learning rule?



## Logistic Regression (03/03)


1.) What is a sigmoid function? Write down the equation and graph for the sigmoid function. Explain the role of the sigmoid function in logistic regression (and in binary classification in general). (Tom's Thoughts: What properties of the sigmoid function make it useful for classification?)

2.) Why is Binary Cross-Entropy Loss commonly used with Logistic Regression? What does it measure? (Tom's Thoughts: why use a log in the loss function?)



**Support Vector Machines (SVMs) (03/05)**

1.) What makes an SVM different from a Perceptron with respect to the decision boundary? (Fall 2024)

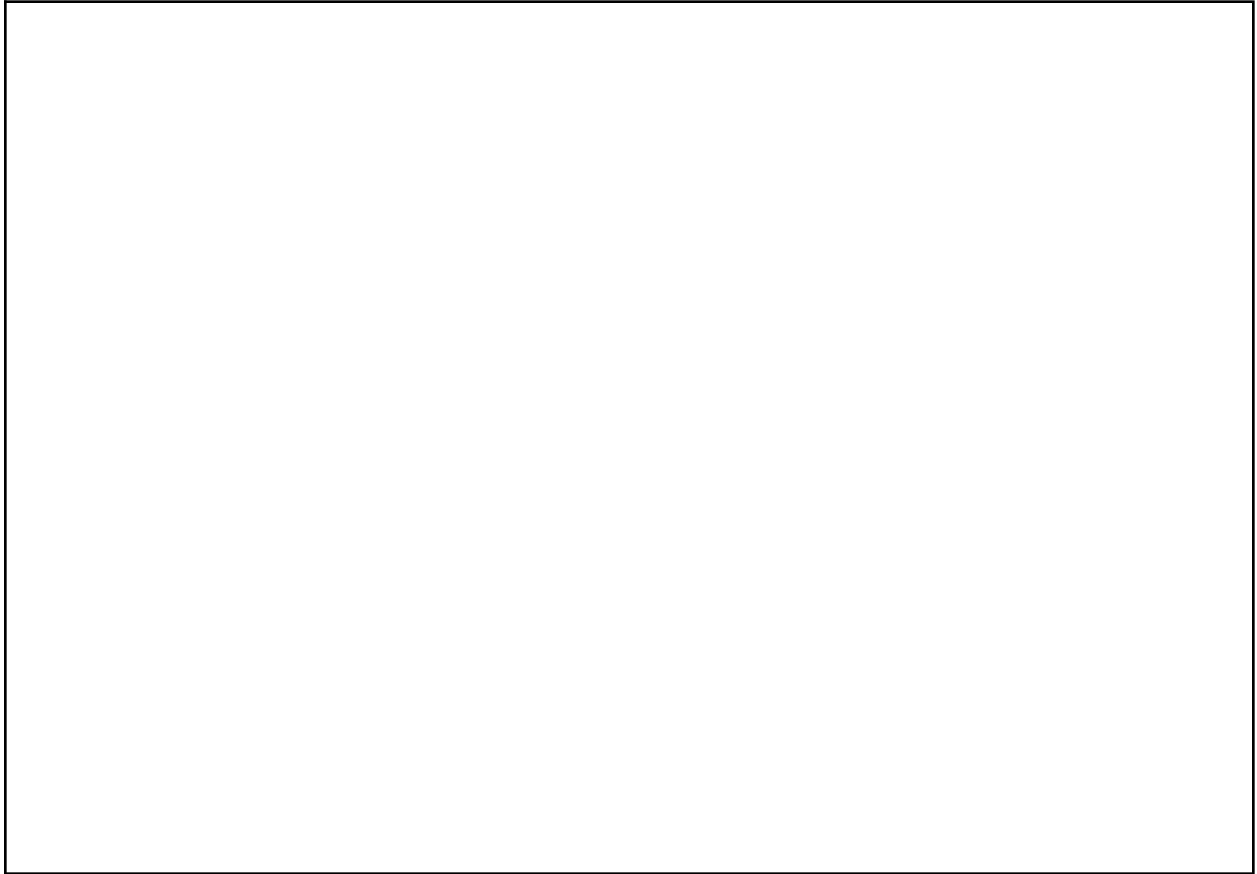


2.) What are support vectors, and why are they important in SVM? (Fall 2024)

3.) Explain the difference between a **hard-margin SVM** and a **soft-margin SVM**. When might you want to use each?

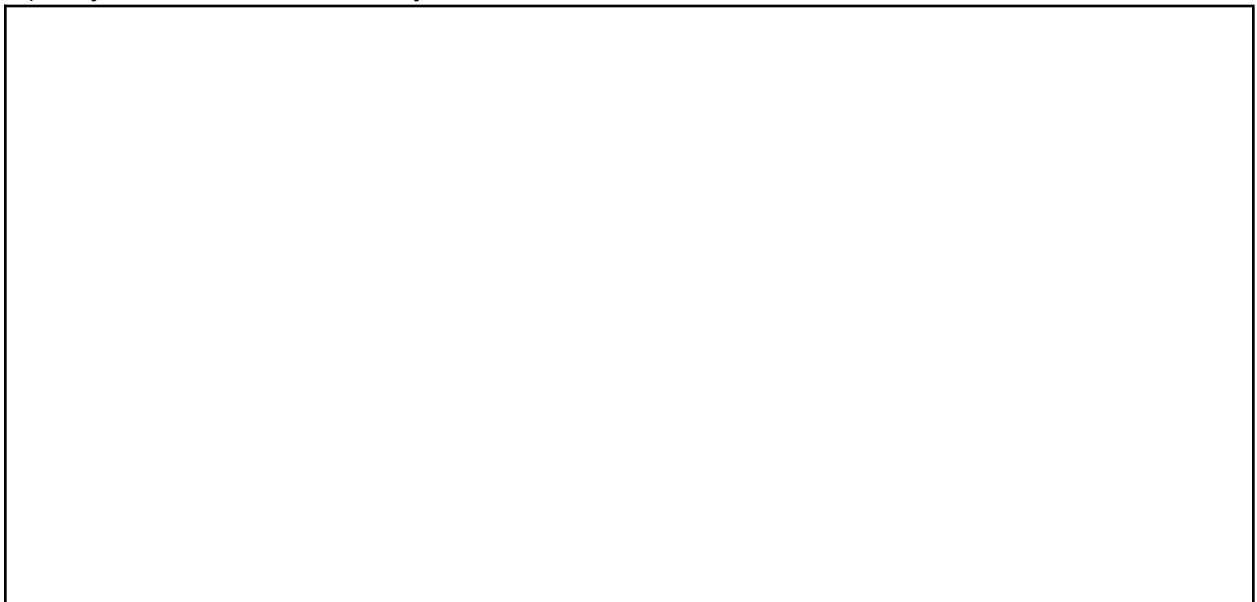
4.) How does SVM handle non-linearly separable data? What role does the kernel function play in this?

5.) What are the three most common kernels used in SVMs? (Tom's Thoughts: What is an advantage and disadvantage of each kernel?)



**K-Nearest Neighbors (KNN) (03/05)**

1.) Why would we call KNN "lazy"?



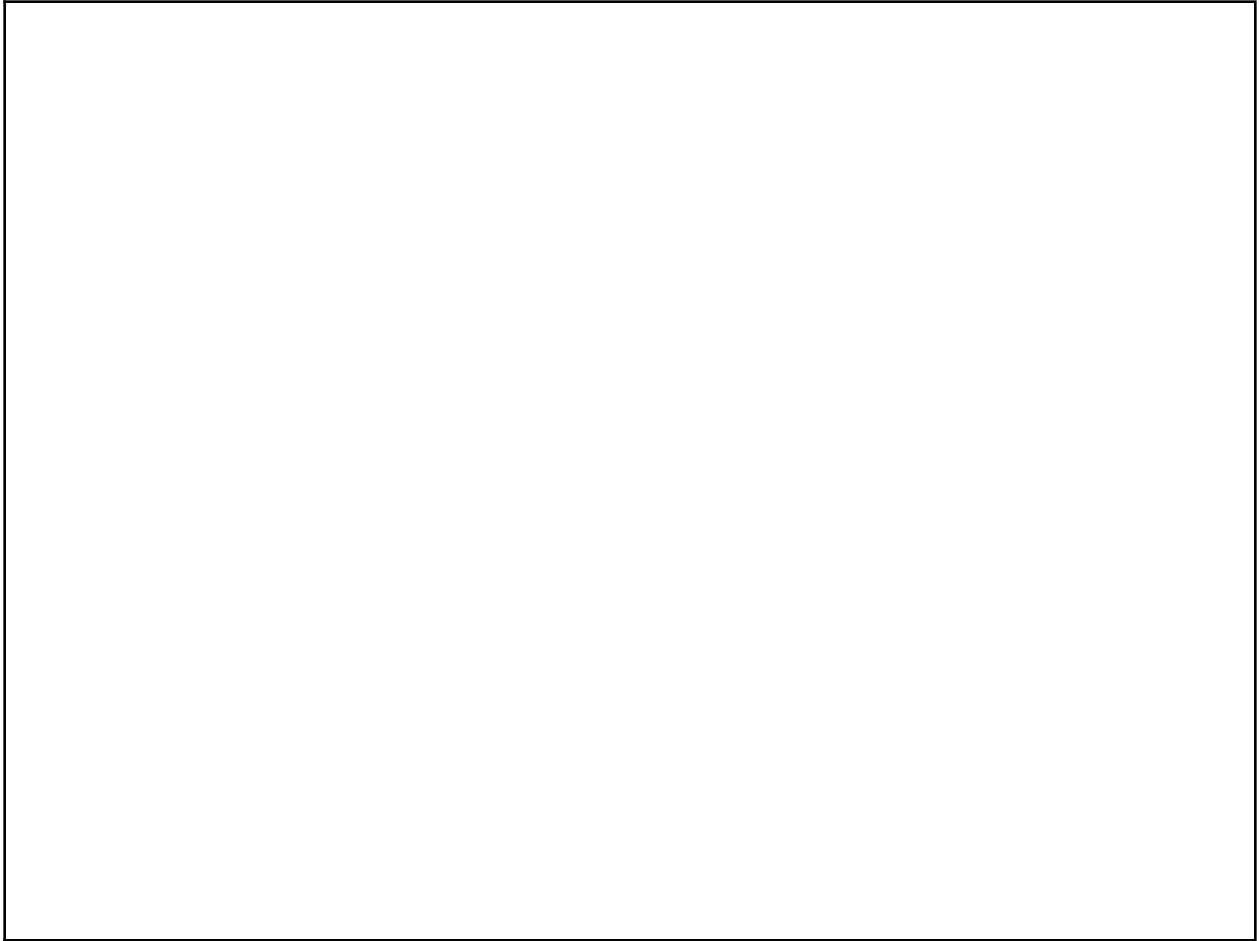
2.) What's a strategy we could use to break ties in voting with a KNN? (Tom's Thoughts: What are "distance weighted knns")

**Clustering (03/17)**

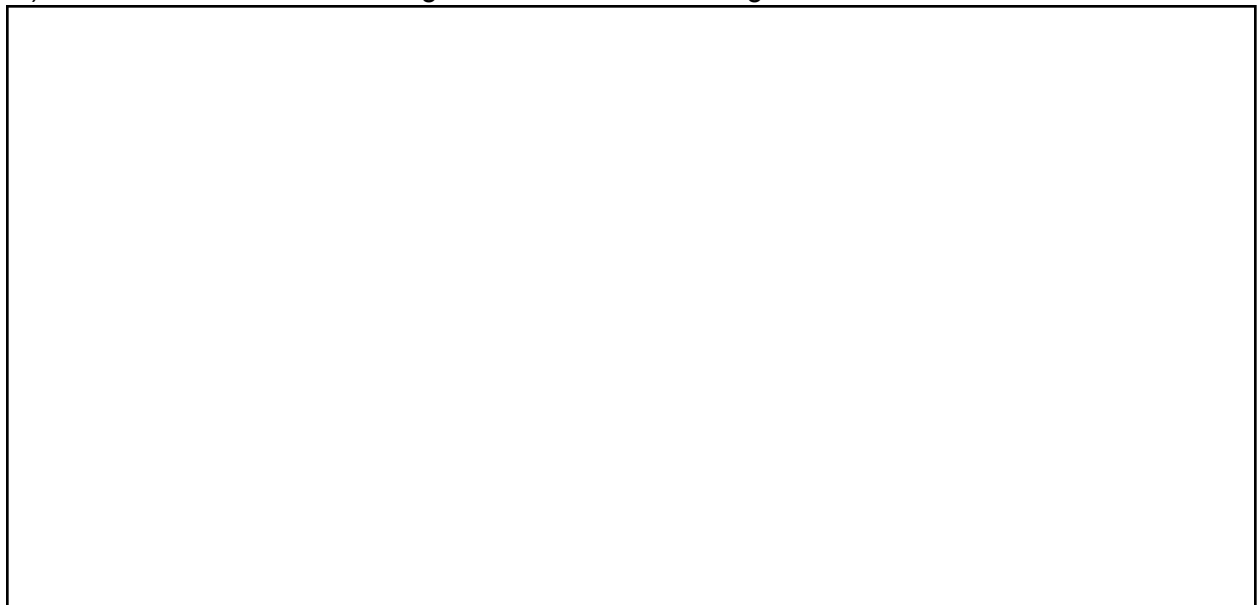
1.) What is the goal of unsupervised learning? What is the goal of clustering? How does clustering differ from classification?



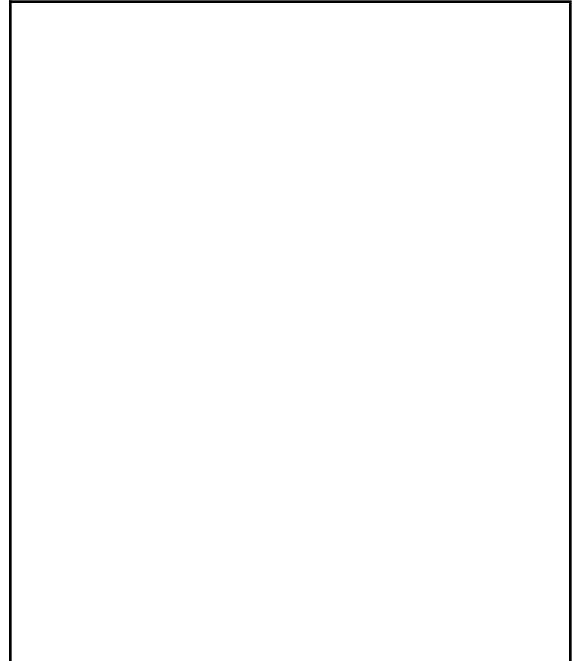
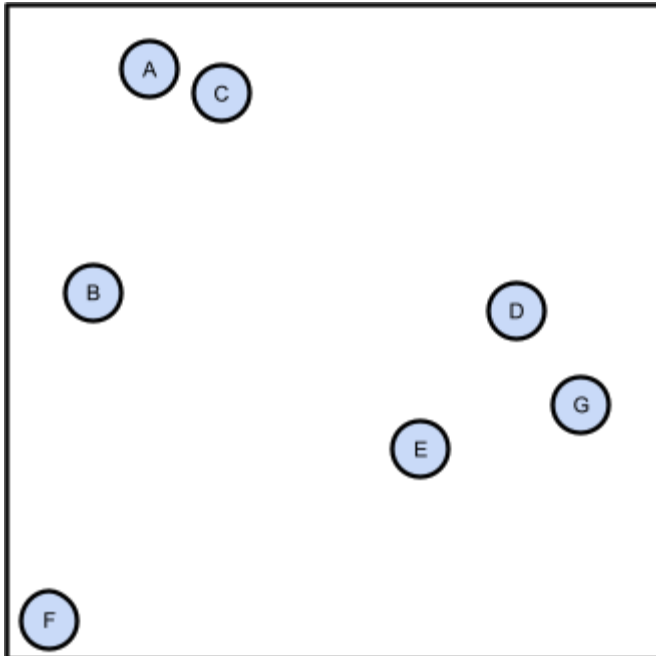
2.) What are the four steps of k-means clustering we saw in class? What assumptions does it make? What are some pros and cons of kmeans?



3.) What is hierarchical clustering and what is an advantage has over k means?



4.) Given the data points, draw the dendrogram that would be created using agglomerative hierarchical clustering and then draw a line on the dendrogram to create 4 clusters. (Fall 2024)



5.) Explain the difference between agglomerative and divisive hierarchical clustering.



6.) Provide two real-world applications where clustering could be used and explain why.

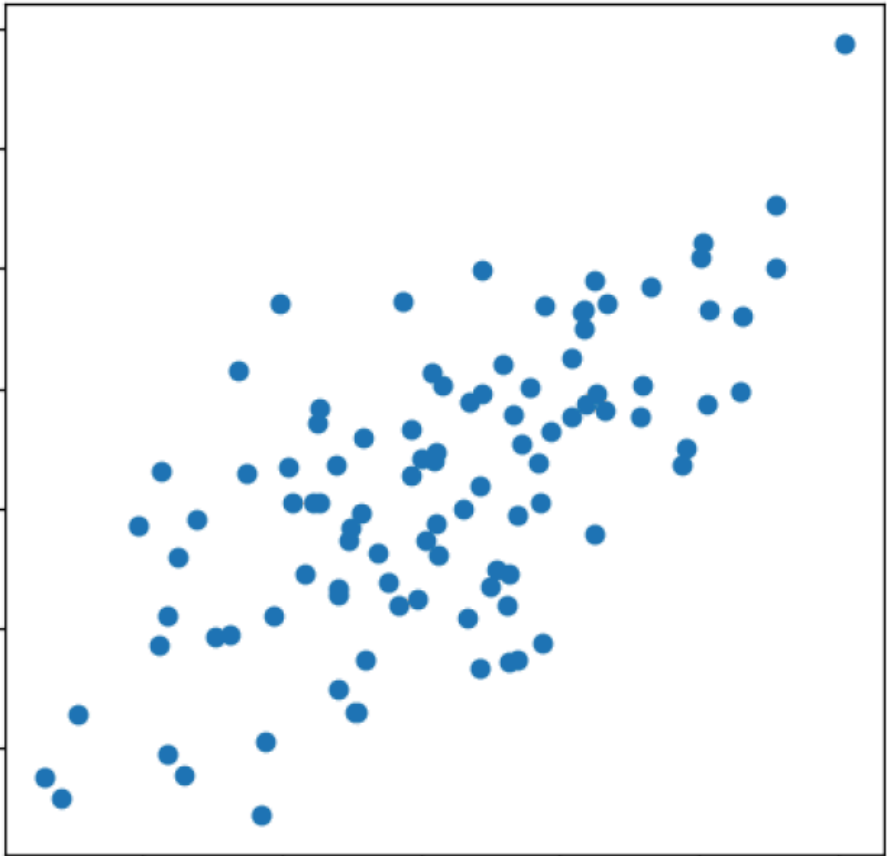


**Dimensionality Reduction (03/19)**

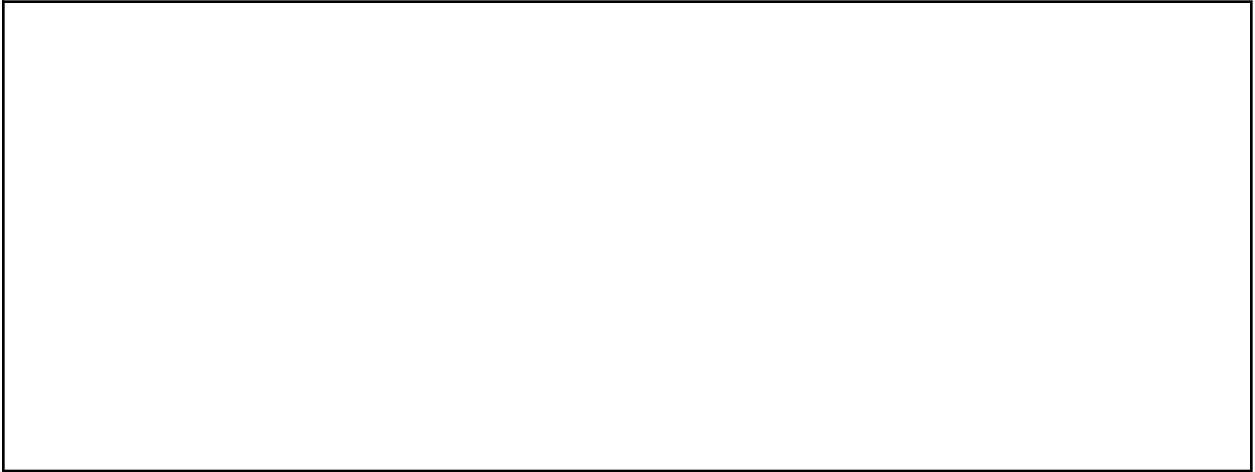
1.) What is dimensionality reduction, and when may we use it?



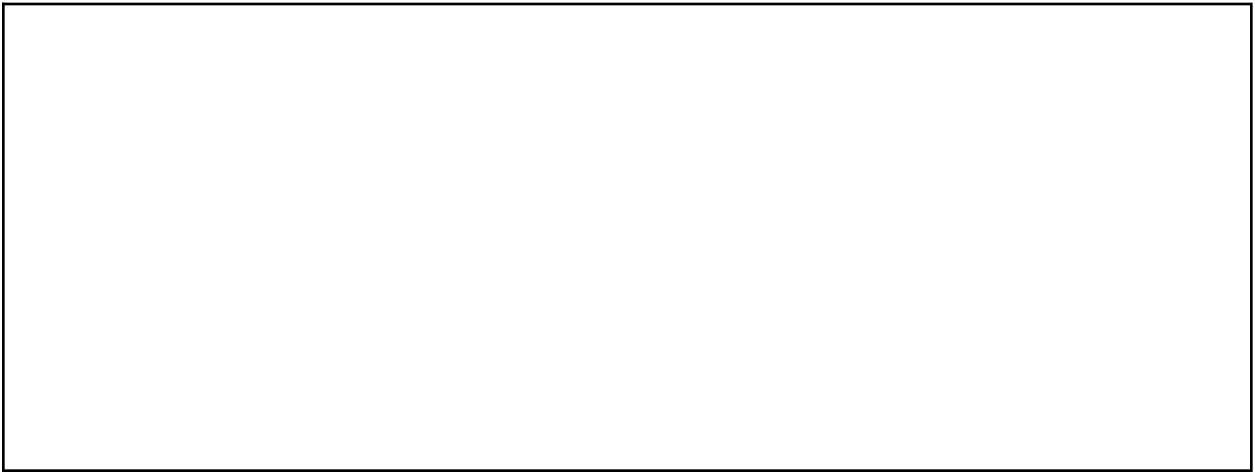
2.) Draw (approximately) the two principal components on the plot and label them (Fall 2024)



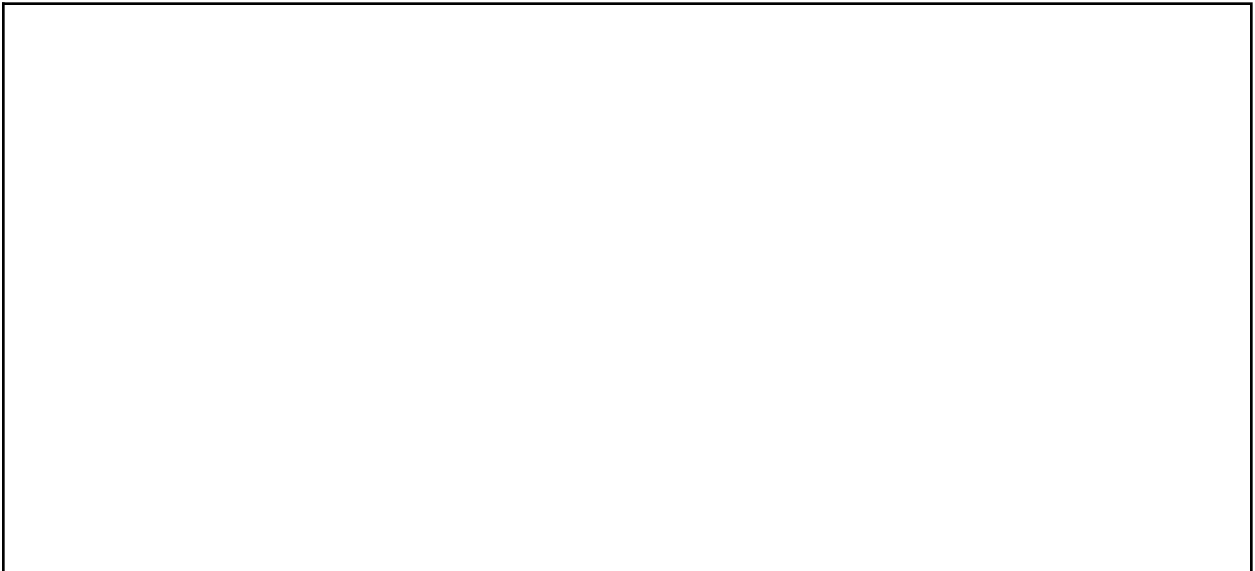
3.) Why are the principal components always orthogonal to each other



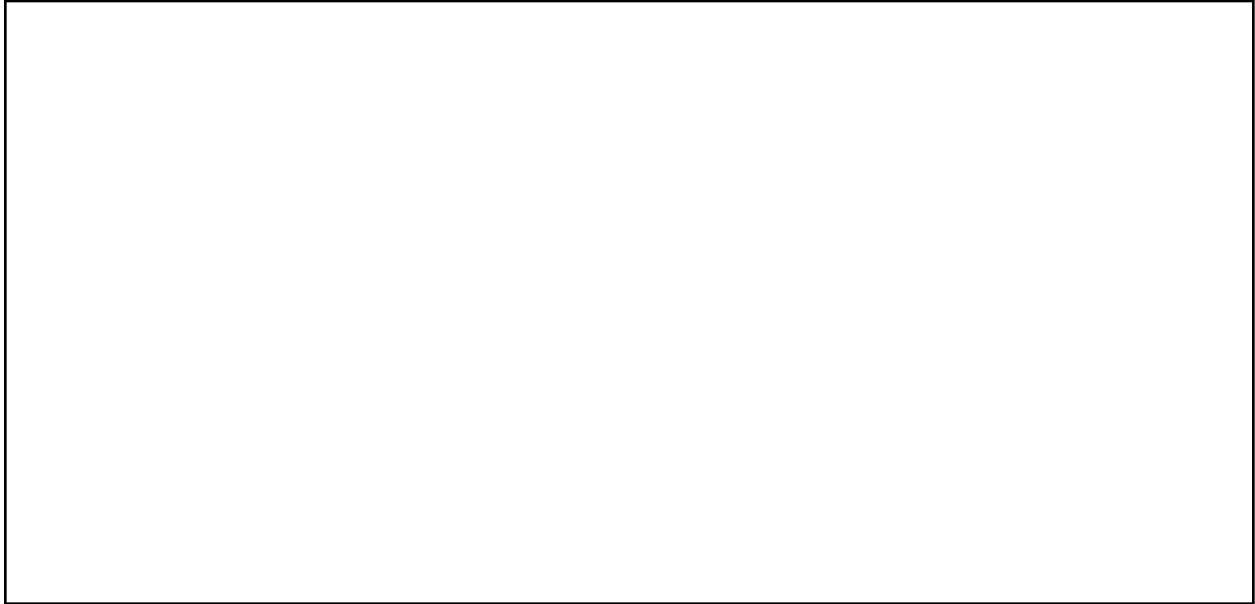
4.) Explain the curse of dimensionality and how it affects machine learning models. (Fall 2024)



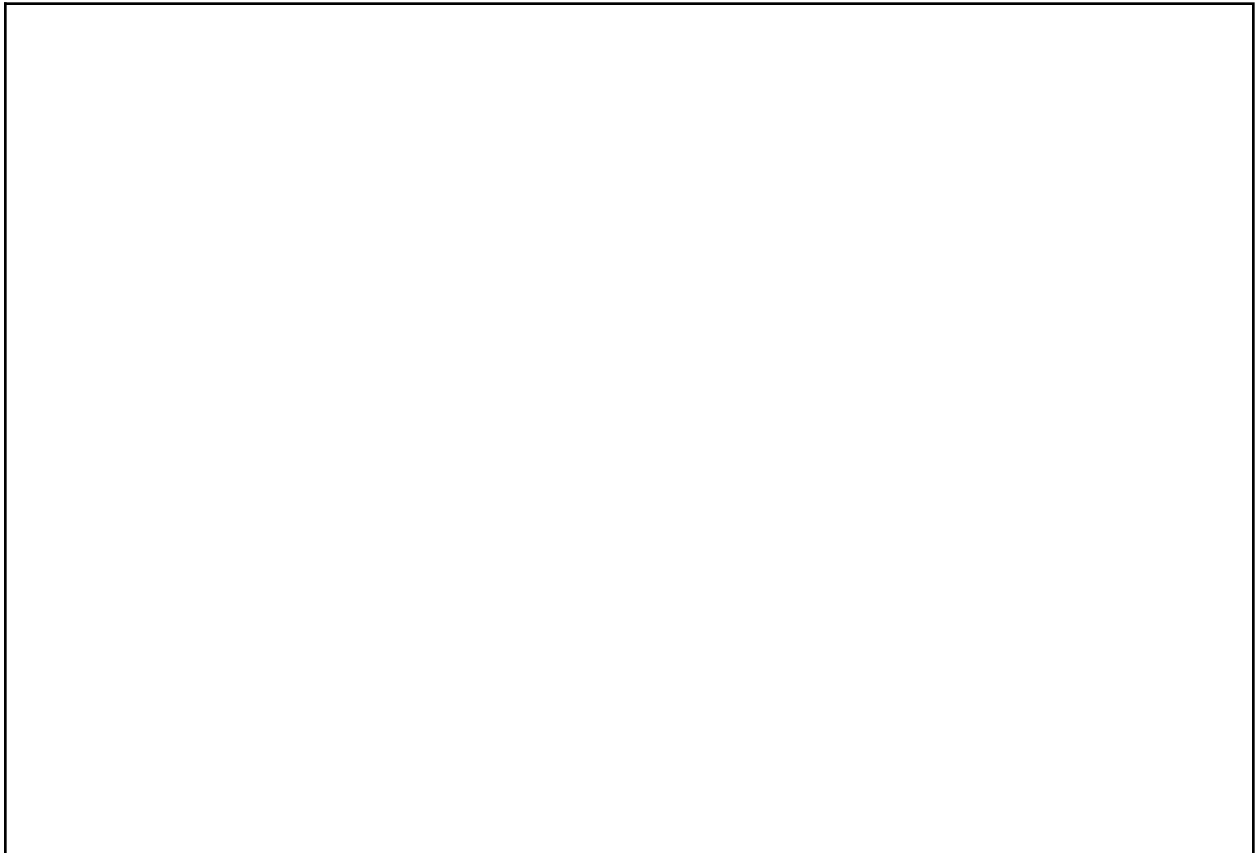
5.) What are the roles of eigenvalues and eigenvectors in PCA (Fall 2024)



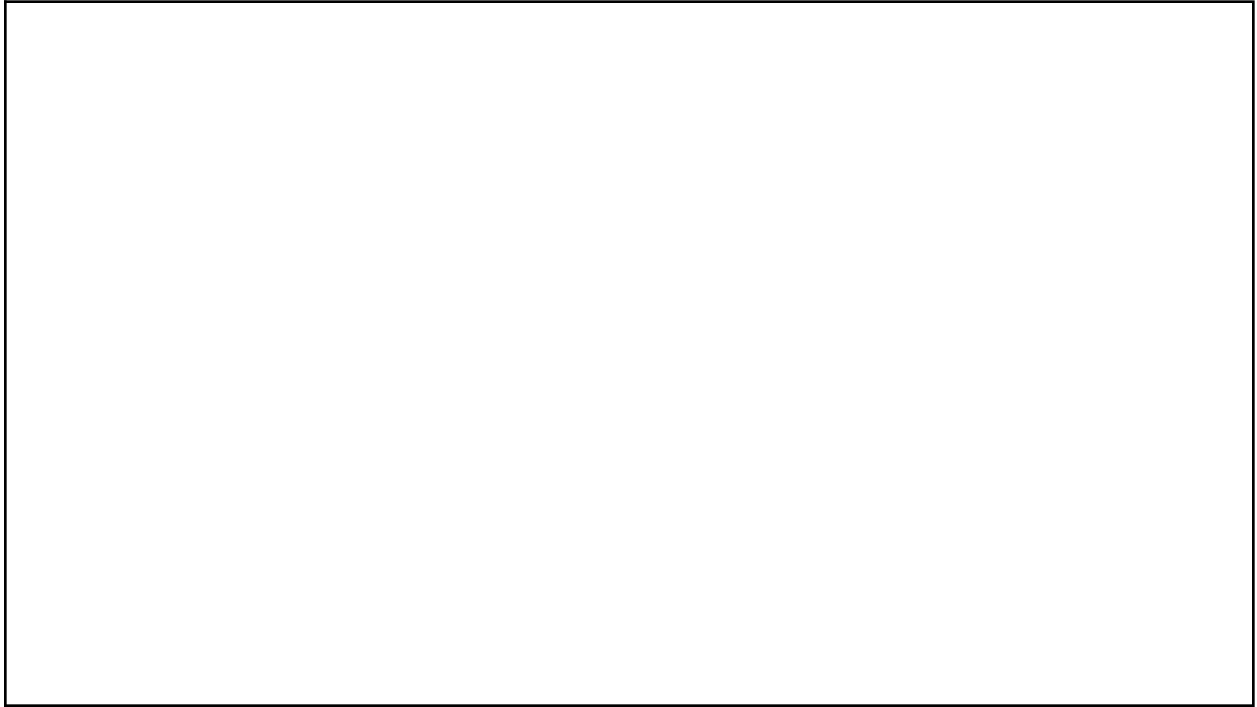
6.) If I have a dataset with 5 input features and then transform the 5 features into 5 principal components, did I lose any information in my dataset? Why or why not? If I only select and use the top 2 principal components did I lose any information? Why or why not?



7.) What does it mean to standardize your data? (Tom's Thoughts: Why do you need to standardize your data before performing PCA?)



8.) What is input space? What is feature space? (Tom's Thoughts: Why is having/learning a good feature space important?)



**Practicum - Evaluation Metrics (03/24)**

1.) What are Type 1 and Type 2 errors?



2.) Given the following experiments, which of the metrics do you think would be most useful for measuring task performance. Select only one. (Multiple Choice) (Fall 2024)

a.) An imbalanced multiclass classification task

- Precision
- Recall
- F1 Score
- MSE

b.) Deciding whether to give someone a loan

- Precision
- MSE
- Accuracy
- Silhouette Score

c.) A regression task

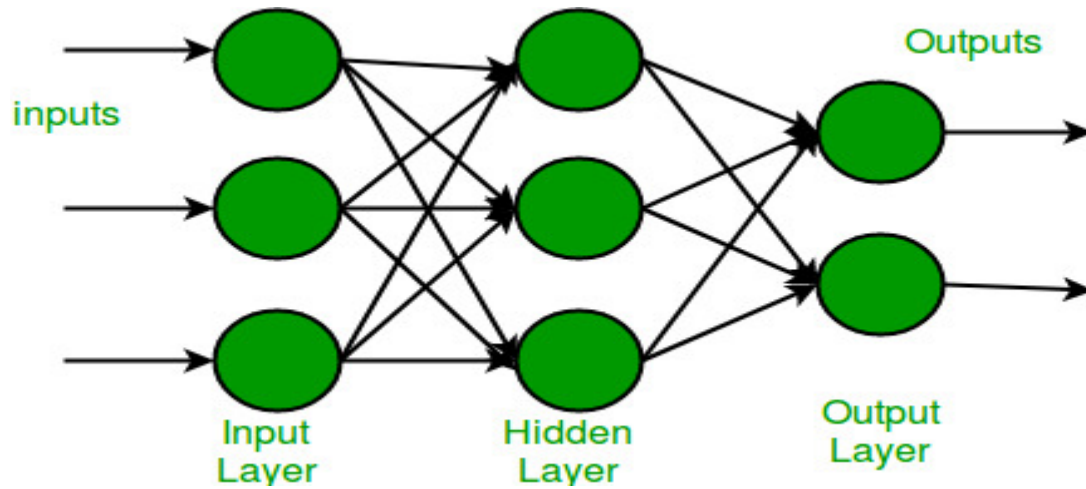
- Precision
- F1 Score
- MSE
- Laplacian Difference

3.) Match the following metrics to their equation and definition: Accuracy, Precision, Recall, F1 Score

	$(TP + TN) / (TP + TN + FP + FN)$		“How close a given set of measurements are to their actual values”
	$TP / (TP + FP)$		“How close measurements are to each other”
	$TP / (TP + FN)$		“Proportion of actual positives that were classified correctly”
	$2 * (Precision * Recall) / (Precision + Recall)$		“The harmonic mean of precision and recall”

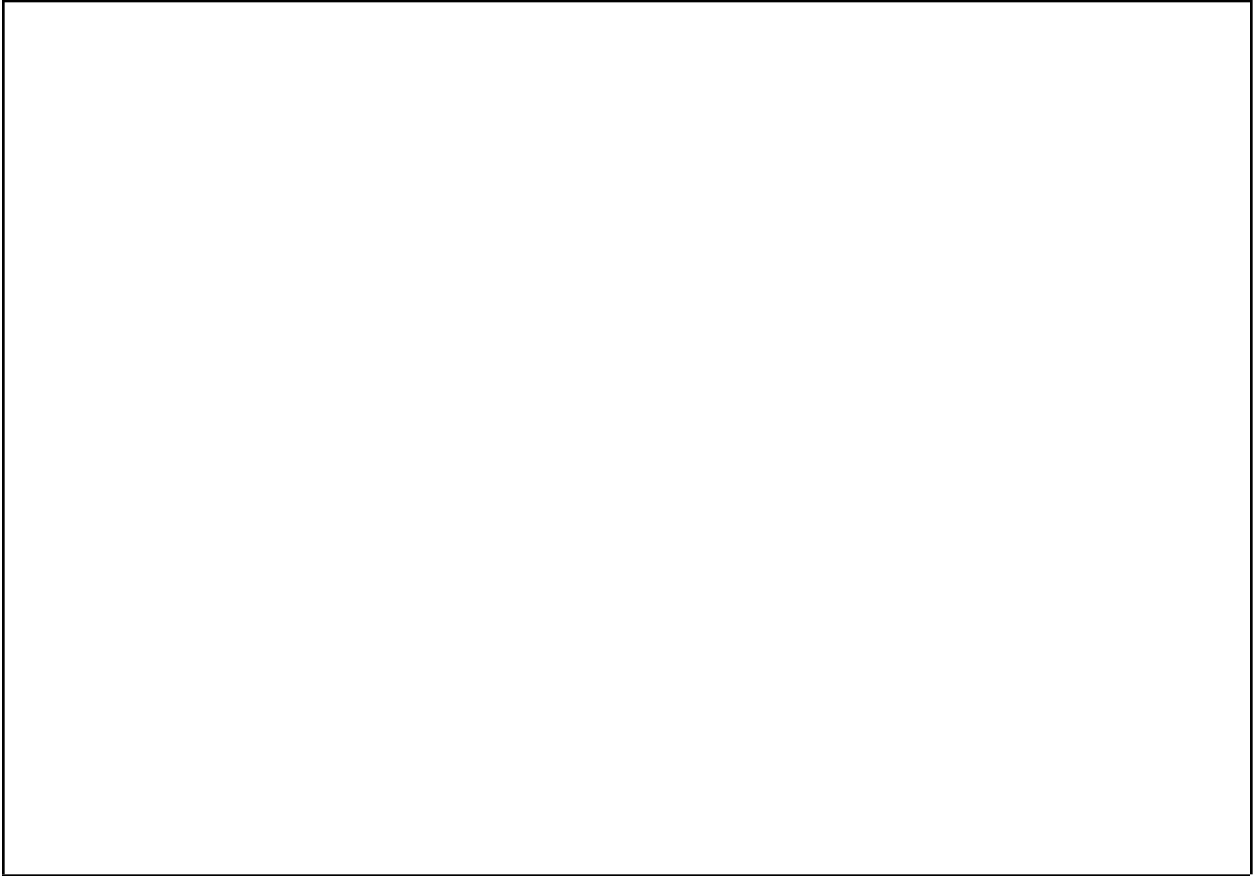
## Feed Forward Neural Networks (FFNs) (MLPs) (03/26)

1.) Explain the structure of a multilayer perceptron (MLP). What are the roles of the input layer, hidden layers, and output layer?

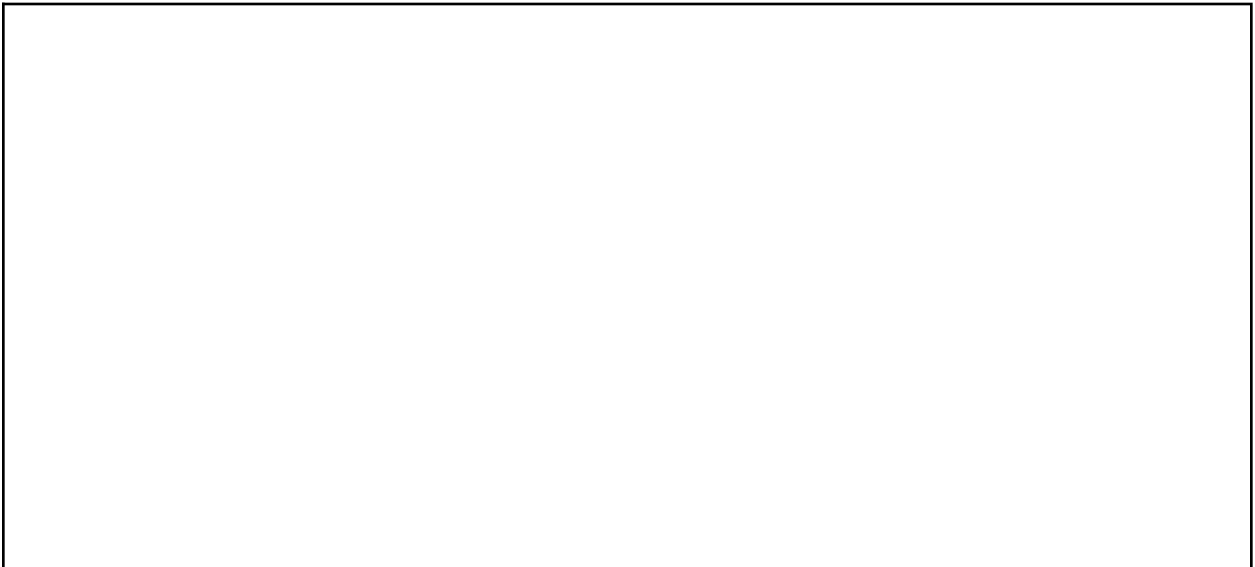




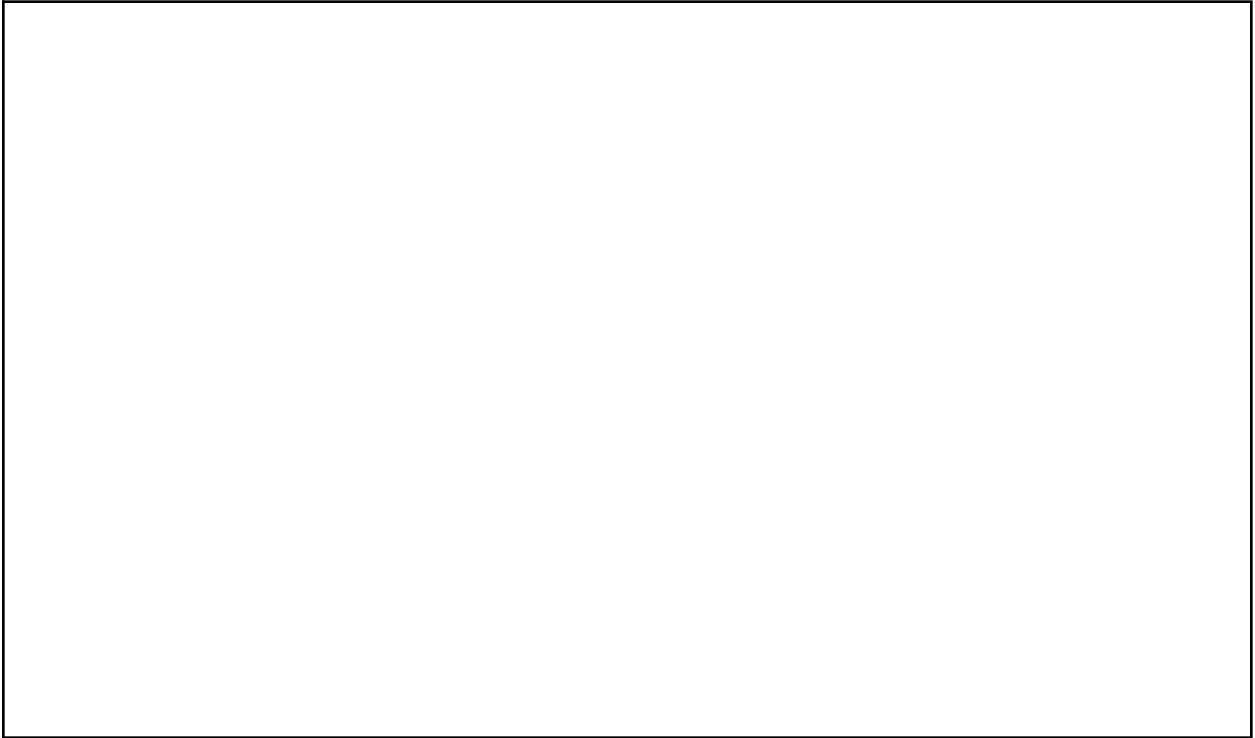
2.) What are activation functions, and why are they necessary in an MLP? (Fall 2024) (Tom's Thoughts: Compare and contrast common activation functions, specifically: **ReLU**, **sigmoid**, and **tanh**.)



3.) Describe the forward propagation process in an MLP. How is the output of a neural network calculated from the input?



4.) What is the role of a loss function in training an MLP? What are common loss functions used in classification and regression tasks? (Fall 2024)

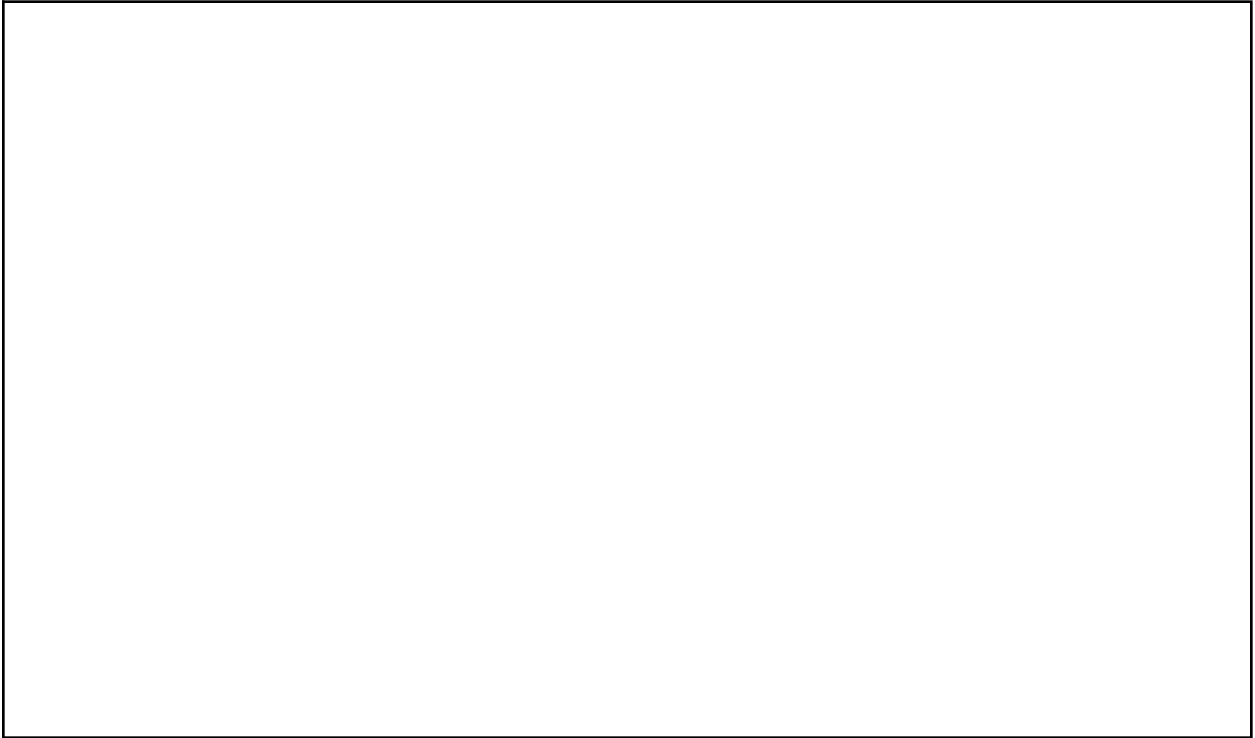


**Gradient Descent and Backprop (03/31)**

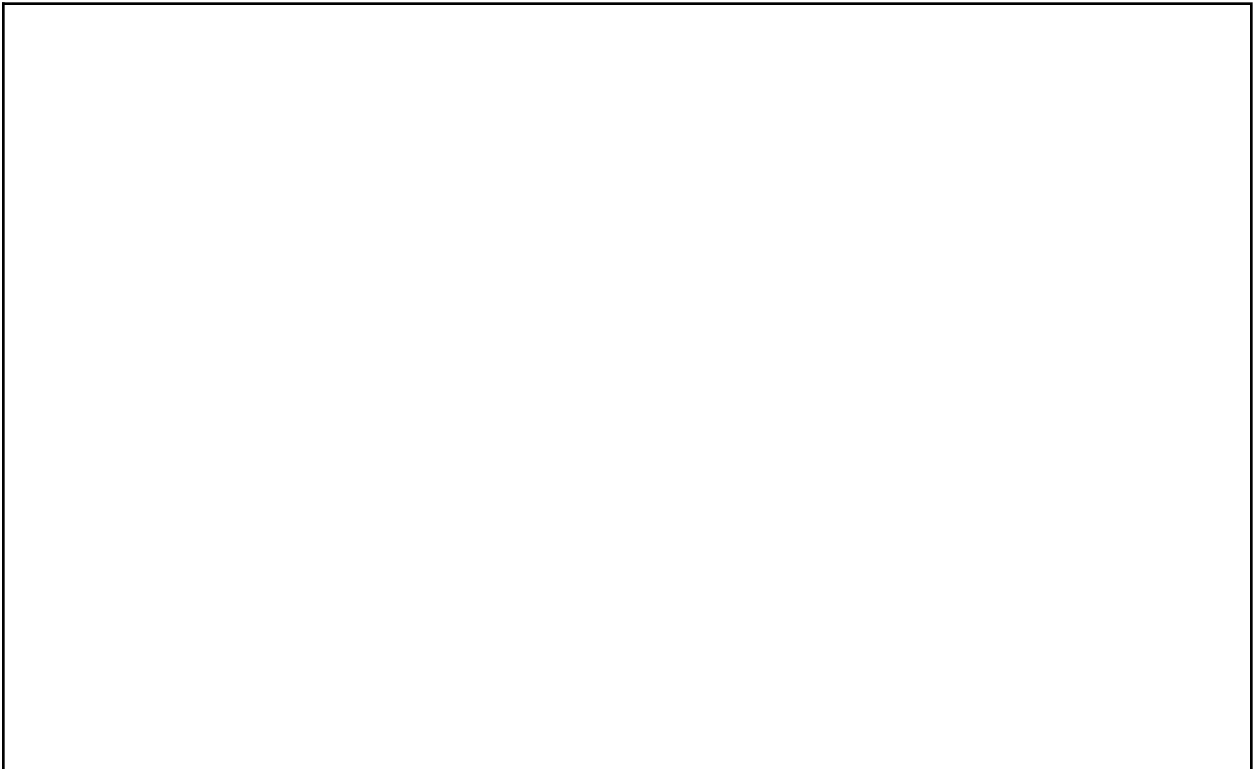
1.) What is the difference between backpropagation and gradient descent? How are they related? (Fall 2024)



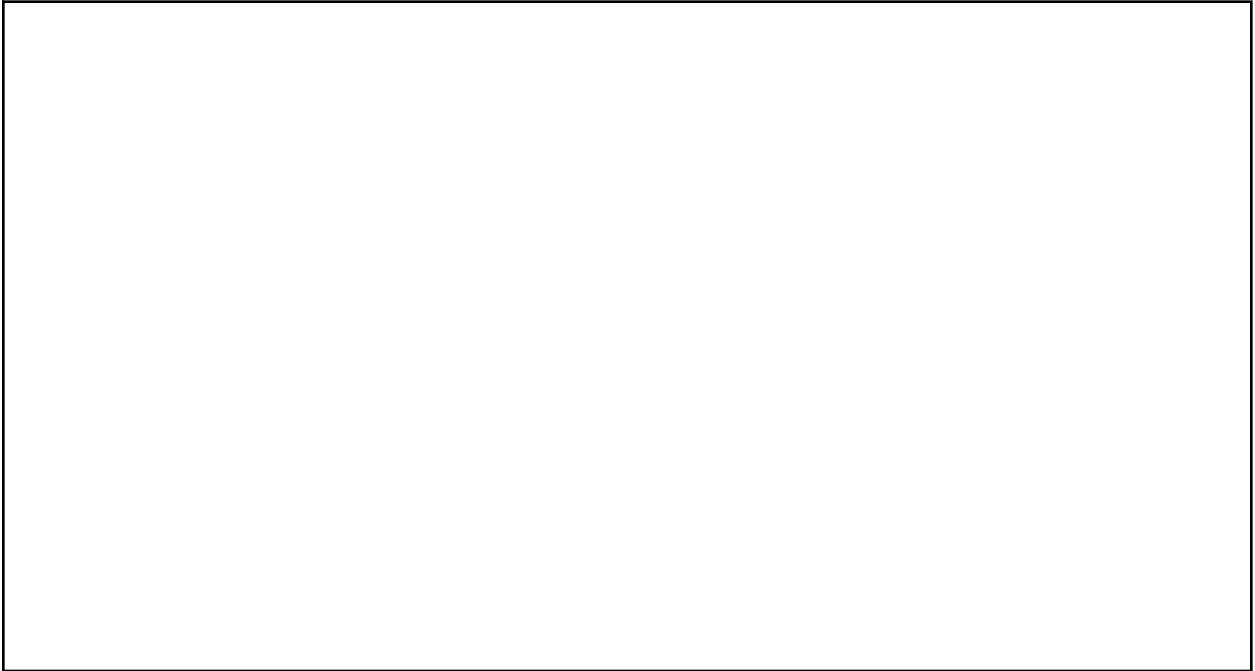
2.) Describe the role of the chain rule in backpropagation. Why is it necessary?



3.) What are vanishing and exploding gradients, and how do they affect backpropagation / neural network training? (Fall 2024)



4.) What is the difference between batch gradient descent, stochastic gradient descent (SGD), and mini-batch gradient descent?



5.) How does the learning rate affect the performance of gradient descent? What are potential issues with too high or too low learning rates? (Fall 2024)



6.) Explain the role of momentum in gradient descent. How does it help optimize convergence?

### Neural Network Application Questions

Given the network diagram, answer the following questions. (Similar to Fall 2024)

```
import torch
import torch.nn as nn
import torch.nn.functional as F

class SimpleNetwork(nn.Module):
    def __init__(self):
        super(SimpleNetwork, self).__init__()
        self.fc1 = nn.Linear(2, 32)
        self.fc2 = nn.Linear(32, 16)
        self.fc3 = nn.Linear(16, 1)

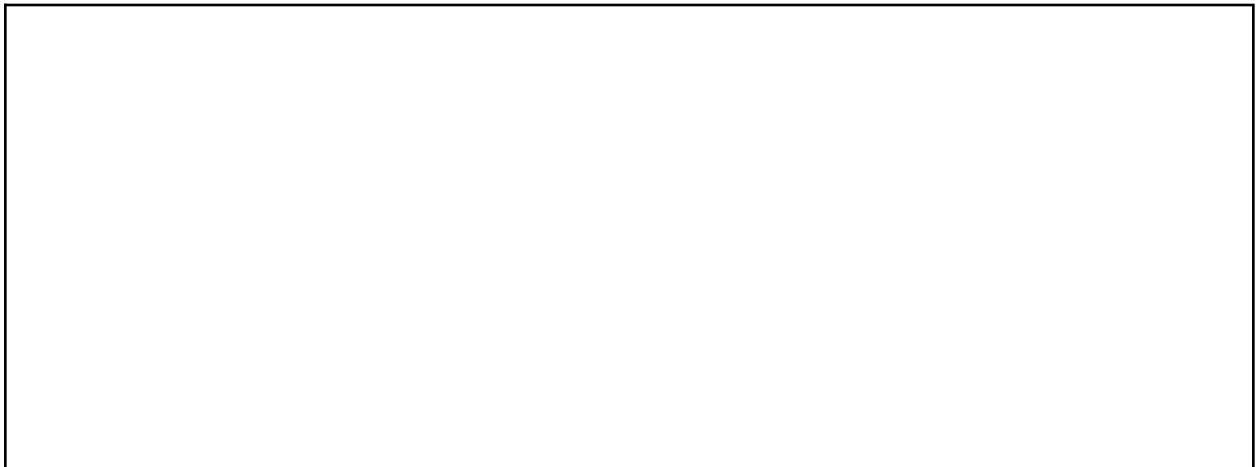
    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = torch.sigmoid(self.fc3(x))
        return x

model = SimpleNetwork()
criterion = nn.BCELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
```

1.) Draw the computation graph for the network (either Left to Right or Top to Bottom is fine)  
(**Note:** Please refer to the final page for a lookup table of functions and their derivatives)







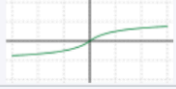
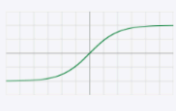




2.) Write the output of the neural network ( $\hat{y}$ ) as a single statement of nested functions.  
(**Note:** Please refer to the final page for a lookup table of functions and their derivatives)



3.) Write the chain of derivatives performed to calculate the gradient of the bias in 1st the hidden layer.

4.) Give the equation you'd use to calculate the gradients. (**Note:** Please refer to the final page for a lookup table of functions and their derivatives)

5.) What type of task is this network probably attempting to solve? How do you know?

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	$f'(x) = 1 - f(x)^2$
ElliotSig Softsign		$f(x) = \frac{x}{1 +  x }$	$f'(x) = \frac{1}{(1 +  x )^2}$
Square Nonlinearity (SQNL)		$f(x) = \begin{cases} 1 & : x > 2.0 \\ x - \frac{x^2}{4} & : 0 \leq x \leq 2.0 \\ x + \frac{x^2}{4} & : -2.0 \leq x < 0 \\ -1 & : x < -2.0 \end{cases}$	$f'(x) = 1 \mp \frac{x}{2}$
Rectified linear unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Bipolar rectified linear unit (BRELU)		$f(x_i) = \begin{cases} ReLU(x_i) & \text{if } i \bmod 2 = 0 \\ -ReLU(-x_i) & \text{if } i \bmod 2 \neq 0 \end{cases}$	$f'(x_i) = \begin{cases} ReLU'(x_i) & \text{if } i \bmod 2 = 0 \\ -ReLU'(-x_i) & \text{if } i \bmod 2 \neq 0 \end{cases}$
Leaky rectified linear unit (Leaky ReLU)		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric rectified linear unit (PReLU)		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

Name	Equation	Derivative
Mean Squared Error (MSE)	$\mathcal{L} = \frac{1}{n} \sum (y - \hat{y})^2$	$\frac{\partial \mathcal{L}}{\partial y} = -2(y - \hat{y})$
Mean Absolute Error (MAE)	$\mathcal{L} = \frac{1}{n} \sum  y - \hat{y} $	$\frac{\partial \mathcal{L}}{\partial y} = -\text{sign}(y - \hat{y})$
Huber Loss	$L = 0.5(y - \hat{y})^2$ if $ y - \hat{y}  \leq \delta$ ; else $\delta( y - \hat{y}  - 0.5\delta)$	Piecewise gradient
Binary Cross Entropy (BCE)	$\mathcal{L} = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]$	$\frac{\partial \mathcal{L}}{\partial y} = \frac{\hat{y} - y}{y(1 - \hat{y})}$
BCE with Sigmoid	$\mathcal{L} = -[y \log(\sigma(z)) + (1 - y) \log(1 - \sigma(z))]$	$\frac{\partial \mathcal{L}}{\partial z} = \sigma(z) - y$
Categorical Cross Entropy	$\mathcal{L} = -\sum_i y_i \log(\hat{y}_i)$	$\frac{\partial \mathcal{L}}{\partial y_i} = \hat{y}_i - y_i$
KL Divergence	$\mathcal{L} = \sum y \log\left(\frac{y}{\hat{y}}\right)$	$\frac{\partial \mathcal{L}}{\partial y} = -\frac{y}{\hat{y}}$
Poisson Loss	$\mathcal{L} = \hat{y} - y \log(\hat{y})$	$\frac{\partial \mathcal{L}}{\partial y} = 1 - \frac{y}{\hat{y}}$
Hinge Loss	$\mathcal{L} = \max(0, 1 - y\hat{y})$	$\partial \mathcal{L} / \partial \hat{y} = -y$ if $y\hat{y} < 1$ ; else 0
Squared Hinge Loss	$\mathcal{L} = \max(0, 1 - y\hat{y})^2$	$\partial \mathcal{L} / \partial \hat{y} = -2y(1 - y\hat{y})$ if $y\hat{y} < 1$ ; else 0