

HOMEWORK 4

SVM, KERNELS, DUALITY, REAL-WORLD PRACTICE

CMU 10-701: MACHINE LEARNING (SPRING 2021)

piazza.com/cmu/spring2021/10701

OUT: April 2, 2021

DUE: Wednesday, April 14, 2021 11:59pm

TAs: Jeffrey Huang, Stefani Karp, Yuchen Shen

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/~aarti/Class/10701_Spring21/index.html.
- **Extension Policy:** See the homework extension policy here:
https://www.cs.cmu.edu/~aarti/Class/10701_Spring21/index.html.
- **Submitting your work:**
 - All portions of the assignments should be submitted to Gradescope (<https://gradescope.com/>).
 - **Programming:** We will autograde your Python code in Gradescope. After uploading your code, our grading scripts will autograde your assignment by running your program on a virtual machine (VM). We recommend debugging your implementation on your local machine (or the linux servers) and making sure your code is running correctly before any submission. **Our autograder requires that you write your code using Python 3.6.9 and Numpy 1.17.0.**
 - **Written questions:** **You must type the answers in the provided .tex file. Hand-written solutions will not be accepted. Make sure to answer each question in the provided box. DO NOT change the size of the boxes, because it may mess up the autograder.** Upon submission, make sure to label each question using the template provided by Gradescope. Please make sure to assign ALL pages corresponding to each question.

1 SVM [36 pts]

1. [6 pts] We are given two sets of data points $A = \{(0,0), (1,1), (2,2)\}$ and $B = \{(0,1), (1,2), P\}$, where $P = (u,v)$ is an unknown point on the 2D Cartesian coordinate plane. Assume these points can be perfectly classified, such that the points in A are labeled $+1$ and the points in B are labeled -1 , using a decision boundary of the form $y = wx + b$ for $w, b \in \mathbb{R}$. (That is, the boundary is the set of (x,y) points that satisfy this equation.) If the best possible decision boundary for this split is **not** $y = x + \frac{1}{2}$, what are all of the possible points that P can be? (Here, the “best” possible decision boundary means the maximum margin decision boundary.)

2. [6 pts] Suppose you are given a set of **linearly separable** labeled vectors $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$, $\phi : \mathbb{R}^p \rightarrow \mathbb{R}^d$, $\mathbf{w} \in \mathbb{R}^d$, and $b \in \mathbb{R}$. Let's define the function $h : \mathbb{R}^p \times \mathbb{R} \rightarrow \overline{\mathbb{R}}$ as:

$$h(\mathbf{w}, b) = \sup_{\alpha_i \geq 0 \ \forall i} \sum_{i=1}^N \alpha_i (1 - y_i (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b)).$$

Note that sup is like max, except it doesn't actually need to be *attained*:
for example, $\sup_{x \in \mathbb{R}} [-e^x] = 0$.

Determine the domain and the range of the function h . The range should be the subset of $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ (known as the extended reals) which is attainable by points in the domain. This is often called the image of the function on the given domain.

Hint: We would like to write out the function h in piecewise fashion. As an example, if we define $f : \mathbb{R} \rightarrow \overline{\mathbb{R}}$ as:

$$f(x) = \sup_{\gamma \geq 0} [x^2 + \gamma(2 - x)]$$

then we see that $f(x) = x^2$ for $x \geq 2$ and $+\infty$ otherwise. Therefore, the domain is \mathbb{R} and the range is $[4, +\infty]$.

3. [24 pts] Suppose you are given a set of labeled vectors $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$, $\phi : \mathbb{R}^p \rightarrow \mathbb{R}^d$, $\mathbf{w} \in \mathbb{R}^d$, and $b \in \mathbb{R}$. If the data is not linearly separable and we want to use a SVM to classify the labels, we can let some points fall within less than 1 from or on the wrong side of the margin. For each example with a margin of $1 - \xi_i$, we penalize our objective by $C\xi_i$ for some constant C . Through C , we control the relative weights of the two goals, which are minimizing $\|\mathbf{w}\|_2^2$ and maximizing the number of points with at least one 1 margin. The soft margin (referring to the fact that the data is not separable) primal and dual problems are given as follows.

Primal

$$\begin{aligned} \min_{\mathbf{w}, b, \{\xi_i\}_{i=1}^N} & \frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i \\ \text{s.t. } & y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i \quad \forall i; \quad \xi_i \geq 0 \quad \forall i \end{aligned}$$

Dual

$$\begin{aligned} \max_{\{\alpha_i\}_{i=1}^N} & -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle + \sum_{i=1}^N \alpha_i \\ \text{s.t. } & \sum_{i=1}^N \alpha_i y_i = 0; \quad 0 \leq \alpha_i \leq C \quad \forall i \end{aligned}$$

The objective of this question is to start from the Primal problem and derive the Dual problem.

The first step to deriving the dual problem is computing the Lagrangian function of the Primal problem based on the constraints we set in the Primal problem.

- a) [1.5 pts] Considering only the first constraint, naming the corresponding Lagrange multiplier α , what constraint should be imposed on α ?

- ☐ $\alpha_i \in \mathbb{R} \quad \forall i$
- ☐ $\alpha_i \geq 0 \quad \forall i$
- ☐ $\alpha_i > 0 \quad \forall i$
- ☐ $0 \leq \alpha_i \leq C \quad \forall i$
- ☐ $0 < \alpha_i < C \quad \forall i$

- b) [3 pts] What would be the desired Lagrangian function $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha})$?

- ☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i + \sum_{i=1}^N (\alpha_i y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - 1 + \xi_i)$
- ☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N (\alpha_i y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - 1 + \xi_i)$

- ☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N (\alpha_i y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - \alpha_i + \alpha_i \xi_i)$
☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i + \sum_{i=1}^N (\alpha_i y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - \alpha_i + \alpha_i \xi_i)$
☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i + \sum_{i=1}^N (y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - \alpha_i + \alpha_i \xi_i)$
☐ $\frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N (y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - \alpha_i + \alpha_i \xi_i)$

c) **[1.5 pts]** We now include the second constraint into the expression from part (a). Because the name α_i is already used, we set the Lagrange multiplier for this constraint to be β_i . What constraint should be imposed on β ?

d) **[3 pts]** What would be the desired Lagrangian function $\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \beta)$?

We now want to determine the optimal primal values for \mathbf{w} , b , and $\boldsymbol{\xi}$.

e) [**3 pts**] Compute $\frac{\partial \mathcal{L}}{\partial \mathbf{w}}$. What relation do we get upon setting this to 0?

f) [**3 pts**] Compute $\frac{\partial \mathcal{L}}{\partial b}$. What relation do we get upon setting this to 0?

g) [**3 pts**] Compute $\frac{\partial \mathcal{L}}{\partial \xi_i}$ for each i . What relations do we get upon setting each of these to 0?

h) [3 pts] With the expressions derived above, we can plug these in and perform some manipulations to arrive at the dual problem displayed earlier on. Notice that the dual problem managed to get rid of the ξ_i variables completely. **Explain why this was possible.**

i) [3 pts] For each i , why do we end up with $\alpha_i \in [0, C]$? (Hint: Think about β_i .)

2 Kernels [15 points]

Figure 1 (on the next page) shows SVM decision boundaries resulting from different combinations of kernels, slack penalties, and RBF bandwidths. Match each plot from Figure 1 with one of the SVM settings.

You need not provide any justification to receive full credit; only the correct matching.

Linear kernel: $K(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}'$. RBF kernel: $K(\mathbf{x}, \mathbf{x}') = \exp(-\kappa \|\mathbf{x} - \mathbf{x}'\|_2^2)$

1. Linear kernel, $C = 0.1$

2. Linear kernel, $C = 100$

3. RBF kernel, $C = 1$, $\kappa = 10$

4. RBF kernel, $C = 1$, $\kappa = 50$

5. RBF kernel, $C = 0.1$, $\kappa = 0.1$

6. RBF kernel, $C = 20$, $\kappa = 0.1$

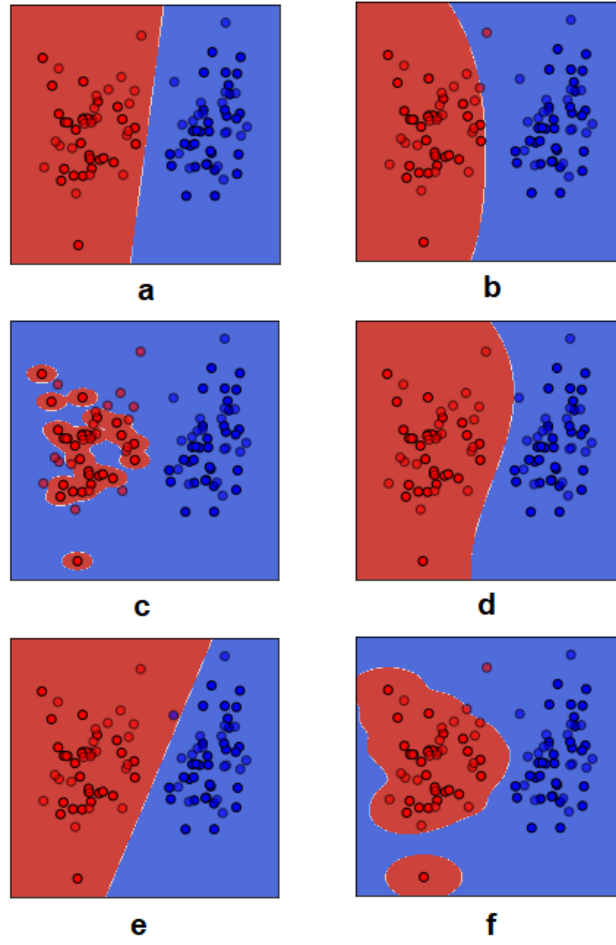


Figure 1: SVM decision boundaries learned with different parameter settings.

The purpose of this question is to explain the choices made in the previous question.

INSTRUCTIONS

In this question you will fill in the blanks in the passage below to make a valid explanation of your answer to the previous question. For each blank, you must choose a word/phrase from the bank below. Please copy the word/phrase exactly as written in the bank. Each word/phrase may be used zero times, once, or more than once.

Additional instructions for certain blanks follow:

The answer marked S1 and the answers marked [Answer to N] will correspond to a (possibly empty) set of letters for the figures on the previous page. The values of [Answer to N] should come from the previous question, so they will not be graded again in this question.

The answers to all blanks labeled with a letter U (as in U2, U3, etc.) will come from the bank below.

Replace each blank with an answer, but keep the boxes around the answers. Please be sure that the positions of the boxes don't change. (They shouldn't if you don't alter the box widths.)

EXPLANATION

We see that choices correspond to linear kernels because the decision boundaries corresponding to these figures are .

To see which linear kernel is which, note that the constant C will adjust the objective to make it . As $C \rightarrow 0$, it changes the objective to make it . As $C \rightarrow \infty$, it changes the objective to make it . Hence, the linear kernel with the lower value $C = 0.1$ matches Figure , while the one with the higher value $C = 100$ matches Figure .

We now move on to the RBF kernels.

The value κ represents a parameter that is the radius of influence of support vectors. As $\kappa \rightarrow 0$, it will make the decision boundary . As $\kappa \rightarrow \infty$, it will make the decision boundary .

Hence, the RBF kernel with $\kappa = 50$ corresponds to , while the RBF kernel with $\kappa = 10$ corresponds to .

Finally, because of our observations about C from above, the RBF kernel corresponding to $\kappa = 0.1$ and $C = 0.1$ corresponds to , while the RBF kernel corresponding to $\kappa = 0.1$ and $C = 20$ corresponds to .

WORD/PHRASE BANK

linear

nonlinear

quadratic

nonquadratic

biased

unbiased

more or less tolerant to errors

classify perfectly if possible

ignore the data

a function of the support vectors

directly proportional to

inversely proportional to

generalize well

3 Duality [18 pts]

1. [2 pts] **Select one:** The dual function $f^*(y)$ of $f(x) = x^2$ is:

☐ $-y^2/2$

☐ $y^2/2$

☐ $y^2/4$

☐ $y/2$

☐ $y/4$

2. [5 pts] Compute the dual $f^*(y)$ of $f(x) = \max\{3x - 2, -5x - 3\}$. For each of the 5 following questions, select exactly one of the 5 answers.

- (a) [1 pt] What is $f^*(0)$?

☐ 2

☐ 2.375

☐ -3

☐ -2.5

☐ 2.625

- (b) [1 pt] What is $f^*(-5)$?

☐ 3

☐ 2

☐ -3

☐ -2

☐ -1

- (c) [1 pt] What is $f^*(3)$?

☐ 3

☐ 2

☐ -3

☐ -2

☐ -1

- (d) [1 pt] What is $f^*(2)$?

- ☐ 2
- ☐ 2.125
- ☐ -3
- ☐ -2.5
- ☐ 2.875

(e) [1 pt] What is $f^*(4)$?

- ☐ 3
- ☐ 2
- ☐ 1
- ☐ $+\infty$
- ☐ $-\infty$

3. [2.5 pts] **Select all that apply:** The dual function $f^*(y)$ of $f(x)$ is...

- ☐ $\sup_x (y \cdot x - f(x))$
- ☐ $\inf_x (f(x) - y \cdot x)$
- ☐ $-\inf_x (f(x) - y \cdot x)$
- ☐ the minimal amount by which we'd have to shift $g(x) = y \cdot x$ upward so that the shifted $g(x) \geq f(x)$ for all x in the domain (assume $0 \leq f^*(y) < \infty$ in this setting, if needed).
- ☐ the minimal amount by which we'd have to shift $g(x) = y \cdot x$ downward so that the shifted $g(x) \leq f(x)$ for all x in the domain (assume $0 \leq f^*(y) < \infty$ in this setting, if needed).

4. [2 pts] **True or False:** For all functions $f(x)$, we have $f(x) = f^{**}(x)$, where f^{**} denotes the double dual.

- ☐ True
- ☐ False

5. [2.5 pts] **Select all that apply:** Based on the definition of the dual cone $K^* = \{y \mid y \cdot x \leq 0 \ \forall x \in K\}$, we know it must be a convex cone because...

- ☐ $y_1 \cdot x \leq 0, y_2 \cdot x \leq 0 \ \forall x \in K \implies (\alpha y_1 + (1 - \alpha)y_2) \cdot x \leq 0 \ \forall x \in K, \ \forall \alpha \in \mathbb{R}.$
- ☐ $y_1 \cdot x \leq 0, y_2 \cdot x \leq 0 \ \forall x \in K \implies (\alpha y_1 + (1 - \alpha)y_2) \cdot x \leq 0 \ \forall x \in K, \ \forall \alpha \in [0, 1].$
- ☐ $y \cdot x \leq 0 \ \forall x \in K \implies ty \cdot x \leq 0 \ \forall x \in K, \ \forall t \in \mathbb{R}.$
- ☐ $y \cdot x \leq 0 \ \forall x \in K \implies ty \cdot x \leq 0 \ \forall x \in K, \ \forall t \geq 0.$

☐ it is **not** necessarily a convex cone

6. [2 pts] **Select one:** The above analysis relies on...

- ☐ only K being convex
- ☐ only K being a cone
- ☐ K being a convex cone
- ☐ none of the above

7. [2 pts] **Select one:** Suppose we rewrite the *same* dual cone definition above as $K^* = \{y \mid f_y(x) \leq 0 \ \forall x \in K \subseteq \mathcal{X}\}$. The set $\{f_y(x) \mid y \in K^*\}$ is a subset of what set of functions on \mathcal{X} ? Choose the *smallest* set that's a valid subset.

- ☐ convex functions on \mathcal{X}
- ☐ affine functions on \mathcal{X}
- ☐ linear functions on \mathcal{X}
- ☐ negative functions on \mathcal{X}
- ☐ positive functions on \mathcal{X}

4 Neural Network Programming [31 pts]

Your goal in this assignment is to label inputs by implementing a **Neural Network model** based on the skeleton code provided. We have set up the autograder to try to mimic a “production” machine learning task. In particular we have constructed a separate training set, validation set, and test set, and **there may be distribution shift** so that the distributions are not quite identical in the three sets. In addition, you will not have any access to the test set while developing your model: just like a real model, you must deploy with the knowledge that your model may encounter a different distribution than it was trained on.

This is a new type of assignment for us. We hope that it will be fun and interesting, but there may also be problems with the design. For that reason, (a) we will try to grade this implementation problem leniently, and (b) if something doesn’t appear to be working as expected, please let us know.

Please carefully read all comments in the provided skeleton code as they contain very important instructions.

The training dataset is provided to you, containing 3600 inputs and their respective labels. Each input has 40 features. The labels belong to 10 unique classes (labeled 0 to 9). You have the freedom to choose the neural network model architecture and any associated hyperparameters, as long as you follow the instructions in the skeleton code file.

This task is very open ended! It would be possible to spend arbitrarily much time optimizing the model structure and hyperparameters. Please do not do this: it will likely make your grade worse! In particular, excessive use of the validation set will likely result in overfitting.

You should train your model using the **Training** dataset, which can be loaded by following the instructions in the comment of the Dataloader class we provided. You can submit your trained model any number of times to Gradescope, which will automatically run your model against a **VALIDATION** dataset, comprised of 400 inputs, and tell you your validation cross-entropy loss and validation accuracy. Note that there may be some distribution shift between the training and validation datasets — this is part of the problem design, so be careful!

IMPORTANT: The “score” that you see on Gradescope will not be your homework score!!! It is the validation accuracy, which is *not* your final score for this problem! After the assignment is due, we will run your model against a TEST dataset and assign a grade based based on your TEST accuracy. So do not try to get a perfect score on the autograder! This would likely result in a low grade due to overfitting. There may again be some distribution shift when moving to the test set — again, this is part of the problem design!

While the training, validation, and test distributions may not be identical, they are still strongly related: in general, good models on the training distribution will also be good models on the validation or test distribution. But remember that a high training accuracy does *not* mean a good model on the training distribution, due to possible overfitting.

You need to write your program using **Python3** and **Pytorch**. Other packages you are

allowed to use include **scikit-learn**, **matplotlib** and **NumPy**. Gradescope will grade submissions using Python 3.6 and PyTorch 1.8.1. You can feel free to use other toolkits as well for training (e.g., TensorFlow), so long as you can translate the final model to a format that works in PyTorch for submission.

You should submit 1) your filled-in `code.py` file and 2) your `mymodel.pth` file with your saved model parameters. See detailed instructions in the skeleton code on how to submit your saved model parameters. **Please do not put your files into a folder and zip it.** You can either zip them together directly or submit them to the auto-grader without producing a zip first.

Instructions on how to install PyTorch: <https://pytorch.org/>

Tips:

- You do not need to write a very complicated model for this problem.
- You should be able to train your model using your CPUs — no GPUs should be needed.
- As a rough benchmark, we averaged together the performance of a couple of different models that the TAs quickly put together. The average validation accuracy was 52%. We checked to make sure that these models did not overfit very much. So if you can get at least somewhere around this validation accuracy, and if you don't overfit too much while doing so, you will have OK (but not excellent) test set performance.

Collaboration Questions Please answer the following:

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

Yes / No.

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

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

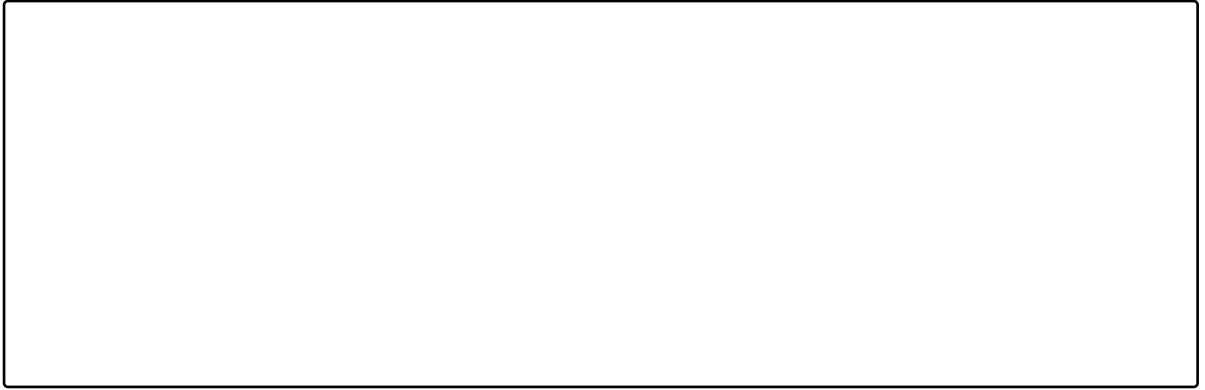
Yes / No.

- 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. Did you find or come across code that implements any part of this assignment ?

Yes / No. (See below policy on "found code")

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

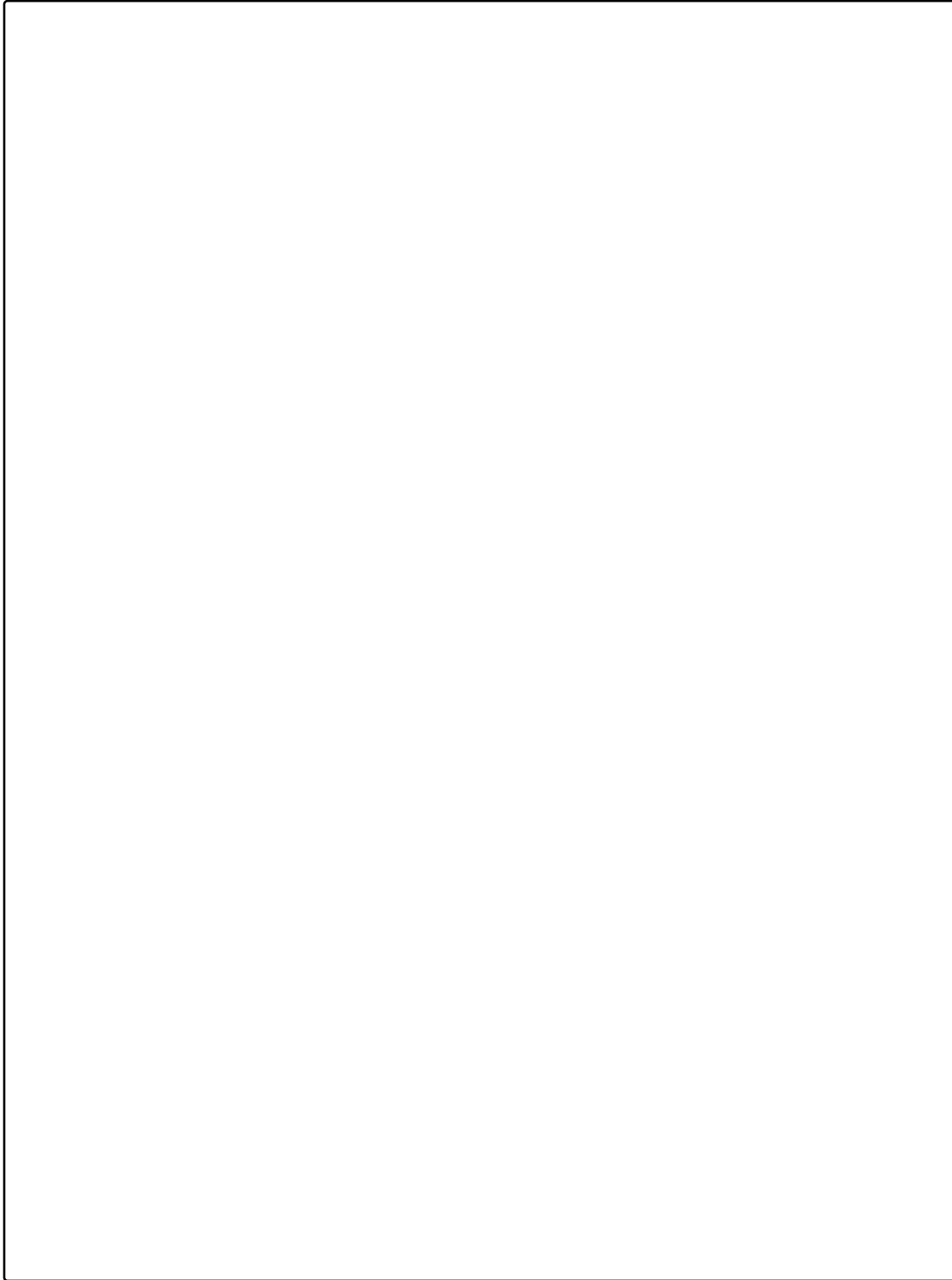


Overflow boxes

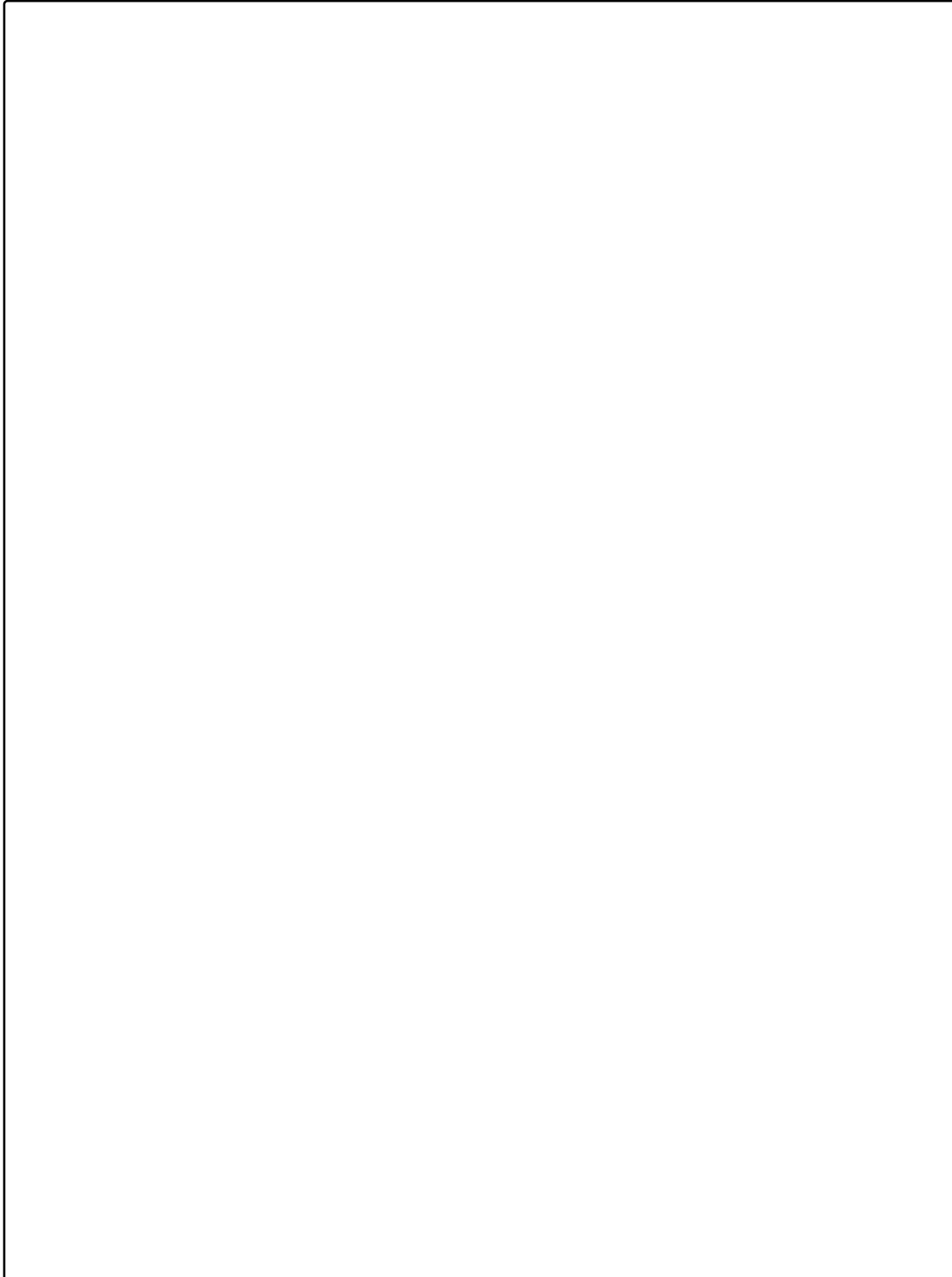
Section 1



Section 2



Section 3



Section 4

