Name:

NetID:

0.) Practice packet submission (5 pts.)

1.) How are gradient descent and backprop related? (2 pt.)

GD updates the weights

Backprop uses the chain rule to find the gradients for GD

2.) What is the difference between a regression task and a classification task? Give a real-world example of each (3 pts.)

Regression → real number output

→ house sale price predictor

Classification → label output

→ Iris species predictor

3.) When may we prefer a soft-margin classifier over a hard-margin classifier? (1 pt.)

Soft can be more robust to outliers by allowing for misclassification during training time

4.) What happens if you were to change the location of a support vector? (1 pt.)

The decision boundary also typically changes

5.) What is the kernel trick? (2 pts.)

SVMs use this to project data into higher dimensions to find non-linear decision boundaries in lower dimensions

6.) Why don't we typically use gradient descent for linear regressions? (2 pts.)

Because we can just use OLS to directly compute the optimal weights

7.) What is the gradient we're descending when we use gradient descent? What are we trying to optimize and what do we take the partial derivatives with respect to to do so? (3 pts.)

Loss function wrt weights/biases
we want to optimize our model
by finding weights to minimize
loss so we partially differentiate
loss wrt every parameter

8.) What are the differences between supervised and unsupervised learning? (2 pts.)

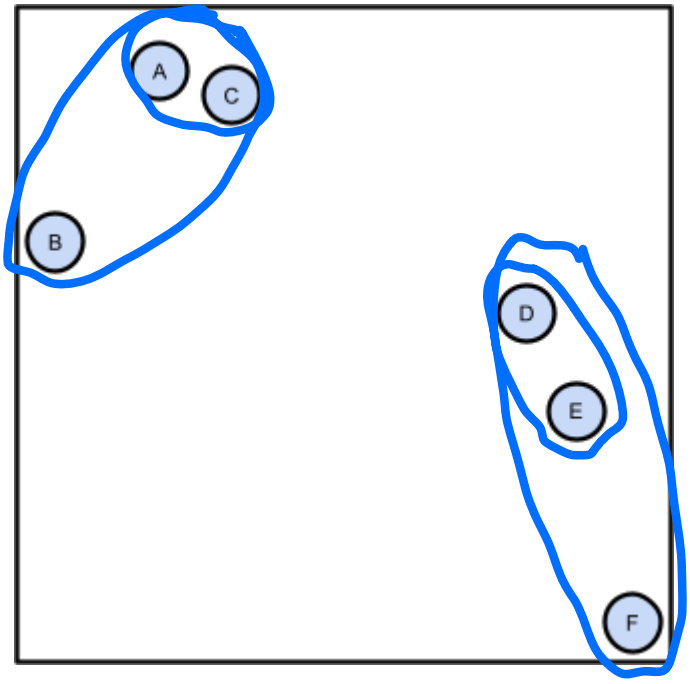
Supervised → ground-truth answers
known and we train our model
to fit these

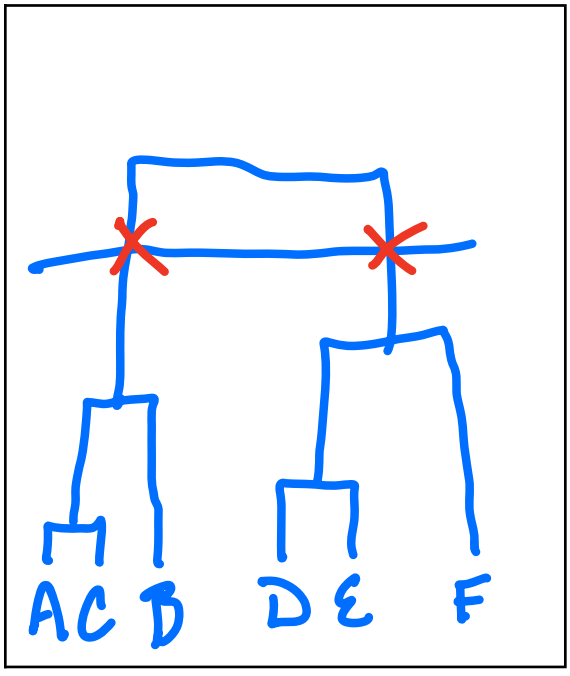
Unsupervised → no gt, finding
patterns by comparison of data
to eachother

9.) What are centroids in k-means clustering? (1 pt.)

the center of each cluster. Typically not a real sample in our data beyond the first iteration

10.) Given the data points, draw the dendrogram that would be created using agglomerative hierarchical clustering and ward linkage, then draw a line on the dendrogram to create 2 clusters (3 pts.)

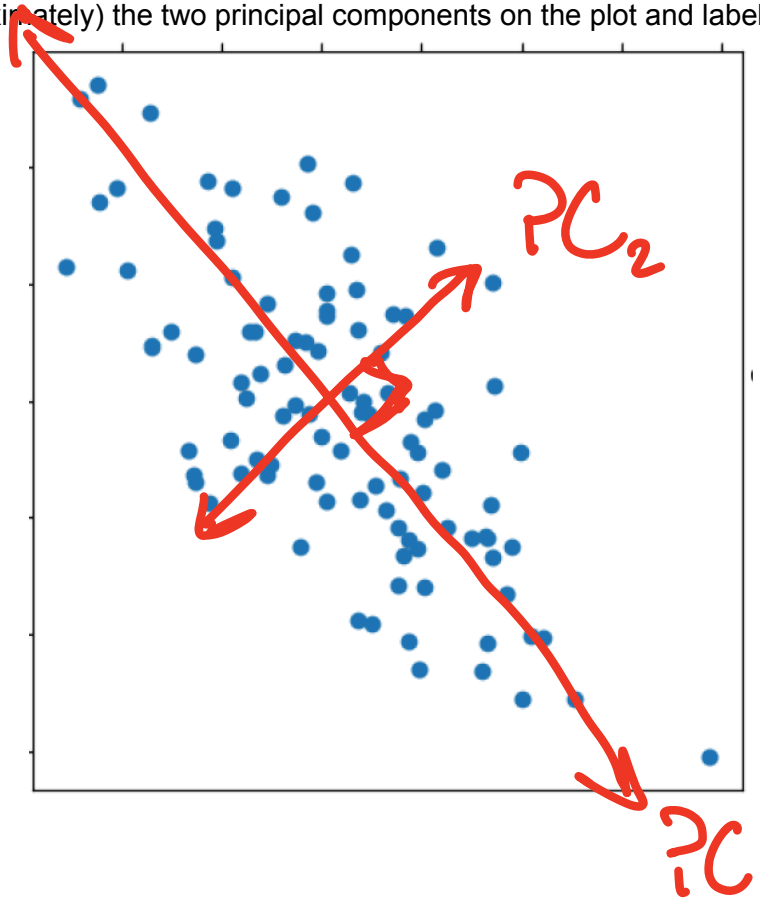




11.) In your own words, what is the curse of dimensionality? (2 pts.)

As we add features/axes to our data it grows more and more sparse so we need exponentially more data to cover the same "space"

12.) Draw (approximately) the two principal components on the plot and label them (2 pts.)



13.) Why are the principal components always orthogonal to each other? What do we do with the covariance matrix to ensure this happens? (2 pts.)

If they weren't then we'd have Covariance / info overlap.

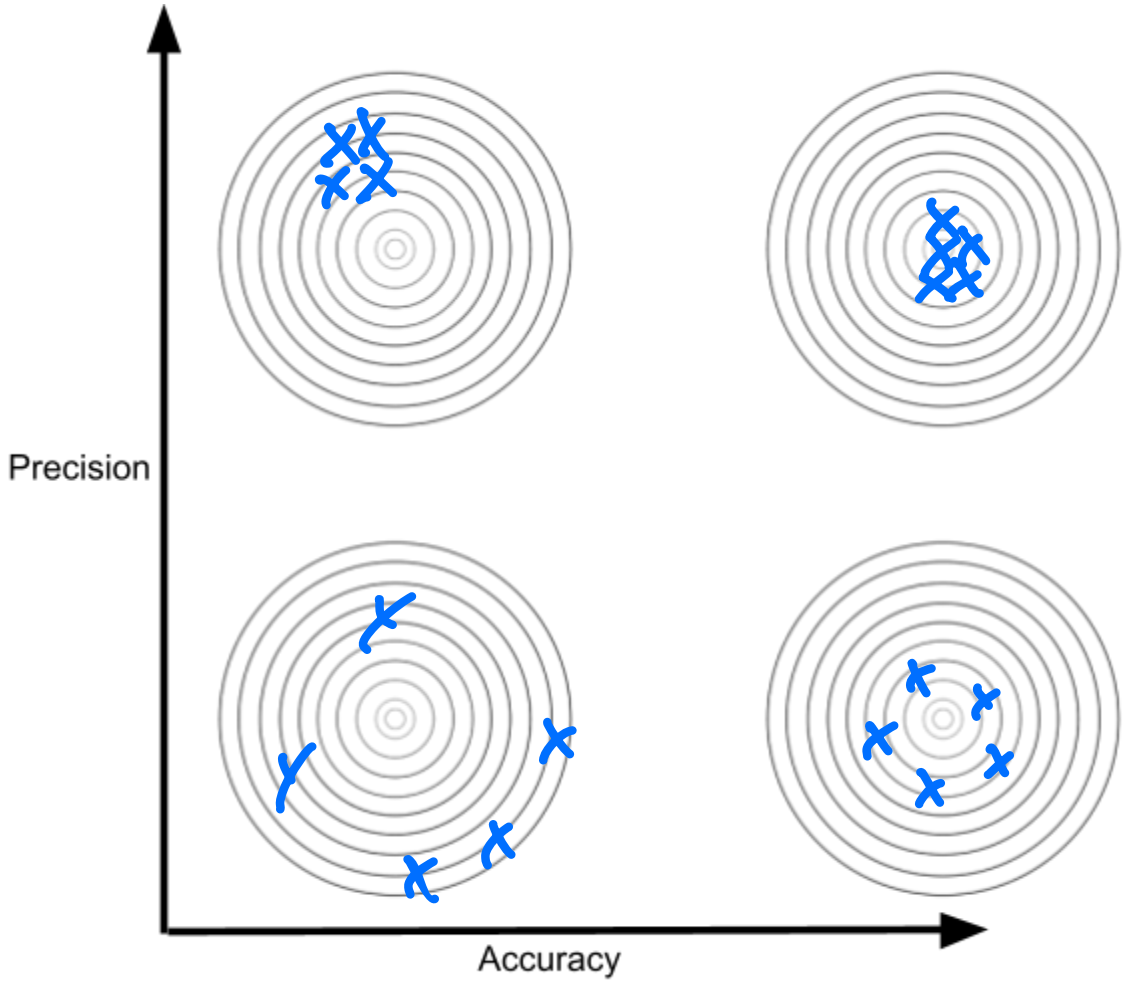
We solve the equation to be equal to 0 and use the identity Matrix

14.) Given the following results, label each as Type 1 or Type 2 error (Multiple Choice) (5 pts.)

Result	Type 1	Type 2
Your twin is allowed through TSA with your ID	X	
You are diagnosed with a rare disease but you don't have it	X	
A rescue victim is declared dead but is actually alive		X
There's a fire in the building but the alarm doesn't go off		X
You reject the null hypothesis when it's actually true		X

Little unclear what target class is

15.) Draw Xs on each of the four targets relating to their position on the X and Y axes (4 pts.)



16.) Write the following statement (1 pt.)

“I must always split my data into training and testing and must not train on the testing data”

No !!

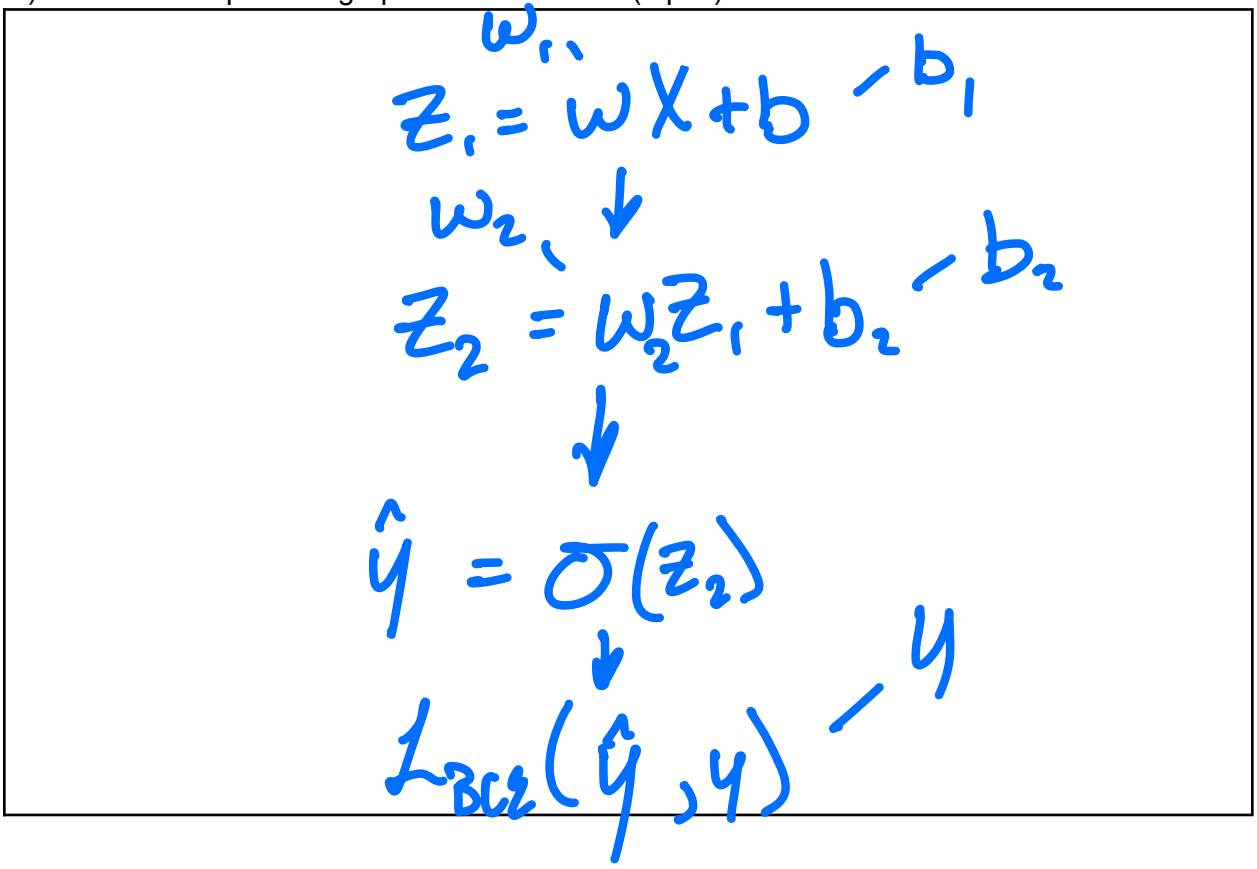
17.) Given the pytorch code below, answer the following questions. Please refer to the loss and activation functions lookup table on the next page.

```
# Define the neural network
class SimpleNN(nn.Module):
    def __init__(self):
        super(SimpleNN, self).__init__()
        self.hidden = nn.Linear(3, 2)
        self.output = nn.Linear(2, 1)

    def forward(self, x):
        hidden = self.hidden(x)
        out = self.output(hidden)
        sig = torch.sigmoid(sig)
        return hidden, out, sig

# Initialize the model, loss, and optimizer
model = SimpleNN()
criterion = nn.BCEWithLogitsLoss()
optimizer = optim.Adam(model.parameters(), lr=0.01)
```

a.) Draw the computation graph for the network. (4 pts.)

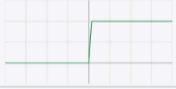
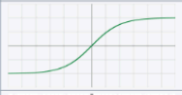
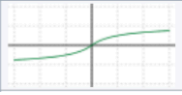


b.) Give the derivative chain for calculating the gradient of the bias in the first layer. (4 pts.)

$$\frac{\partial \mathcal{L}}{\partial b} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_2} \frac{\partial z_2}{\partial z_1} \frac{\partial z_1}{\partial b_1}$$

c.) Give the equation you'd use to actually calculate the gradients. (4 pts.)

$$\frac{\partial \mathcal{L}}{\partial b_1} = (\hat{y} - y)(w_2)(1)$$

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		$f(x) = \frac{x}{1 + \lvert x \rvert}$	$f'(x) = \frac{1}{(1 + \lvert x \rvert)^2}$

Name	Equation	Derivative
Mean Squared Error (MSE)	$\mathcal{L} = \frac{1}{n}\sum(y - \hat{y})^2$	$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -2(y - \hat{y})$
Mean Absolute Error (MAE)	$\mathcal{L} = \frac{1}{n}\sum\lvert y - \hat{y} \rvert$	$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\text{sign}(y - \hat{y})$
Huber Loss	$L = 0.5(y - \hat{y})^2$ if $\lvert y - \hat{y} \rvert \leq \delta$; else $\delta(\lvert y - \hat{y} \rvert - 0.5\delta)$	Piecewise gradient
Binary Cross Entropy (BCE)	$\mathcal{L} = -[y\log(\hat{y}) + (1-y)\log(1-\hat{y})]$	$\frac{\partial \mathcal{L}}{\partial \hat{y}} = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})}$
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	$\mathcal{L} = -\sum_i y_i \log(\hat{y}_i)$	$\frac{\partial \mathcal{L}}{\partial \hat{y}_j} = \hat{y}_j - y_j$
KL Divergence	$\mathcal{L} = \sum y \log\left(\frac{y}{\hat{y}}\right)$	$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\frac{y}{\hat{y}}$
Poisson Loss	$\mathcal{L} = \hat{y} - y\log(\hat{y})$	$\frac{\partial \mathcal{L}}{\partial \hat{y}} = 1 - \frac{y}{\hat{y}}$

Answer the following questions about homework03, which was using clustering and SVMs for image segmentation.

18.) What is image segmentation? (2 pt.)

Split image by regions of interest

19.) Training the SVMs was very very slow, even after using PCA. Why do you think that is? What properties about the SVM model result in a quadratic complexity? (3 pts.)

SVM does pairwise calculations in the dual formulation leading to quadratic complexity

20.) For the SVM, we created features for our image (using the default patch size of 5) resulting in a (249900, 78) training matrix. How was that 78 calculated? Spitball some ideas for features that could have been better for our image than what we did. (2 pts.)

Each pixel has [R, G, B]

5 x 5 patch size = 25

25 pixels x 3(r, g, b) = 75

+ middle pixel rgb = 78

Average color maybe?

Bonus.) I've tried to show more code in class, but I mostly just kind of read over it and run it. Do you think it'd be more useful for you if we wrote some of it instead of just reading? (1 bonus pt.)

Let Tom do it

Lectures please!