CS 189/289A, Midterm

Fall 2025

Name: _____

Email: _____@berkeley.edu

Student ID: _____

Room where you're taking the exam: _____

Name and SID of the person on your right: _____

Name and SID of the person on your left: _____

Instructions:

This exam consists of **69 points** spread out over **7 questions** and the Honor Code certification. The exam must be completed in **110 minutes** unless you have accommodations supported by a DSP letter.

Note that you should **select one choice** for questions with **circular bubbles**. There is always at least one correct answer. Please **fully** shade in the circle to mark your answer. For all math questions, **please simplify your answer**. Please also **show your work** if a large box is provided. For all coding questions, you may use commas and/or one or more function calls in each blank.

For all Python questions, you may assume `Pandas` has been imported as `pd`, `NumPy` has been imported as `np`, and `Plotly Express` has been imported as `px`.

You MUST write your Student ID number at the top of each page.

Honor Code [1 Pt]:

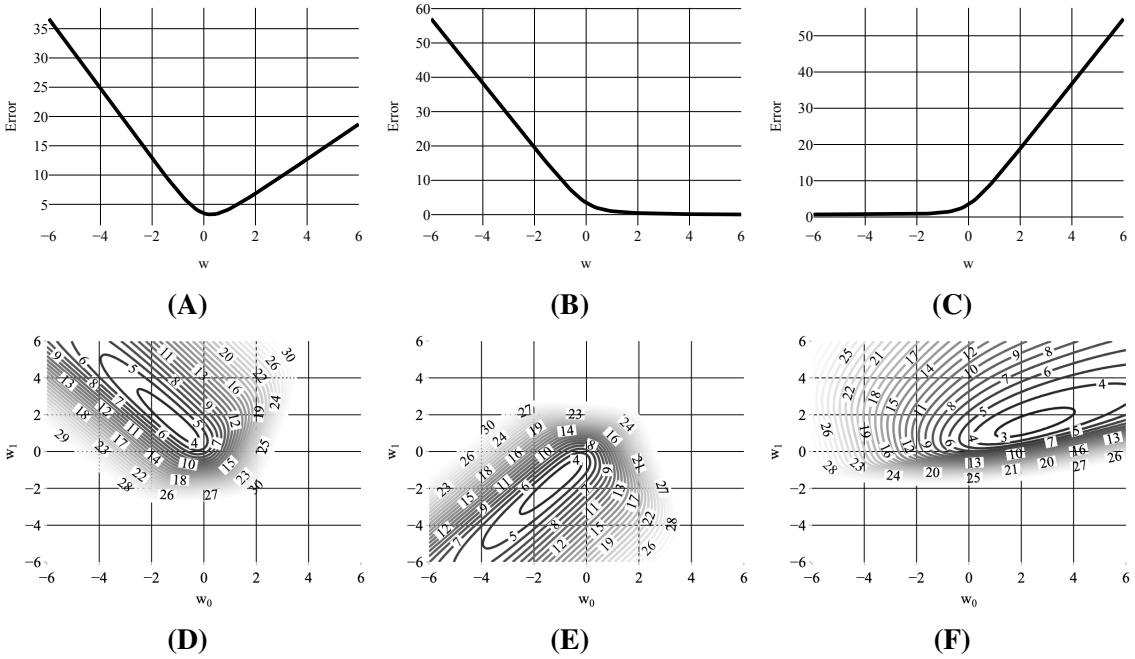
As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others. I am the person whose name is on the exam, and I completed this exam in accordance with the Honor Code.

Signature: _____

This page has been intentionally left blank.

1 Matching the Loss Surface

The following plots depict the error surfaces for several models on different datasets. The contour plots show the contours of the error surface. Use these plots to answer the questions below.



(A) **(B)** **(C)**

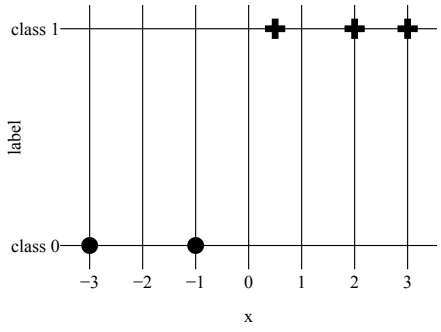
(D) **(E)** **(F)**

(a) [1 Pt] Which of the following most accurately describes the loss surface in Figure **(A)**?

- ◯ This is the loss for a least squares regression problem.
- ◯ This is the loss of a linearly separable classification problem.
- ◯ Significant L_2 regularization is needed for this loss surface.
- ◯ **This is the loss of the classification problem.**

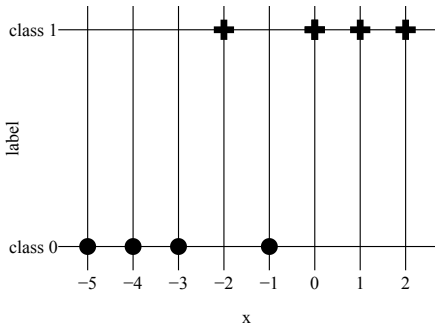
(b) [2 Pts] Which of the above loss functions corresponds to the cross entropy loss for a logistic regression model of the form $y(x) = \sigma(x \times w)$ where $x, w \in \mathbb{R}$ with the following data:

- ◯ (A)
- ◯ **(B)**
- ◯ (C)
- ◯ (D)
- ◯ (E)
- ◯ (F)



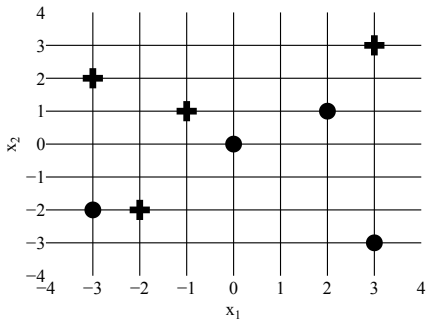
(c) [2 Pts] Which of the above loss functions corresponds to the cross entropy loss for a logistic regression model of the form $y(x) = \sigma\left(x \times w_1 + w_0\right)$ where $x, w_1, w_0 \in \mathbb{R}$ with the following data:

◯ (A)
◯ (B)
◯ (C)
◯ (D)
◯ (E)
◯ **(F)**



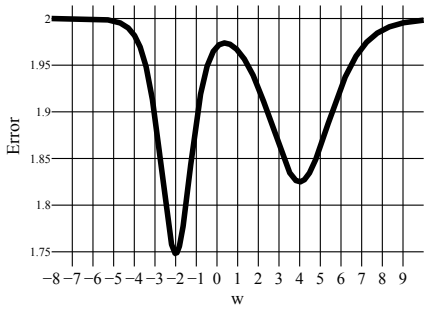
(d) [2 Pts] Suppose you fit a logistic regression model of the form $y(\mathbf{x}) = \sigma\left(\mathbf{w}^\top \mathbf{x}\right)$ where $\mathbf{x}, \mathbf{w} \in \mathbb{R}^2$ without any regularization to the following data. Which is the most likely loss function for this dataset?

◯ (A)
◯ (B)
◯ (C)
◯ **(D)**
◯ (E)
◯ (F)

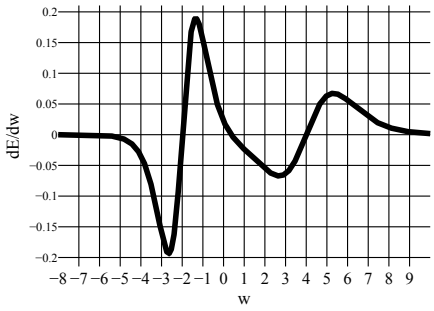


2 Gradient Descent

The following plots depict the error surface and its derivative for some unknown model, error function, and dataset. The error surface is flat (zero gradient) outside of the domain of the plot.



(a) Error



(b) Gradient

(a) [1 Pt] How many critical points does this error function have between $w = -4$ and $w = 7$?

◯ 1 ◯ 2 ◯ **3** ◯ 4

(b) [1 Pt] If you apply gradient descent starting at $w^{(0)} = 2$ using a learning rate of 100 for many iterations, you will likely find:

◯ $w^{(\tau)}$ **diverges to values greater than 6.** ◯ $w^{(\tau)}$ gradually approaches 4.

◯ $w^{(\tau)}$ gradually approaches 2.9. ◯ $w^{(\tau)}$ gradually approaches -2.

(c) [1 Pt] When applying stochastic gradient descent with batch size 1, the calculated gradient:

◯ Always points in the direction of steepest ascent on the error surface.

◯ Always points in the direction of steepest descent on the error surface.

◯ Is always zero at the true minimum of the error surface.

◯ **May be non-zero at the true minimum of the error surface.**

(d) [1 Pt] When applying gradient descent to the error function $E[w] = w^2$ starting at $w^{(0)} = 1$, please select all learning rates that will guarantee convergence.

☐ 0.8001 ☐ **0.9** ☐ 1 ☐ 1.1

3 Bias–Variance Decomposition

Recall the bias-variance decomposition for the expected mean squared error:

$$\mathbb{E}\big[(\mathbf{t} - f_{\mathbf{w}}(\mathbf{x}))^2\big] = \underbrace{\mathbb{E}\big[(\mathbf{t} - h(\mathbf{x}))^2\big]}_{\text{(i)}} + \big(h(\mathbf{x}) - \mathbb{E}[\mathbf{f}_{\mathbf{w}}(\mathbf{x})]\big)^2 + \underbrace{\mathbb{E}\big[(f_{\mathbf{w}}(\mathbf{x}) - \mathbb{E}[f_{\mathbf{w}}(\mathbf{x})])^2\big]}_{\text{(ii)}},$$

where we have the following variables:

$\mathbf{t}$: Observed (noisy) target, defined by $\mathbf{t} = h(\mathbf{x}) + \epsilon$, where ϵ is zero-mean noise.

$f_{\mathbf{w}}(\mathbf{x})$: Model prediction. It varies across different training datasets.

$h(\mathbf{x})$: True function. It is the underlying noiseless true target value for an input $\mathbf{x}$.

(a) [2 Pts] Select the most fitting description for each term of the bias-variance decomposition.

(i) Select the label corresponding to $\mathbb{E}\big[(t - h(x))^2\big]$.

◯ Bias ◯ Variance ◯ **Noise**
◯ Bias^2 ◯ Variance^2

(ii) Select the label corresponding to $\mathbb{E}\big[(f_w(x) - \mathbb{E}[f_w(x)])^2\big]$.

◯ Bias ◯ **Variance** ◯ Noise
◯ Bias^2 ◯ Variance^2

Suppose we're only interested in the family of functions $f_{\mathbf{w}}(\mathbf{x})$ that are linear models, that is

$$f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}),$$

where $\boldsymbol{\phi}(\mathbf{x})$ denotes a basis of the features.
We fit a collection of linear models to a dataset $\mathcal{D} = \{(t_n, \mathbf{x}_n)\}_{n=1}^N$ by minimizing the LASSO regression objective:

$$\mathbf{w}^\star = \arg\min_{\mathbf{w}} \frac{1}{2}\sum_{n=1}^{N} \big(t_n - \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n)\big)^2 + \lambda \|\mathbf{w}\|_1.$$

Suppose we start with the following choice of hyperparameters:

- $\boldsymbol{\phi}_0(\mathbf{x}) = \mathbf{x}$ as our basis function.
- $\lambda_0 = 1$ to control the strength of our L_1 regularization.

(b) [3 Pts] Suppose we train $f_{\mathbf{w}}(\mathbf{x})$ with $\lambda_{\text{new}} < \lambda_0$.

(i) What is most likely to happen to $\big(h(x) - \mathbb{E}[f_w(x)]\big)^2$?

◯ Increases ◯ Remains the same ◯ **Decreases**

(ii) Briefly justify your answer to (i). *Partial credit will not be given for justifying an incorrect answer to (i).*

Solution: $\lambda_{\text{new}} < \lambda_0$ indicates that regularization is decreased. $\big(h(x) - \mathbb{E}[f_w(x)]\big)^2$ is Bias^2, which decreases as regularization decreases.

(c) [3 Pts] Suppose we train $f_{\mathbf{w}}(\mathbf{x})$ with a very simple basis function, $\phi_{\text{new}}(\mathbf{x}) = \begin{bmatrix}1\end{bmatrix}$.

(i) What is most likely to happen to $\mathbb{E}\big[(f_w(x) - \mathbb{E}[f_w(x)])^2\big]$?

◯ Remains the same ◯ Increases ◯ **Decreases**

(ii) Briefly justify your answer to (i). *Partial credit will not be given for justifying an incorrect answer to (i).*

Solution: $\mathbb{E}\big[(f_w(x) - \mathbb{E}[f_w(x)])^2\big]$ is Variance. $\phi_{\text{new}}(\mathbf{x}) = \begin{bmatrix}1\end{bmatrix}^\top$ means that we have $f_{\mathbf{w}}(\mathbf{x}) = w \times 1 = w$ for any input $\mathbf{x}$. Since this model is much less expressive than the model with $\phi(\mathbf{x}) = \mathbf{x}$, its predictions vary less across different training datasets.

4 Matcha Munchies

Oski is building a classifier to predict whether a matcha dessert is "popular" ($t = 1$) or "not popular" ($t = 0$). Each input $\mathbf{x}_n$ has two attributes: $x_1 =$ bitterness and $x_2 =$ sweetness. We use logistic regression to predict the probability that $t_n = 1$ given $\mathbf{x}_n$ by estimating the probability:

$$y_n \;=\; p(t = 1 \mid \mathbf{x}_n, \mathbf{w}) \;=\; \sigma(\mathbf{x}_n^\top \mathbf{w}) \;=\; \frac{1}{1 + e^{-\mathbf{x}_n^\top \mathbf{w}}}.$$

Consider the following 4 training points:

n	x_1	x_2	t
1	1	1	1
2	1	-1	0
3	-1	1	0
4	-1	-1	1

(a) [1 Pt] Is this dataset linearly separable?

◯ Yes ◯ **No**

(b) [3 Pts] We observe a dataset $\mathcal{D} = \{(\mathbf{x}_n, t_n)\}_{n=1}^N$ of IID samples, where each label $t_n \sim \text{Bernoulli}(p_n)$ with probability of success $p_n = \sigma(\mathbf{w}^\top \mathbf{x}_n)$. Write the log-likelihood of the data $\log p(\mathcal{D} \mid \mathbf{w})$ and expand it into a sum over the N examples. Your final answer can be in terms of the Bernoulli parameters: $\{p_n\}$.

Solution:

$$\log p(\mathcal{D} \mid \mathbf{w}) \;=\; \log \prod_{n=1}^{N} \underbrace{p_n^{t_n}(1 - p_n)^{1 - t_n}}_{\text{PMF of Bernoulli}(t_n; p_n)}$$

$$\log p(\mathcal{D} \mid \mathbf{w}) \;=\; \sum_{n=1}^{N} \underbrace{\big[\, t_n \log p_n + (1 - t_n) \log(1 - p_n) \,\big]}_{\text{Log-likelihood per example}}$$

(c) [3 Pts] We select $\mathbf{w}$ by minimizing the negative log-likelihood from the previous part and then use the following decision function to make classification decisions:

$$y = \begin{cases} 1 & \text{if } \sigma(\mathbf{w}^\top \mathbf{x}) \geq T \\ 0 & \text{otherwise} \end{cases}.$$

Indicate whether the following statements are True or False if we **increase** the decision threshold T?

True	False	Statement
		The True Positive Rate (TPR) and False Positive Rate (FPR) both tend to decrease.
		Precision will increase.
		Recall will decrease or stay the same.

Solution:

True	False	Statement
X		The true positive rate (TPR) and false positive rate (FPR) both tend to decrease.
	X	The precision necessarily increases.
X		Recall will decrease or stay the same.

(d) [4 Pts] Select True or False for the following statements about logistic regression.

True	False	Statement
		Logistic regression is an example of a discriminative model.
		Cross-entropy loss is convex in $\mathbf{w}$ for logistic regression.
		Minimizing squared error is equivalent to minimizing cross-entropy on the same data.
		Logistic regression always produces a linear decision boundary in the input space.

Solution:

True	False	Statement
X		Logistic regression is an example of a discriminative model.
X		Cross-entropy loss is convex in $\mathbf{w}$ for logistic regression.
	X	Minimizing squared loss is equivalent to minimizing cross-entropy on the same data.
X		Logistic regression always produces a linear decision boundary in the feature space.

(e) [4 Pts] Select True or False for the following statements about multiclass classification.

True	False	Statement
		The one-vs-rest approach trains one classifier for each pair of classes.
		The one-vs-one approach trains a single classifier that handles all classes simultaneously.
		The softmax function generalizes the logistic sigmoid to handle multiple classes.
		In the softmax model, predicted class probabilities always sum to one.

Solution:

True	False	Statement
	X	The one-vs-rest approach trains one classifier for each pair of classes.
	X	The one-vs-one approach trains a single classifier that handles all classes simultaneously.
X		The softmax function generalizes the logistic sigmoid to handle multiple classes.
X		In the softmax model, predicted class probabilities always sum to one.

Softmax and one-vs-rest/one-vs-one setups are described on Lecture 11 slides (Multiclass Classification section).

5 Linear Regression with a Nonzero Mean Gaussian Prior

We know that least-squares regression arises as the *Maximum Likelihood Estimator (MLE)* under Gaussian noise, and ridge regression corresponds to a *Maximum A Posteriori (MAP)* estimator with a zero-mean Gaussian prior on the weights. Here, we consider a natural extension: **a Gaussian prior with *non-zero mean***.

Suppose our data are generated as

$$p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}) \sim \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}),$$

and the prior on $\mathbf{w}$ is

$$\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \tau^2\mathbf{I}),$$

where $\boldsymbol{\mu} \in \mathbb{R}^D$ is a fixed prior mean vector.

(a) [2 Pts] Write the log-likelihood $\ell(\mathbf{w}) = \ln p(\mathbf{t} \mid \mathbf{X}, \mathbf{w})$, ignoring additive constants independent of $\mathbf{w}$.

Solution:

Solution:

$$\ell(\mathbf{w}) = -\frac{1}{2\sigma^2} \|\mathbf{t} - \mathbf{X}\mathbf{w}\|^2$$

(b) [4 Pts] Write the log-prior $\ln p(\mathbf{w})$ under $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \tau^2\mathbf{I})$. Then form the log-posterior

$\ln p(\mathbf{w} \mid \mathbf{X}, \mathbf{t})$ up to additive constants.

Solution:

Solution:

$$\ln p(\mathbf{w}) = -\frac{1}{2\tau^2} \|\mathbf{w} - \mu\|^2$$

Thus, the log-posterior is

$$\ln p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}) \ \propto \ -\frac{1}{2\sigma^2} \|\mathbf{t} - \mathbf{X}\mathbf{w}\|^2 \ - \ \frac{1}{2\tau^2} \|\mathbf{w} - \mu\|^2.$$

(c) [3 Pts] Show that maximizing the log-posterior is equivalent to solving

$$\min_{\mathbf{w}} \ \|\mathbf{t} - \mathbf{X}\mathbf{w}\|^2 + \lambda \|\mathbf{w} - \boldsymbol{\mu}\|^2, \qquad \lambda = \tfrac{\sigma^2}{\tau^2}.$$

Solution:

Solution: Multiply the log-posterior by $-2\sigma^2$ and drop constants:

$$\arg\max_{\mathbf{w}} \ln p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}) \equiv \arg\min_{\mathbf{w}} \|\mathbf{t} - \mathbf{X}\mathbf{w}\|^2 + \tfrac{\sigma^2}{\tau^2} \|\mathbf{w} - \mu\|^2.$$

(d) [2 Pts] Interpret the effect of the prior mean $\boldsymbol{\mu}$. How does it compare to ridge regression?

Solution:

Solution: In ridge regression, coefficients are shrunk toward $\mathbf{0}$. Here, they are shrunk toward μ. The prior mean acts like a "default guess" for $\mathbf{w}$; if the data are weak, the estimate will lean more heavily toward μ.

(e) [1 Pt] What does R-squared measure, and why is it significant?

- ◯ It measures the average absolute error of predictions; higher R^2 means lower errors.
- ◯ **It measures how much of the variance in the target variable the model explains compared to a baseline that always predicts the mean.**
- ◯ It measures the correlation between residuals and predictions; higher R^2 means less correlation.
- ◯ It measures model bias; higher R^2 indicates less overfitting.

6 Another Fashion Classifier

Just as you did in HW1, you are building a **FashionMNIST** classifier. You downloaded the FashionMNIST dataset and have the access to the following two `NumPy` arrays:

1. `images` – A set of 6000 28×28 grayscale images stored in a `NumPy` array of shape `(6000,28,28)` with `dtype` of `np.uint8`.
2. `labels` – The corresponding set of labels stored in a 1D `NumPy` array of length 6000 with `dtype` of `str`.

(a) [2 Pts] You create the following `Pandas DataFrame`:

```
df = pd.DataFrame({
        "image": images.tolist(),
        "label": labels})
df["image"] = df["image"].apply(lambda x: np.asarray(x))
```

For each snippet below, select the **data type** of the resulting object.

(i) `df["label"].value_counts()`

- ◯ `Dictionary`
- ◯ `numpy.ndarray`
- ◯ `pandas.Series`
- ◯ `pandas.DataFrame`

(ii) `np.stack(df["image"].values)`

- ◯ `pandas.Index`
- ◯ `numpy.ndarray`
- ◯ `pandas.Series`
- ◯ `pandas.DataFrame`

(b) [3 Pts] Continuing with the `Pandas DataFrame` created in part (a), create a `DataFrame` that only contains data samples from the class `"Ankle boot"`. Then print the first row of the new `DataFrame`.

```
# Construct ankle_df with only rows with label "Ankle boot"
ankle_df = df[____(i)____][["image", "label"]]
# Print only the first row
____(ii)____
```

(i) Fill in blank `(i)`:

Solution:
`labels == "Ankle boot"` or `df["label] == "Ankle boot"`

(ii) Fill in blank `(ii)`:

Solution: `print(df.head(0))`

(c) [3 Pts] The following code attempts to train and test an `MLPClassifier` on the Fashion-MNIST dataset from part (a) using an 80/20 split with standardized image features.

You notice that the model performance is poor. Your instructors tell you that there are **three issues** with this implementation. Identify each line number containing a bug and provide the corrected code. (*Hint: The code executes successfully; none of the bugs are syntax errors.*)

```
1 from sklearn.model_selection import train_test_split
2 from sklearn.preprocessing import StandardScaler
3 from sklearn.neural_network import MLPClassifier
4
5 X = np.stack(df["image"].values)
6 y = df["label"].values
7 X_test, y_test, X_train, y_train = train_test_split(X, y,
  ↪ test_size=0.2)
8 scaler = StandardScaler()
9 X_train_sc = scaler.fit_transform(X_train)
10 X_test_sc  = scaler.fit_transform(X_test)
11 model = MLPClassifier(hidden_layer_sizes=(256,),
  ↪ max_iter=200, random_state=42)
12 model.fit(X_train, y_train)
13 y_pred = model.predict(X_test_sc)
14 acc = np.mean(y_pred == y_test)
15 print(f"Test accuracy: acc:.3f")
```

Line #	Updated Code

Solution:

Line #	Corrected Code
7	`X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)`
10	`X_test_sc = scaler.transform(X_test)`
12	`model.fit(X_train_sc, y_train)`

Explanation: (7) The train and test sets were reversed — training should come first to ensure we're training with enough data (80% of the total data as desired).
(10) The `StandardScaler` should only be fit on the training data and then applied to the test data.
(12) The model should be trained using the standardized training features.

(d) [3 Pts] The following code implements K-Means clustering to create K clusters. Assume `X` is an $(N \times D)$ `NumPy` array whose n^{th} row is the transpose of the n^{th} data point $\mathbf{x_n} \in \mathbb{R}^D$.

After running the code, you notice that all of the points are assigned to one cluster. Your instructors tell you that there are **three issues** with this implementation. Identify each line number containing a bug and provide the corrected line. (*Hint: The code executes successfully; none of the bugs are syntax or broadcasting errors.*)

```
 1 import numpy as np
 2
 3 def kmeans_numpy(X, K, max_iters=100, tol=1e-4, rng=0):
 4     N, D = X.shape
 5     np.random.seed(rng)
 6     centers = np.zeros((K, D))
 7     labels  = np.zeros(N, dtype=int)
 8     it = 0
 9     prev_objective = np.inf
10     while it < max_iters:
11         dists = np.linalg.norm(X[:, np.newaxis] - centers,
   ↪ axis=2)
12         labels = np.argmin(dists, axis=1)
13         centers = np.array([X[labels == 0].mean(axis=0) for i
   ↪ in range(K)])
14         objective = np.sum(np.linalg.norm(X - centers[0],
   ↪ axis=1) ** 2)
15         if abs(prev_objective - objective) < tol:
16             break
17         prev_objective = objective
18         it += 1
19     return centers, labels, objective
```

Line #	Updated Code

Solution:

Line #	Corrected Code
6	`centers = X[np.random.choice(N, K, replace=False)]`
13	`centers = np.array([X[labels == i].mean(axis=0) for i in range(K)])`
14	`objective = np.sum(np.linalg.norm(X - centers[labels], axis=1) ** 2)`

Explanation: (6) Initializing all centers to zero causes every point to start in the same cluster. Sample K data points from `X` instead.

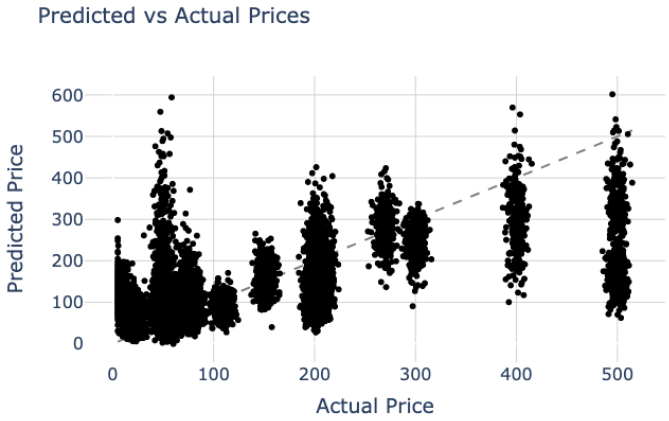
(13) Each cluster center must be recomputed using the points assigned to that cluster (`labels == i`), not just cluster 0.

(14) The objective should measure the total squared distance between each point and *its assigned cluster center*. Using only `centers[0]` ignores assignments and leads to degenerate behavior.

(e) [3 Pts] After training a regression model to predict product prices from FashionMNIST features, you compute the model's performance metrics and visualize predictions as follows:

```
y_train_pred = price_model.predict(X_train_sc)
y_test_pred = price_model.predict(X_test_sc)

fig = px.scatter(
    x=prices_test["Price"],
    y=y_test_pred,
    title="Predicted vs Actual Prices",
    labels="x": "Actual Price", "y": "Predicted Price"
)
fig.show()
```



Now, fill in the blanks below to calculate residuals and create a **residual plot**.

```
residuals = _______(i)_________
fig = px.scatter(
    x= _____(ii)_______,
    y= ______(iii)______,
    title="Residual Plot",
    labels="x": "Predicted Price", "y": "Residual"
)
fig.show()
```

(i) Fill in blank (i):

Solution: `prices_test["Price"] - y_test_pred`

(ii) Fill in blank (ii):

Solution: `y_test_pred`

(iii) Fill in blank (iii):

Solution: `residuals`

7 Oski's Two Boats

Oski is training for a rowing race. His speed $y \in \mathbb{R}$ depends on his effort $x \in \mathbb{R}$, measured in watts. During training, he uses two different boats, each with its own relationship between effort and speed. For **Boat A**, the relationship is $y \sim w_A x$, and for **Boat B**, the relationship is $y \sim w_B x$, where $w_A, w_B \in \mathbb{R}$ represent how efficiently each boat converts effort into speed.

Unfortunately, Oski mixed together his training data: He recorded his effort and speed, but not which boat he used for each session. He now wants to estimate w_A and w_B.

To model this, Oski uses a *mixture of two linear regression models*:

$$p(y_n \mid z_n = k) \ \sim \ \mathcal{N}(x_n w_k, \ \sigma^2), \quad p(z_n = k) = \pi_k, \quad k \in \{A, B\}$$

where z_n indicates the (unknown) boat used for data point n, and π_A and π_B are the prior probabilities of using Boat A or Boat B. Oski knows how often he trained with each boat.

(a) [3 Pts] Write down the expression for the single-sample likelihood $p(y_n \mid x_n)$ by marginalizing over the (unknown) boat variable $z_n \in \{A, B\}$. *Express your answer in terms of the distributions defined in the problem statement.*

$p(y_n \mid x_n) =$ _____

Solution:

$$\begin{aligned} p(y_n \mid x_n) &= \sum_{k \in \{A,B\}} p(z_n = k)\, p(y_n \mid x_n, z_n = k) \\ &= \pi_A \mathcal{N}(x_n w_A, \sigma^2) + \pi_B \mathcal{N}(x_n w_B, \sigma^2). \end{aligned}$$

(b) [3 Pts] **E-step:** Compute the responsibility $\gamma_{nk} = p(z_n = k \mid x_n, y_n)$, the probability that boat k (with $k \in \{A, B\}$) generated data point n. *Express your answer in terms of the distributions defined in the problem statement.*

$\gamma_{nk} =$ _____

Solution:

$$\begin{aligned} \gamma_{nk} &= p(z_n = k \mid x_n, y_n) \\ &= \frac{p(z_n = k)\, p(y_n \mid x_n, z_n = k)}{\sum_{j \in \{A,B\}} p(z_n = j)\, p(y_n \mid x_n, z_n = j)} \\ &= \frac{\pi_k \mathcal{N}(x_n w_k, \sigma^2)}{\pi_A \mathcal{N}(x_n w_A, \sigma^2) + \pi_B \mathcal{N}(x_n w_B, \sigma^2)}. \end{aligned}$$

(c) [1 Pt] In no more than two sentences, what do the responsibilities $\{\gamma_{nk}\}$ represent?

Solution: The posterior probability (soft assignment) that data point n was generated by component k (i.e., each boat).

(d) [1 Pt] Which statement about the EM algorithm is true?

- ◯ EM always finds the global optimum of the likelihood.
- ◯ **EM alternates between assigning soft labels and re-estimating parameters.**
- ◯ EM only applies when all clusters are Gaussian with equal variance.
- ◯ EM is unnecessary if we already know the responsibilities.

You are done with the midterm! Congratulations!

Use this page to draw your favorite CS 189/289A moment!