ORF 363/COS 323 PROBLEM SET 4 Fall 2023, Princeton University
Instructor: A.A. Ahmadi
TAs: Beneventano, Chaudhry, Hua, Li, Lok
Due on Thursday, October 26, 2023, at 1:30pm EST, on Gradescope

For all problems that involve coding, please include your code. Our instructions are given with MATLAB in mind (recommended language), but you are free to use a different language.

Problem 1: Support Vector Machines (SVMs)

Recall our Support Vector Machines application of convex optimization from lecture. We have m feature vectors $x_1, \dots, x_m \in \mathbb{R}^n$ with each x_i having a label $y_i \in \{-1, 1\}$. The goal is to find a linear classifier, that is a hyperplane $a^T x - b$, where $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$, by solving the optimization problem

$$\begin{array}{ll} \min\limits_{a,b} & \|a\| \\ \text{s.t.} & y_i(a^T x_i - b) \geq 1, \ \forall i = 1, \dots, m. \end{array} \tag{1}$$

We will then use this classifier to classify new data points.

1. Uniqueness of the optimal solution.

   (a) Is the objective function $\|a\|$ convex? Strictly convex?

   (b) What about $\|a\|^2$? Is it convex? Strictly convex?

   (c) Prove that the solution to (1) is unique.

2. We would like to show that the optimization problem (1) is equivalent to

$$\begin{array}{ll} \max\limits_{a,b,t} & t \\ \text{s.t.} & y_i(a^T x_i - b) \geq t, \ \forall i = 1, \dots, m \\ & \|a\| \leq 1, \end{array} \tag{2}$$

   which is easier to interpret in terms of finding a classifier with maximum margin. Show that if (1) is feasible (with a positive optimal value), then (2) is feasible (and has a positive optimal value). Conversely, show that if (2) is feasible (with a positive optimal value), then (1) is feasible (and has a positive optimal value). You can assume that there is at least one data point with $y_i = 1$ and one with $y_i = -1$ as otherwise there is nothing to classify.

3. Assume the optimal value of (2) is positive. Show that an optimal solution of (2) always satisfies $\|a\| = 1$.

Problem 2: SVMs with linearly separable data

Open the Matlab file `HWSVM.mat`. To do this, download the file into your working directory and open it by calling `"load HWSVM"` in Matlab. This will load 6 vectors into MATLAB. You will need three of these vectors (`"x1part2"`, `"x2part2"` and `"ypart2"`) for this part of the problem. These three vectors correspond to $m = 53$ points in $\mathbb{R}^2$ whose components $(x_1, x_2)_{i,i=1,\dots,m}$ are given in the first two vectors and whose labels y_i are given in the vector `ypart2`. For Python users, you can use the following code to load the data file.

```
import scipy
mat = scipy.io.loadmat(‘HWSVM.mat’)
x1 = mat[‘x1part2’]
x2 = mat[‘x2part2’]
y = mat[‘ypart2’]
```

1. Plot all the 53 points on a graph. We need to be able to tell the difference between points that are labelled 1 and points that are labelled -1.

2. Solve optimization problem (1) and plot on the same graph the optimal linear classifier (hyperplane) and the two shifted hyperplanes corresponding to the boundaries of the margin. Give the equations of these three lines.

3. Which points are the support vectors? Give their coordinates.

Problem 3: SVMs with data that is not linearly separable

You will now need the data vectors `"x1part3"`, `"x2part3"` and `"ypart3"` from `"HWSVM.mat"`. These three vectors correspond to $m = 100$ points $(x_1, x_2)_{i,i=1,\dots,m}$ in $\mathbb{R}^2$ and an associated vector y which has the label of each point.

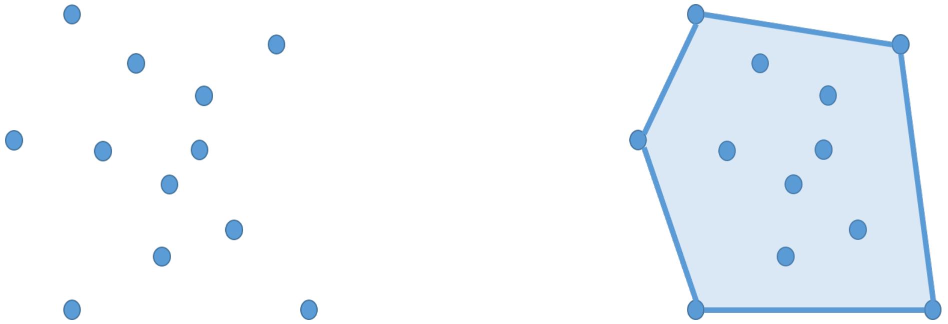
1. Let S be a set consisting of s points $z_1, \dots, z_s$ in $\mathbb{R}^k$. The convex hull of S is defined as

$$\text{conv}(S) = \left\{ \sum_{i=1}^{s} \lambda_i z_i \mid z_i \in S, \lambda_i \geq 0, \text{ and } \sum_{i=1}^{s} \lambda_i = 1 \right\}.$$

In words, this is the set of points that can be written as a convex combination of the points in S. A geometric interpretation of this definition is given in Figure 1.

Define

$$A = \{(x_1, x_2)_{i,i=1,\dots,m} | y_i = 1\}$$



(a) A set of points S (b) Convex hull of S

Figure 1: Convex hull of a set

and

$$B = \{(x_1, x_2)_{i,i=1,\dots,m} | y_i = -1\}.$$

We say that the sets A and B are linearly separable if there exists a hyperplane $a^T x - b$ that takes value ≥ 1 on A and ≤ -1 on B. Prove that if A and B are linearly separable, then their convex hulls do not intersect.

2. For the numerical data given, find a point that is both in conv(A) and conv(B) using CVX. Plot this point on the graph and give its coordinates.
   Hint: Write the problem as a convex optimization problem.

3. Recall the following convex optimization problem from lecture that attempts to simultaneously minimize the number of misclassified points and maximize the length of the margin:

$$\begin{aligned} \min_{a,b,\eta} \quad & \|a\| + \gamma \|\eta\|_1 \\ \text{s.t.} \quad & y_i(a^T x_i - b) \geq 1 - \eta_i, \ \forall i = 1, \dots, m \\ & \eta_i \geq 0, \ \forall i = 1, \dots, m. \end{aligned} \tag{3}$$

   Solve this problem for $\gamma = 1, 2, \dots, 10$ and generate two plots: The first one will give the length of the margin (counting both sides) as a function of γ; the second one will give the number of misclassified points as a function of γ. Discuss the overall trends of the two plots; are they what you were expecting?

Problem 4: Hillary or Bernie?

You would like to use the knowledge you've acquired in optimization over the past few weeks to see if you could have predicted the outcome of each Hillary-Bernie race in the Democratic primaries. To make things easier, you consider only the counties in the tri-state area and New England, i.e., those that belong to the states of New York, New Jersey, Maine, New Hampshire, Pennsylvania, Vermont, Massachussetts, Connecticut, or Rhode Island.

Your goal is to find a linear classifier that, for each county, labels it either as a Bernie win or as a Hillary win. To do this, you have access to a feature vector comprising the following features: mean income, percentage of hispanics, percentage of whites, percentage of residents with a Bachelor's degree or higher, and population density.

1. Load the data file `Hillary_vs_Bernie` in MATLAB/Python. This file includes 4 parts: `features_train`, `features_test`, `labels_train` and `labels_test`.

   In `features_train`, we have given you the feature vectors for 175 counties and in `label_train`, their corresponding labels (-1 is a Bernie win and 1 is a Hillary win). As there was a wide disparity in the orders of magnitude of the original data (average income is around 10^4 whereas the percentages are between 0 and 1), each feature vector has already been normalized by its standard deviation. The original data can be found at `https://www.kaggle.com/benhamner/2016-us-election` (as fact checking is popular at the moment :)). Solve problem (3) to build a linear classifier for this training set for $\gamma = 0.1, 1, 10$. For each value of γ, specify the optimal a^*, b^* obtained.

2. Test the performance of your classifier using the feature vectors from 21 other counties (given in `features_test`) by comparing the labels obtained to the ones given in `label_test`. Which γ gives you the highest success rate in terms of prediction? Take a look at the entries of a^* in this case – what does this suggest about the people who vote for Hillary compared to those who vote for Bernie?

Problem 5: Newton fractals

The sensitivity of Newton's method to initial conditions is beautifully demonstrated using plots over the complex plane known as *Newton fractals*. You may have seen a picture of Newton fractals in lecture notes, and now your task in this problem is to produce the Newton fractal associated with the critical points of $f(z) = z^5 - 5z$. The steps below are only meant to help you do this — there is no grade assigned to them.

1. Note that z is a complex number throughout this exercise. Verify that the critical points of f, i.e., the roots of f' are $z_1 = 1, z_2 = -1, z_3 = i, z_4 = -i$.

2. Discretize $[-1, 1] \times [-1, 1]$ using intervals of length 0.0031. We recommend that you define the sequence of points $x = -1 : 0.0031 : 0.999$, and $y = -0.999 : 0.0031 : 1$ to avoid certain numerical issues. For each point (x_j, y_l) in your discrete grid, apply Newton's method with $z_{jl} = x_j + iy_l$ as its initial point.
   Hint: Consider using the `meshgrid` function to create your grid (which will contain 645^2 points on the complex plane). To run the Newton method, we recommend using matrix operations in MATLAB instead of for-loops. Finally, you may set the maximum number of iterations for Newton's method to 200 for simplicity.

3. Map each of the critical points of f to some color code; e.g., $z_1 \leftrightarrow 1, z_2 \leftrightarrow 2, z_3 \leftrightarrow 3, z_4 \leftrightarrow 4$. Then, to each initial condition (i.e., to each (x_j, y_l) on the grid) assign one of the four color codes based on the root that the iterations are converging to (up to some tolerance error, say, $\epsilon = 0.01$). Depending on how you discretize, for some initial conditions the algorithm may not converge. In that case, assign color code 0 to that particular (x_j, y_l). You will obtain a 645×645 matrix of color codes representing your Newton fractal. Plot it using the `imagesc` function in MATLAB or the `imshow` function in Python (matplotlib).

Submit a print out of your code and plot. To get credit, your code must produce the plot.