Homework 2

Decision Trees, Linear Regression, Logistic Regression[1]

CMU 10-701: Machine Learning (Fall 2020)

piazza.com/cmu/fall2020/10701/home
OUT: Wednesday , Sep 23rd, 2020

DUE: Wednesday, Oct 7th, 2020, 11:59pm
TAs: John Grace, Bhuvan Agrawal

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 2.1"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the Academic Integrity Section on the course site for more information: https://www.cs.cmu.edu/~epxing/Class/10701-20/about.html
- **Late Submission Policy:** See the late submission policy here: https://www.cs.cmu.edu/~epxing/Class/10701-20/about.html
- **Submitting your work:**
  - **Gradescope:** There will be two submission slots for this homework on Gradescope: Written and Programming.
    For the written problems such as short answer, multiple choice, derivations, proofs, or plots, we will be using the written submission slot. Please use the provided template. The best way to format your homework is by using the Latex template released in the handout and writing your solutions in Latex. However submissions can be handwritten onto the template, but should be labeled and clearly legible. If your writing is not legible, you will not be awarded marks. Each derivation/proof should be completed in the boxes provided below the question, **you should not change the sizes of these boxes** as Gradescope is expecting your solved homework PDF to match the template on Gradescope. If you find you need more space than the box provides you should consider cutting your solution down to its relevant parts, if you see no way to do this it please add an additional

[1]Compiled on Wednesday 23rd September, 2020 at 15:05

page a the end of the homework and guide us there with a 'See page xx for the rest of the solution'.

You are also required to upload your code, which you wrote to solve the final question of this homework, to the Programming submission slot. Your code may be ran by TAs so please make sure it is in a workable state.

Regrade requests can be made after the homework grades are released, however this gives the TA the opportunity to regrade your entire paper, meaning if additional mistakes are found then points will be deducted.

For multiple choice or select all that apply questions, shade in the box or circle in the template document corresponding to the correct answer(s) for each of the questions. For LaTeXusers, use ■ and ●for shaded boxes and circles, and don't change anything else.

1 Nearest Neighbors and Decision Trees [2 Points]

Let us try and classify data points in 2D Euclidean space. We are given n instances of such points: $P_1, P_2, \ldots, P_n$ and the corresponding category for each point $C_1, C_2, \ldots, C_n$ (where $C_1, C_2, \ldots, C_n$ take values from the set of all possible class labels). Under the k nearest neighbors classification scheme, each new element Q is simply categorized by a majority vote among its k nearest neighbors in instance space. The 1-NN is a simple variant of this which divides up the input space for classification purposes into a convex region (see Figure 1 below for the 1-NN decision boundaries under the Euclidean distance measure), each corresponding to a point in the instance set.

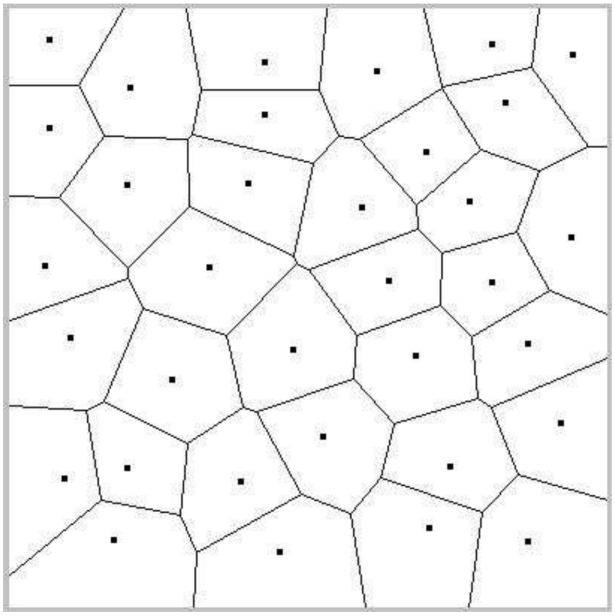


Figure 1: 1-NN decision boundaries under the Euclidean distance measure.

[2 Points] Is it possible to build a decision tree (with decisions at each node of the form "is $x > a$", "is $x < b$", "is $y > c$", or "is $y < d$" for any real constants a, b, c, d) which classifies exactly according to the 1-NN scheme using the Euclidean distance measure? If so, explain how. If not, explain why not.

2 Decision Trees and Information Gain [18 Points]

2.1 Warm-up

Consider the following dataset with 7 examples, each with 3 features denoted by A, B and C, respectively. Each data sample also has a label Y.

A	B	C	Y
1	1	0	0
1	1	2	1
1	0	0	0
0	1	1	1
0	0	2	0
0	1	1	0
0	0	0	0

Use the above dataset to answer the following questions.

1. **[1 Point]** What is the entropy of Y, $H(Y)$, in bits? Note that the unit *bits* implies that you need to use log base 2 in your calculations. (Please round your answer to the fourth decimal place, e.g. 0.1234).

2. **[1 Point]** What is the information gain of Y when you learn A in bits, $I(Y; A)$? (Please round your answer to the fourth decimal place, e.g. 0.1234).

3. **[1 Point]** What is the information gain of Y when you learn B in bits, $I(Y; B)$? (Please round your answer to the fourth decimal place, e.g. 0.1234).

4. **[1 Point]** What is the information gain of Y when you learn C in bits, $I(Y; C)$? (Please round your answer to the fourth decimal place, e.g. 0.1234).

5. [**2 Points**] Which feature, A, B or C, would your decision tree algorithm choose to branch on first, if we use the information gain as the splitting criterion? Briefly explain your decision.

6. [**2 Points**] If you apply the same decision tree algorithm until the above data are perfectly classified, what would the tree look like? Please draw your completed Decision Tree. Label the non-leaf nodes with which feature the tree will split on, the edges with the value of the attribute, and the leaf nodes with the classification decision. If two different variables can give the same information gain, pick the variable which comes first in the alphabet: A, B, C,... Your final tree should not perform any unnecessary splits (Where nothing is gained).

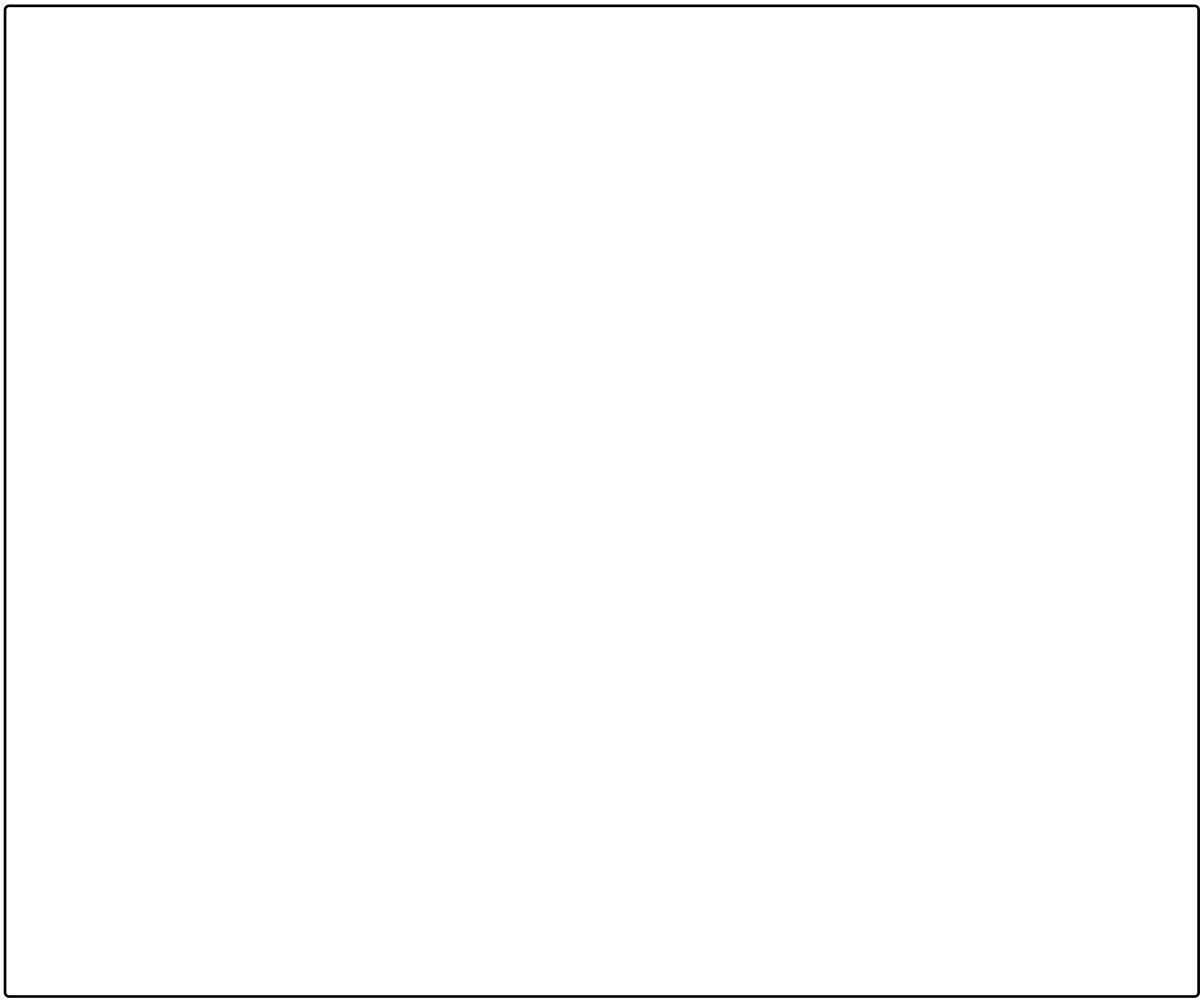
2.2 Relative Entropy and Information Gain

We define **relative entropy** between two discrete distributions $\mathbf{p} \in \{p_1, \cdots, p_n\}$ and $\mathbf{q} = \{q_1, \cdots, q_n\}$ as:

$$D(\mathbf{p}\|\mathbf{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

The above quantity is equal to 0 if and only if $\mathbf{p} = \mathbf{q}$. Otherwise, this quantity is positive. We can think of **relative entropy** as a measure of "distance" between two distributions. This quantity is widely known as the **Kullback-Leibler (KL) divergence** between two distributions.

1. **[3 Points]** Prove that the **KL divergence** is always non-negative. Show all steps of your work
   Hint: You may use this inequality without proof: $x - 1 \geq \log(x)$, where $x \in \mathbb{R}$ with equality if and only if $x = 1$.

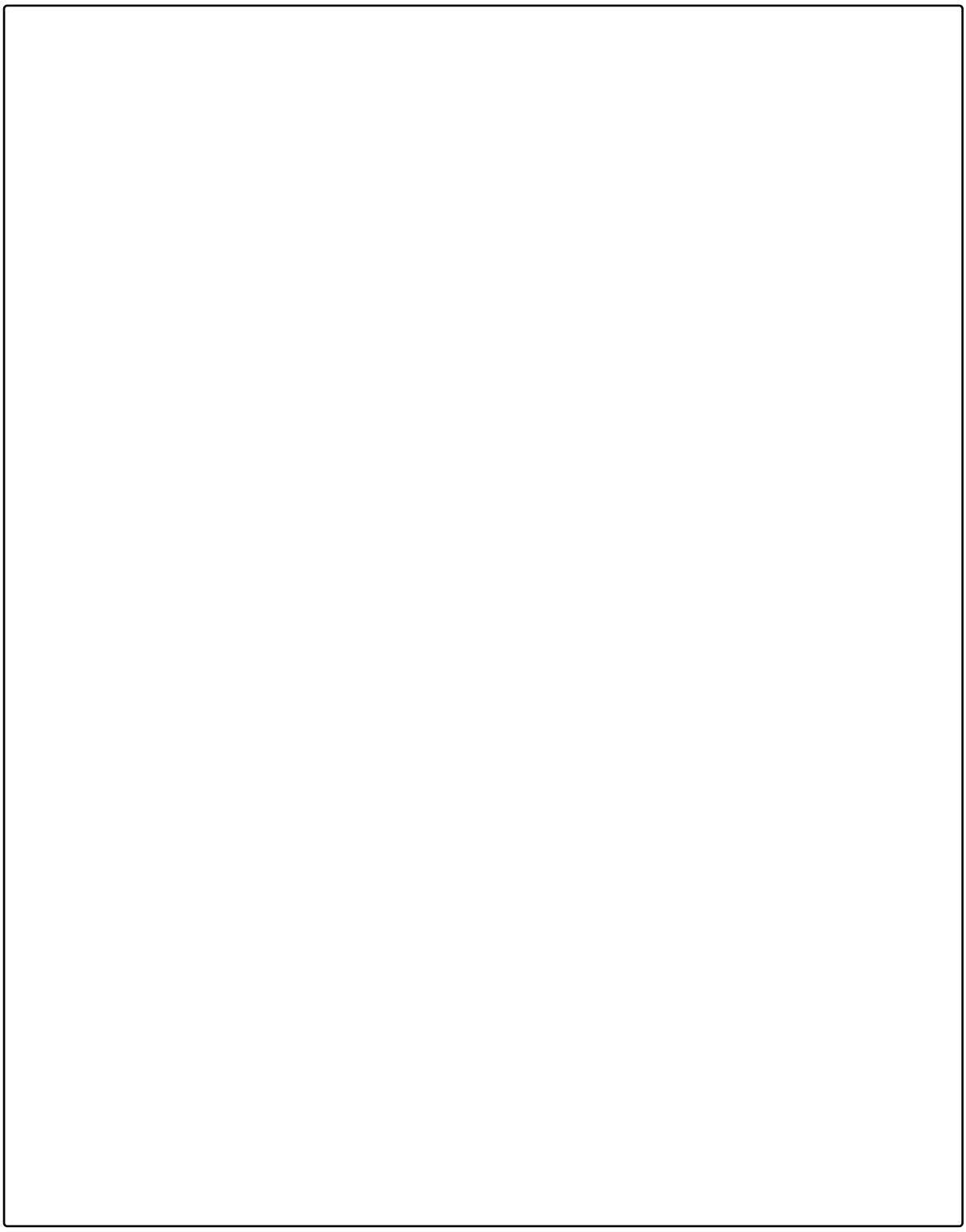


2. **[3 Points]** True or False: **KL-divergence** is symmetric (i.e. $D(\mathbf{p}\|\mathbf{q}) = D(\mathbf{q}\|\mathbf{p})$). If you think the statement is true, please write down a sketch of proof. Otherwise, give a counterexample. Show all steps of your work

3. [**4 Points**] Let X and Y be random variables taking the values $\{1, \cdots, n\}$. Recall the definition of information gain between X and Y:

$$I(X;Y) = H(Y) - H(Y|X)$$

Using the conclusion of part 1), show that the information gain is always non-negative. Show all steps of your work.



3 Regularized Linear Regression Using Lasso [10 Points]

Lasso is a form of regularized linear regression, where the L1 norm of the parameter vector is penalized. It is used in an attempt to get a sparse parameter vector where features of little "importance" are assigned to zero weight. But why does lasso encourage sparse parameters? For this question, you are going to examine this.

Let $\mathbf{X}$ denote an $n \times d$ matrix where rows are training points, $\mathbf{y}$ denotes an $n \times 1$ vector of corresponding output value, $\mathbf{w}$ denotes a $d \times 1$ parameter vector and $\mathbf{w}^\star$ denotes the optimal parameter vector. To make the analysis easier we will consider the special case where the training data is whitened (i.e., $X^\top X = I$). For lasso regression, the optimal parameter vector is given by

$$\mathbf{w}^\star = \underset{\mathbf{w}}{\operatorname{argmin}}\, J_\lambda(\mathbf{w}), \tag{1}$$

where $J_\lambda(\mathbf{w})$ is the function we want to minimize, which is given by

$$J_\lambda(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1, \tag{2}$$

where $\lambda > 0$. Note that the L_1 norm for a vector $\mathbf{w} = [w_1, \ldots, w_d]^\top \in \mathbb{R}^d$ is defined as $\|\mathbf{w}\|_1 = |w_1| + \ldots + |w_d|$.

1. **[2 Points]** In 3.2 and 3.3, we will show that whitening the training data nicely decouples the features, making $\mathbf{w}_i^\star$ determined by the ith feature and the output regardless of other features. To show this, begin by writing $J_\lambda(\mathbf{w})$ in the form

$$J_\lambda(\mathbf{w}) = g(\mathbf{y}) + \sum_{i=1}^{d} f\left(X_{\cdot i}, \mathbf{y}, w_i, \lambda\right), \tag{3}$$

where $X_{\cdot i}$ is the ith column of X, g is a function of only $\mathbf{y}$ and f is a function of $X_{\cdot i}, \mathbf{y}, w_i, \lambda$

2. **[2 Points]** Assume that $w_i^\star > 0$, what is the value of $w_i^\star$ in this case?

3. **[2 Points]** Assume that $w_i^\star < 0$, what is the value of $w_i^\star$ in this case?

4. **[2 Points]** From 2 and 3, what is the condition for $w_i^\star$ to be 0? How can you interpret that condition?

5. **[2 Points]** Now consider ridge regression where the regularization term is replaced by $\frac{1}{2}\lambda\|\mathbf{w}\|_2^2$. What is the condition for $w_i^* = 0$? How does it differ from the condition you obtained in 4?

4 Logistic Regression; Improving our understanding of Convexity [21 points]

Consider a binary classification problem where the goal is to predict a class $y \in \{0, 1\}$, given an input $x \in \mathcal{R}^p$. A method that you can use for this task is *Logistic Regression*. Recall that in *Logistic Regression*, the conditional log likelihood probability can be written as follows:

$$\mathcal{L}(w) = \log P(y|\mathbf{X}, w) = \sum_{i=1}^{n} [y_i w^T x_i - \log(1 + \exp(w^T x_i))]$$

where:

- $\mathbf{X} \in \mathcal{R}^{n \times (1+p)}$ is a data matrix, with the first column composed of all ones
- $w \in \mathcal{R}^{(p+1) \times 1}$ is the weight vector, with the first index w_1 acting as the bias term
- x_i is a column vector of the i^{th} row of $\mathbf{X}$
- $y \in \mathcal{R}^{n \times 1}$ is a column vector of labels $y_i \in \{0, 1\}$
- p is the dimension of data (number of features in each observation)

Our goal is to find the weight vector w that maximizes this likelihood. Unfortunately, for this model, we cannot derive a closed-form solution with MLE. An alternative way to solve for w is to use gradient ascent, and update w step by step towards the optimal w. But we know gradient ascent will converge to the optimal solution w that maximizes the conditional log likelihood $\mathcal{L}$ when $\mathcal{L}$ is concave. In this question, you will prove that $\mathcal{L}$ is indeed a concave function.

1. **[3 points]** A real-valued function $f : S \to \mathcal{R}$ defined on a convex set S, is said to be *convex* if,

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2), \forall x_1, x_2 \in S, \forall t \in [0,1].$$

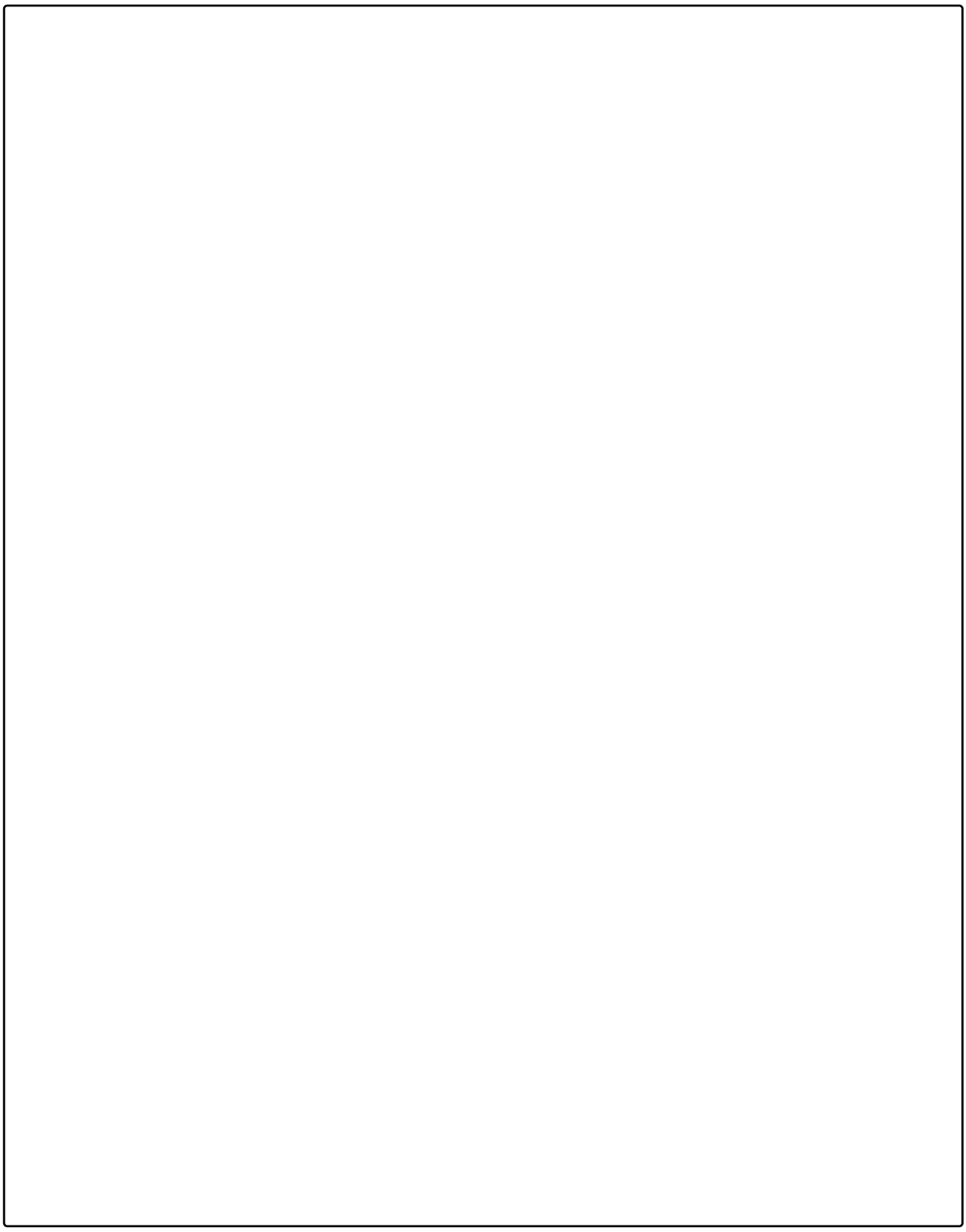
Show that a linear combination of n convex functions, $f_1, f_2, ..., f_n$, $\sum_{i=1}^{n} a_i f_i(x)$ is also a convex function $\forall a_i \in R^+$.

2. [**2 points**] Show that a linear combination of n concave functions, $f_1, f_2, ..., f_n$, $\sum_{i=1}^{n} a_i f_i(x)$ is also a concave function $\forall a_i \in R^+$. Recall that if a function $f(x)$ is convex, then $-f(x)$ is concave. (You can use the result from part (1))

3. [**4 points**] Another property of twice differentiable convex functions is that the second derivative is non-negative. Using this property, show that $f(x) = \log(1 + \exp x)$ is a convex function. Note that this property is both sufficient and necessary. i.e. (if $f''(x)$ exists, then $f''(x) \geq 0 \iff f$ is convex)

4. [**4 points**] Let $f_i : \mathcal{S} \rightarrow \mathcal{R}$ for $i = 1, \ldots, n$ be a set of convex functions. Is $f(x) = max_i f_i(x)$ also convex? If yes, prove it. If not, provide a counterexample.

5. [**8 points**] Show that the log likelihood of *Logistic Regression* is a concave function. You may use the fact that if f and g are both convex, twice differentiable and g is non-decreasing, then $g \circ f$ is convex.



5 Programming Linear Regression [24 Points]

Note: Your code for all of the programming exercises including this one should be submitted to the corresponding programming submission slot on Gradescope. Feel free to use any programming language, as long as your TAs can read your code. Turn in your code in a single .tar ball that might contain multiple source code files along with a README which contains instructions on how to run your code and generate the results and plots. Visualizations and written answers should still be submitted as a part of the rest of the homework in this PDF. In your code, **please use comments to point out primary functions that compute the answers to each question.**

In this problem you will implement linear regression with different regularization techniques and use stochastic gradient descent to learn the model parameters.

5.1 Preliminaries

Recall from class that Linear Regression assumes a linear relationship between the covariates $x_1, \ldots, x_m$ and the continuous response y. Mathematically, this can be written as:

$$y = \hat{y} + \epsilon = \sum_{j=1}^{m} w_j x_j + b + \epsilon$$

Here $\hat{y}$ represents the model prediction for the true y, $\{w_i\}_{i=1}^{m}$ and b are the model parameters to be estimated and ϵ is a zero-mean random error term. We estimate the parameters by minimizing the following loss function:

$$\mathcal{L}(w_1, \ldots, w_m, b) = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}^{(i)} - y^{(i)})^2$$

Here, the superscript $^{(i)}$ denotes a particular datapoint in our dataset and $\hat{y}^{(i)} = \sum_{j=1}^{m} w_j x_j^{(i)} + b$. Note that here we are assuming that we have n points in our dataset with m covariates. The optimal model parameters can then be written as the solution to this equation:

$$w_1^\star, \ldots, w_m^\star, b^\star = \underset{w_1,\ldots,w_m,b}{\operatorname{argmin}} \mathcal{L}(w_1, \ldots, w_m, b)$$

There are a number of ways to minimize the above loss function and find the optimal parameters. In this assignment, you will be implementing **stochastic gradient descent**.

5.2 Data preprocessing

Data preprocessing is a very important step in all ML algorithms including linear regression. You will be using the Carseats dataset, the details of which can be found at

https://rdrr.io/cran/ISLR/man/Carseats.html

You can find the train and test data in `carseats_train.csv` and `carseats_test.csv` files in the handout respectively. This dataset contains **10** covariates (`CompPrice, Income, Advertising, Population, Price, ShelveLoc, Age, Education, Urban, US`) and the response variable that is to be predicted is `Sales`. There are three important steps that you must do.

1. **Binary variable encoding**: In this dataset, `Urban, US` are both binary variables which take values `No` and `Yes`. You must convert them to 0/1 binary variables so that these numerical values can be used for linear regression.
2. **Categorical variable encoding**: `ShelveLoc` is a categorical variable that takes **three** values, namely `Bad, Good, Medium`. You must do **one-hot encoding** for this particular variable. This means that you must create three dummy variables: `ShelveLocBad, ShelveLocGood, ShelveLocMedium`. `ShelveLocBad` should be 1 when the value of `ShelveLoc` is `Bad` and 0 otherwise. `ShelveLocGood, ShelveLocMedium` should be encoded in a similar way. This means that for any datapoint, **exactly** one of `ShelveLocBad, ShelveLocGood, ShelveLocMedium` will take the value 1 and the other two will be 0.
3. **Feature standardization**: Feature standardization makes the data such that it has zero mean and unit variance. For every continuous covariate, you must subtract the mean and divide it by the variance.

$$\hat{x}_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{\sigma_j}$$

where $\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_j^{(i)}$ and $\sigma_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_j^{(i)} - \mu_j)^2}$.
NOTE: You must do the feature standardization for the **test** set using the mean and variance calculated from the **train** set.

5.3 Stochastic Gradient Descent

You will use stochastic gradient descent (SGD) to train your linear regression model. Stochastic gradient descent is a stochastic approximation to gradient descent in which the gradient of the loss function is replaced by an estimate from a single datapoint. Mathematically

$$\mathcal{L} \approx (\hat{y}^{(i)} - y^{(i)})^2$$

Recall the gradient descent update rule:

$$w_j \leftarrow w_j - \eta \frac{\partial \mathcal{L}}{\partial w_j}$$

After computing $\frac{\partial \mathcal{L}}{\partial w_j}$, this becomes

$$w_j \leftarrow w_j - 2\eta(\hat{y}^{(i)} - y^{(i)})x_j^{(i)}$$

Similarly for b:

$$b \leftarrow b - 2\eta(\hat{y}^{(i)} - y^{(i)})$$

NOTE: Make sure that the parameters ($w_1, ..., w_m$ and b) are updated *simultaneously*. **You might want to vectorize your code to make this easier**.

This is also a good time to learn about some jargon related to SGD. A **step** or a **training step** is one gradient update. An **epoch** is when one full cycle over the training data is completed, i.e., each datapoint has been seen once.

[**1 Point**] Suppose your training data consists of 1000 datapoints and you are using SGD to learn your model parameters. How many steps will the algorithm take in **2 epochs**?

Note on reproducibility: When you use Stochastic Gradient Descent in the wild, you must **randomly shuffle** the training data at the beginning of each epoch and then perform the gradient update rule for each datapoint (this is the reason why SGD is *stochastic*).
In this assignment however, please make sure that you do *NOT* shuffle the data before every epoch and instead cycle through the datapoints in the order in which they're given in the .csv file so that your solution can be compared with the reference solution. This will remove the randomness from your experiments and produce a deterministic answer [2] in each run.

5.4 Regularization

Regularization is an important technique to prevent overfitting and achieve better generalization. In this assignment you will implement L_2 and L_1 regularization (also known as ridge and lasso regression respectively).

5.4.1 Ridge regression

The loss for L_2 regularization can be written as

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2 + \lambda\sum_{j=1}^{m} w_j^2$$

Not that we do not regularize b. With the SGD approximation, this becomes

$$\mathcal{L} \approx (\hat{y}^{(i)} - y^{(i)})^2 + \lambda\sum_{j=1}^{m} w_j^2$$

[2] In practice you would want to set a (pseudo) random number generator seed to make your experiments reproducible. However, that is specific to the programming language and libraries that you use. So it won't be comparable to the reference solution in this case.

[**2 Points**] Write the stochastic gradient descent update rules for w_j and b for ridge regression.

5.4.2 Lasso regression

The loss for L_1 regularization can be written as

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2 + \lambda\sum_{j=1}^{m}|w_j|$$

Note that we do not regularize b. With the SGD approximation, this becomes

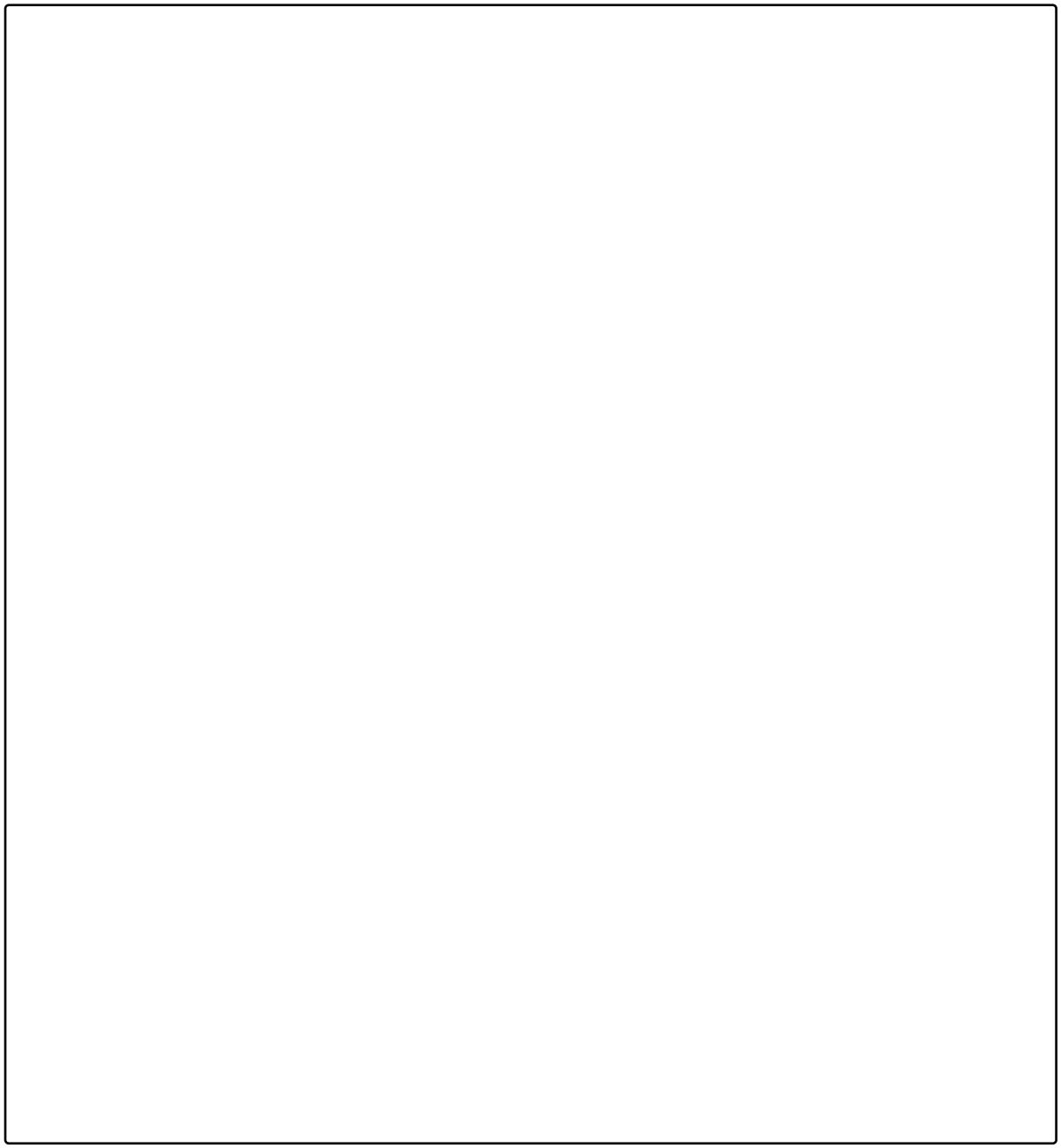
$$\mathcal{L} \approx (\hat{y}^{(i)} - y^{(i)})^2 + \lambda\sum_{j=1}^{m}|w_j|$$

[**2 Points**] Write the stochastic gradient descent update rules for w_j and b for lasso regression.

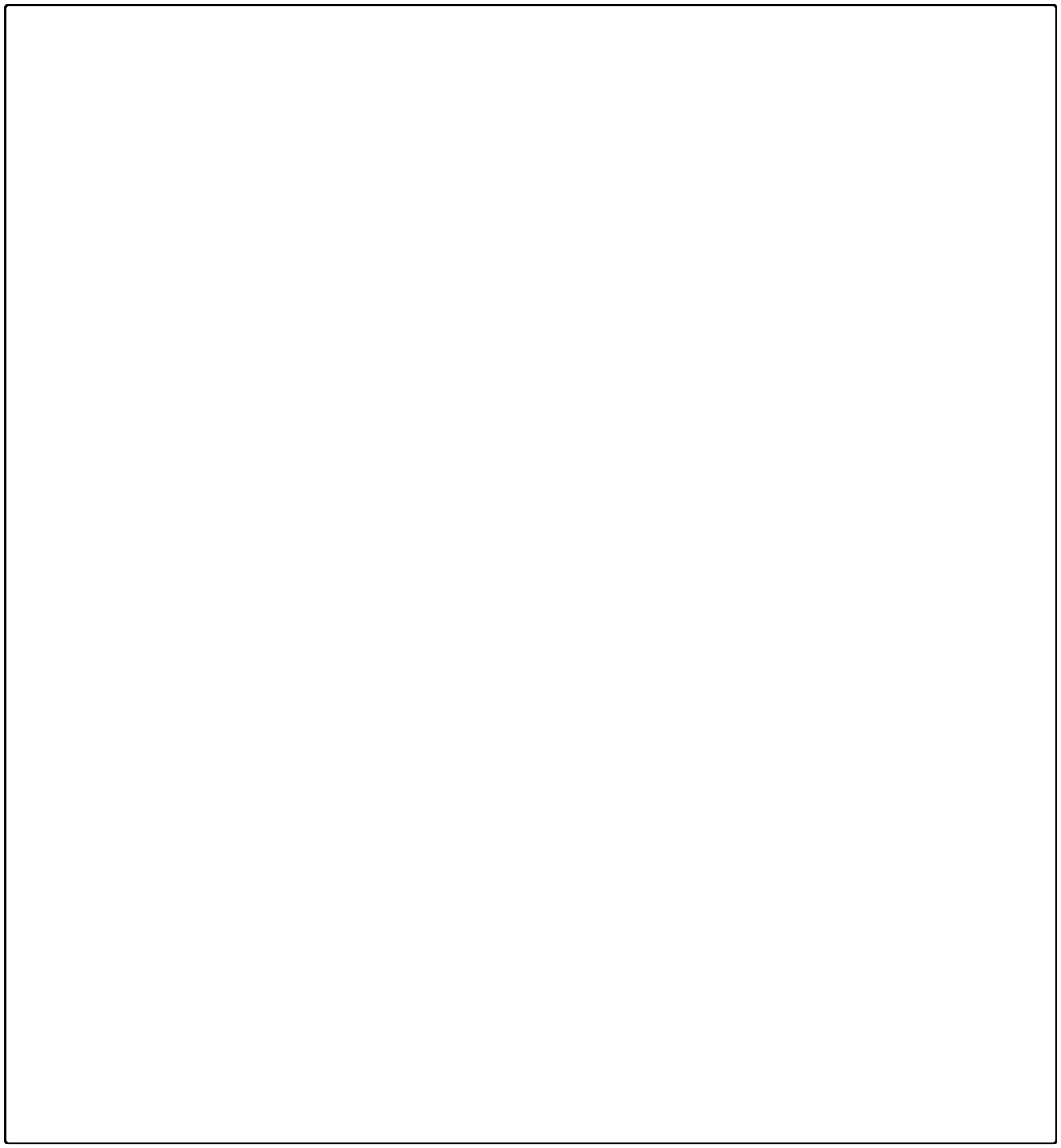
5.5 Training

Now is the time to actually train your linear model. Plot the loss curves for the following training configurations. Please note that you should plot the **training loss** computed at each step. Also note that the loss that you plot should be the SGD approximation to the real training loss computed using the particular datapoint for that step. **You must initialize the model parameters to zeros**. Make sure to clearly label the axes.

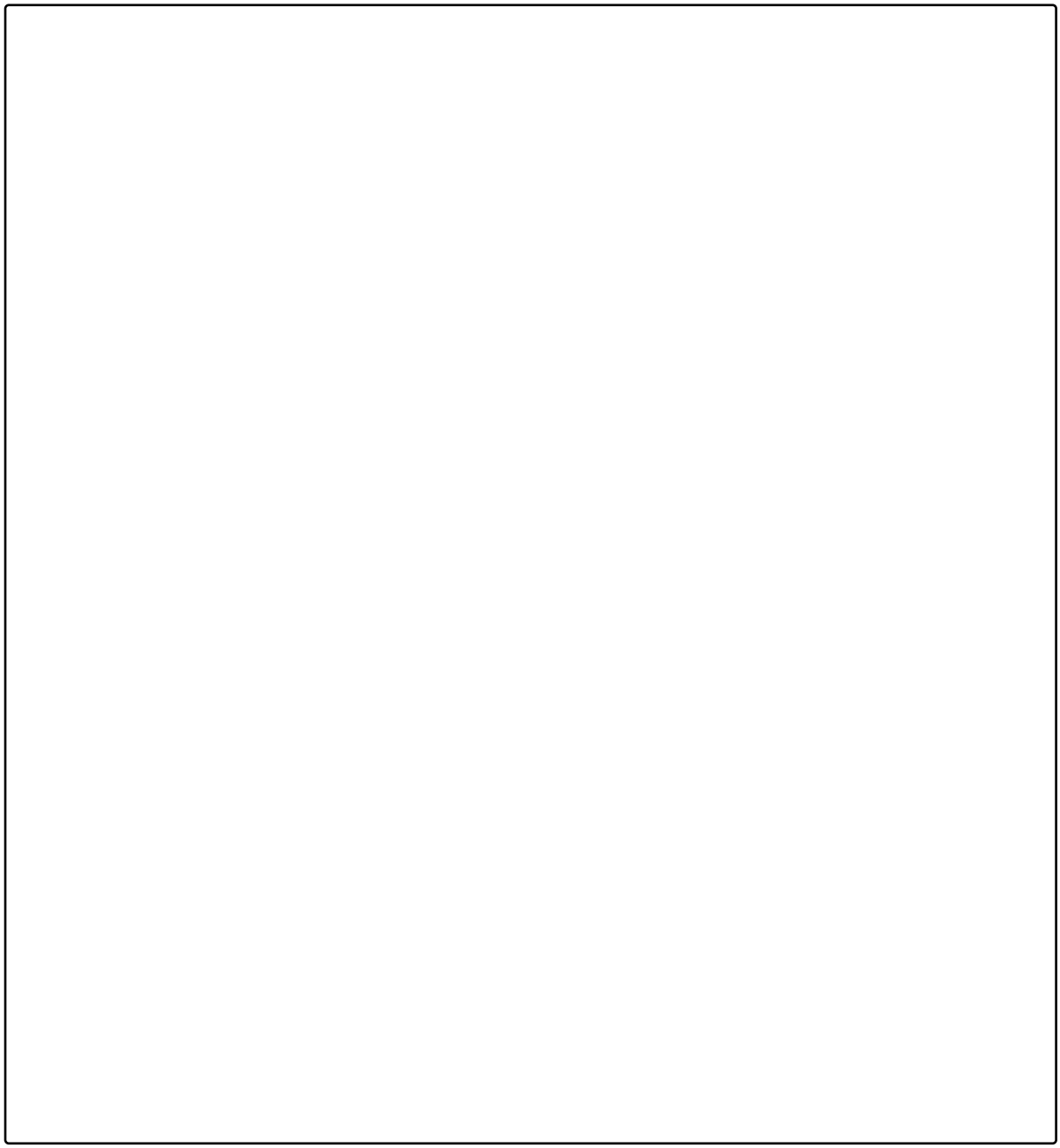
[3 Points] 50 epochs, no regularization, $\eta = 0.01$



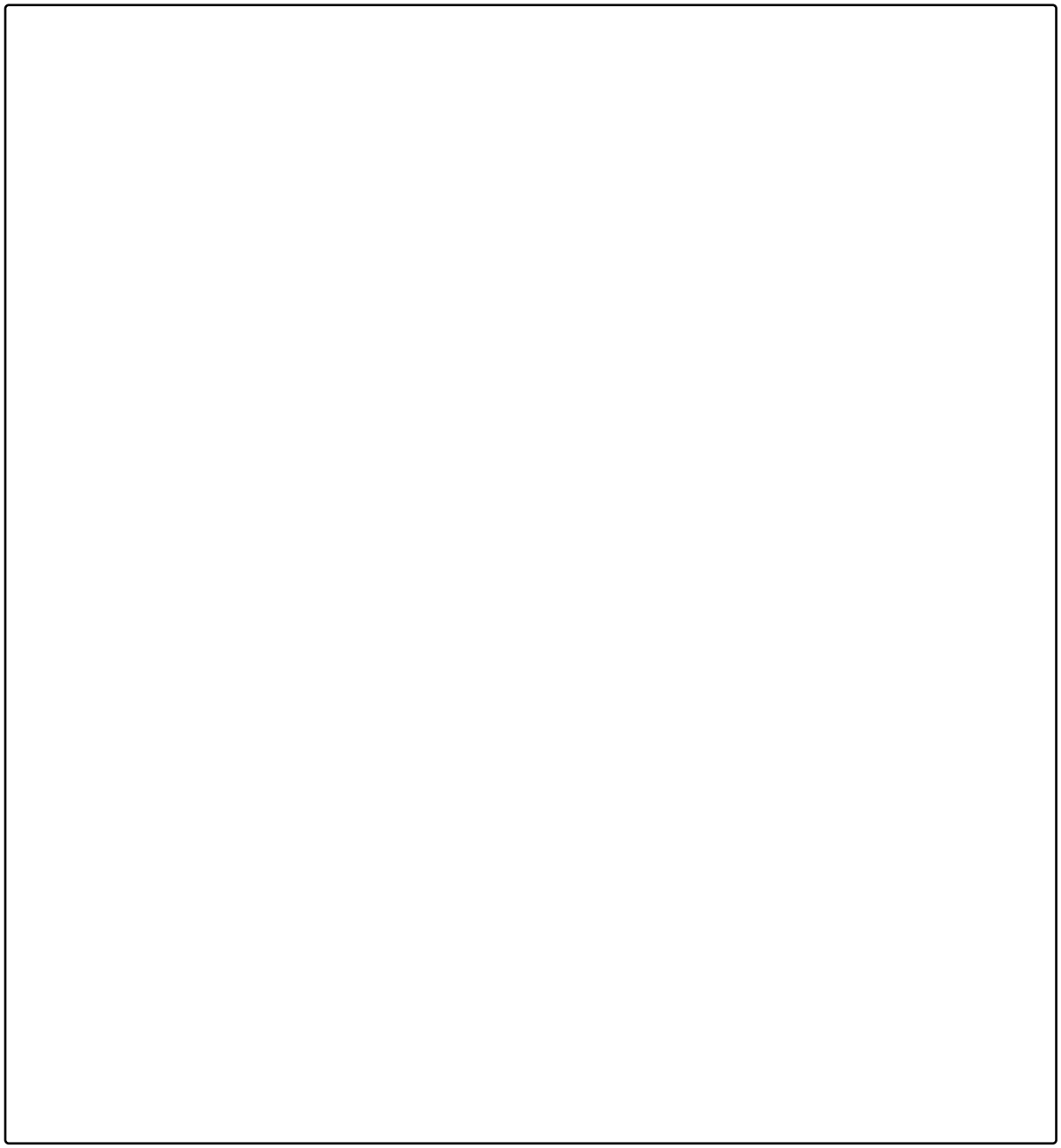
[2 Points] 50 epochs, no regularization, $\eta = 0.001$



[5 Points] 50 epochs, L_2 regularization, $\eta = 0.001$, $\lambda = 0.1$



[5 Points] 50 epochs, L_1 regularization, $\eta = 0.001$, $\lambda = 0.1$



5.6 Evaluation

You will now use your trained linear models to make predictions for datapoints not seen during training and evaluate the performance of your model. In the following questions, you have to compute the test loss which can be written as:

$$\mathcal{L}_{\text{test}} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} (\hat{y}^{(i)} - y^{(i)})^2$$

Note that you are not supposed to add the L_1 or L_2 penalties when you are evaluating your model. For each of the following training configurations, report the test loss after training. Please report the loss values to at least **five** decimal places.

[1 Points] 50 epochs, no regularization, $\eta = 0.01$

[1 Points] 50 epochs, no regularization, $\eta = 0.001$

[1 Points] 50 epochs, L_2 **regularization,** $\eta = 0.001$**,** $\lambda = 0.1$

[1 Points] 50 epochs, L_1 **regularization,** $\eta = 0.001$**,** $\lambda = 0.1$

5.7 Code Submission Checklist

1. Did you add all your source code files in a single .tar ball?

   ◯ Yes

   ◯ No

2. Did you include a README in the .tar ball which contains instructions on how to run your code and generate the different plots and evaluate the trained models?

   ◯ Yes

   ◯ No

3. Did you add comments to point out primary functions that compute the answers to each question?

   ◯ Yes

   ◯ No

6 Collaboration Questions

1. (a) Did you receive any help whatsoever from anyone in solving this assignment?

   (b) If you answered 'yes', give full details (e.g. "Jane Doe explained to me what is asked in Question 3.4")

2. (a) Did you give any help whatsoever to anyone in solving this assignment?

   (b) If you answered 'yes', give full details (e.g. "I pointed Joe Smith to section 2.3 since he didn't know how to proceed with Question 2")

3. (a) Did you find or come across code that implements any part of this assignment?

   (b) If you answered 'yes', give full details (book & page, URL & location within the page, etc.).