EPFL MGT-483— Project
Robust Regression

April 15, 2021

This project is due on **May 14, 2021, at 23:59**. You may form teams of up to three people. Each team should upload a single zip-file containing their report and MATLAB or Python code to Moodle. Make sure to clearly state the team members in your report. You can use the Moodle forum to find additional team members. There are 100 points and 10 bonus points in total.

Notation: For any $N \in \mathbb{N}$, we use $[N]$ to denote $\{1, 2, \ldots, N\}$.

1 Introduction

Linear regression relates a multivariate *input* $x \in \mathbb{R}^d$ to a scalar *output* $y \in \mathbb{R}$ via a linear model, that is, $y \approx \theta^\top x$ for some parameter vector $\theta \in \mathbb{R}^d$. The goal of linear regression is to learn θ from a finite dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$. In practical applications only part of the output can be explained by the inputs, and therefore we have $y^{(i)} = \theta^\top x^{(i)} + \xi^{(i)}$, where the error $\xi^{(i)}$ is a realization of a noise random variable with zero mean. In this case, the best one can hope for is to find an estimator $\widehat{\theta} \approx \theta$ constructed from the data. The estimator $\widehat{\theta}$ can be used to predict the output y corresponding to a new unseen input x, which is at the core of many applications in statistics and machine learning. We assume throughout this project that $x_1^{(i)} = 1$ for all $i \in [N]$, and thus θ_1 can be interpreted as a bias term that represents the mean or the median of $y^{(i)}$.

In this project you will derive and implement several linear programming-based methods to learn the unknown parameter vector θ from data.

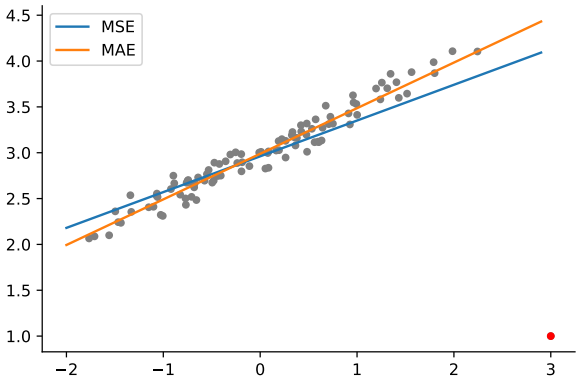

Figure 1: MSE vs. MAE estimators for a dataset with 1 outlier (red dot)

2 Mean-Absolute-Error Penalized Regression

There exist different methods to construct the estimator $\widehat{\theta}$. The most widely used approach minimizes the mean-squared-error (MSE) corresponding to the given data to obtain the MSE estimator

$$\widehat{\theta}_{\text{MSE}} \in \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} (y^{(i)} - \theta^\top x^{(i)})^2. \tag{1}$$

Alternatively, one can minimize the mean-absolute-error (MAE) to obtain the more robust MAE estimator

$$\widehat{\theta}_{\text{MAE}} \in \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} |y^{(i)} - \theta^\top x^{(i)}|. \tag{2}$$

Question 1 (5 points)**.** *Figure 1 visualizes the outputs predicted by the MSE and MAE estimators on a dataset with 1 outlier. Explain the difference between the two estimators intuitively. Hint: Compare how (1) and (2) penalize errors.*

In case of high-dimensional inputs ($d > N$), it makes sense to seek a *sparse* parameter vector θ. This means that many components of θ should be zero. The non-zero components of θ then correspond to the key features or key inputs that have a significant impact on the outputs. Sparsity can be induced by adding an ℓ_1-penalty to the objective function of (1) or (2), which yields the least absolute shrinkage and selection operator (LASSO) method of regression analysis. In case of the MAE objective, the resulting LASSO estimator satisfies

$$\widehat{\theta} \in \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} |y^{(i)} - \theta^\top x^{(i)}| + \lambda \|\theta\|_1, \tag{3}$$

where $\|\theta\|_1 = \sum_{i=1}^{d} |\theta_i|$ denotes the ℓ_1-norm of θ, and $\lambda \geq 0$ is a hyperparameter that represents the weight of the penalty. In practice, λ is typically chosen by cross validation. That is, we randomly partition $\mathcal{D}$ into a training dataset $\mathcal{D}_{\text{train}}$ and a validation dataset $\mathcal{D}_{\text{val}}$. We then solve (3) using only $\mathcal{D}_{\text{train}}$ for different values of λ and select the estimator that has minimal MAE on $\mathcal{D}_{\text{val}}$.

Question 2 (15 points: 5+5+5)**.**

1. *Rewrite* (3) *as a linear program (not necessarily in standard form).*
2. *Implement this linear program in* MATLAB *or* Python *and solve it on the* `Diabetes Dataset` *for* $\lambda = 0.5$*. Use the code skeletons provided on Moodle. The diabetes dataset links a feature vector* x *comprising patient information (age, sex, body mass index) and blood measurements (blood pressure, T-Cells, lipoproteins, thyroid, lamotrigine and blood sugar) to an output* y *quantifying the progression of the diabetes disease. Compute the MAE of the resulting estimator on the separate test sets provided on Moodle. Compare the test performance against the training performance. Hint: Do not forget to append a constant bias term to the features.*
3. *Use cross-validation to tune the hyperparameter* λ*. Randomly select 75% of the data to construct* $\mathcal{D}_{\text{train}}$ *and use the rest of the data to construct* $\mathcal{D}_{\text{val}}$*. Use 50 logarithmically spaced values between* $[10^{-5}, 10^{-1}]$ *as candidates for* λ*, select the one performing best on the validation set in terms of MAE. Compare again the test performance against the training performance.*

3 Convex Hulls

Standard linear programs involve only real decision variables. However, many applications require binary decision variables that are restricted to $\{0, 1\}$. Such variables emerge, for example, if you have to decide whether a project should be implemented or not or whether an item should be loaded on a truck or not etc. A linear optimization problem with binary decision variables is given below.

$$\begin{aligned} \min_{x_1, x_2} \quad & -x_1 - 2x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 1 \\ & x_1, x_2 \in \{0, 1\} \end{aligned} \tag{4}$$

Any binary linear program (such as problem (4)) is equivalent to a standard linear program, which is obtained by replacing the original feasible set with its convex hull, that is, the smallest convex set containing all feasible points.

Definition 3.1 (Convex hull)**.** *The convex hull of an arbitrary (possible non-convex) set* $X \subset \mathbb{R}^n$ *is defined as*

$$\text{conv}(X) = \left\{ z \in \mathbb{R}^n : z = \sum_{i=1}^{n} \theta_i x_i, \;\; \sum_{i=1}^{n} \theta_i = 1, \;\; x_i \in X, \, \theta_i \geq 0 \;\forall i \in [n] \right\}.$$

Question 3 (5 points). *Denote by X the feasible set of problem (4). Describe the convex hull of X and find its vertices. Plot both X and* $\mathrm{conv}(X)$.

Question 4 (5 points). *Replace the feasible set of problem (4) with its convex hull and solve the resulting linear program using* MATLAB *or* Python. *Report the optimal decision variables. Explain why they must be binary. Hint: Characterize the BFS of the convex hull of the feasible set.*

4 Zero-Sum Games

In the standard cross validatioin scheme described in Section 2, training and validation sets were constructed randomly. In the remainder of the project we will explore an adversarial cross validation procedure, where the training set is chosen by a fictitious adversary who aims to maximize the prediction error of our estimator. This procedure will give rise to a MinMax problem of the form:

$$\min_{x\in\mathcal{X}} \max_{z\in\mathcal{Z}} \quad f(x,z). \tag{5}$$

MinMax problems are zero-sum games, where one player's profit is the other player's loss. They are more intricate than standard minimization problems. We will see, however, that problem (5) can sometimes be simplified to a standard minimization problem if $\mathcal{Z}$ is convex and $f(x,z)$ is concave in z for all $x\in\mathcal{X}$.

As an example, suppose you own \$10,000, which you want to hide in your apartment. There are I hiding spots with capacities of C_i dollars, $i\in[I]$. In case of a burglary, you want to lose as little money as possible. Assume that the police need T minutes to reach your apartment, and therefore any thief must leave within T minutes to avoid arrest. Assume further that the amount of money found in hiding spot i equals $x_i\cdot z_i\cdot p_i$, where x_i stands for the amount of money hidden, z_i denotes the amount of time the thief spends searching spot i and p_i represents a constant reflecting the difficulty of searching spot i. If the thief spends an amount of time exceeding $1/p_i$ in location i, then all the money hidden in that spot will be found. It therefore makes sense to restrict $z_i\le 1/p_i$.

We seek a plan for hiding the money in the best possible way. Specifically, we hope to loose the least amount of money in the worst case, that is, if a very efficient thief shows up. This problem can be modeled as a zero-sum game against the thief (indeed, our loss is the thief's profit).

Question 5 (20 points: 3+3+3+8+3). *This question will guide you through formulating and solving the problem of finding the best hiding strategy.*

1. *Characterize your decision variables and your feasible set.*
2. *Characterize the thief's decision variables and feasible set.*
3. *Use the solutions of the first two questions to formulate a MinMax problem with objective function*

$$f(x,z)=\sum_{i=1}^{I} x_i z_i p_i.$$

4. *Dualize the inner maximization problem to reformulate the MinMax problem as a standard minimization problem, which can be addressed with linear programming solvers. Hint: The decision variables of the outer minimization problem represent constants for the inner maximization problem.*

5. *Implement the resulting linear program in* Python *or* MATLAB *and solve it for the provided input data $p, C \in \mathbb{R}^I$ and $T \in \mathbb{R}$. Use the code skeletons available from Moodle. Report the worst amount of money you lose, the optimal plan for hiding the money and the thief's optimal search strategy. Hint: Think about the meaning of the problem's dual variables.*

5 Adversarial Training

Standard cross validation partitions $\mathcal{D}$ randomly into a training set $\mathcal{D}_{\text{train}}$ and a validation set $\mathcal{D}_{\text{val}}$ of prescribed cardinalities. Hence, it could happen that the training set contains no outliers (lucky) or all outliers (unlucky). The resulting estimator $\widehat{\theta}$ thus depends on the choice of the training set. We now propose an alternative approach to construct $\widehat{\theta}$ in an adversarial (worst-case optimal) manner. Specifically, we will train the regression model on the worst possible training set $\mathcal{D}_{\text{train}}$ we could have possibly constructed from $\mathcal{D}$.

We now construct the best estimator for the worst-case training set containing $k = \lfloor 0.75 \cdot N \rfloor$ samples, where $\lfloor y \rfloor$ denotes the largest integer smaller or equal to y. To that end, we formulate a MinMax problem, where the inner maximization problem optimizes over the binary variables z_i, $i \in [N]$, corresponding to the N samples. The binary variable z_i is set to 1 if sample $(y^{(i)}, x^{(i)})$ is included in the training set and to 0 otherwise. The outer optimization problem over θ aims to minimize the MAE with LASSO penalty in view of the worst-case training set. In summary, we thus find the following MinMax problem.

$$\begin{aligned} \min_{\theta} \max_{z_1,\ldots,z_N} \quad & \frac{1}{k} \sum_{i=1}^{N} z_i |y^{(i)} - \theta^\top x^{(i)}| + \lambda \|\theta\|_1 \\ \text{s.t.} \quad & \sum_{i=1}^{N} z_i = k \\ & z_i \in \{0,1\} \quad \forall i \in [N] \end{aligned} \tag{6}$$

This MinMax problem can be interpreted as a zero-sum game between the statistician and an evil adversary who selects the worst possible training set.

Question 6 (15 points)**.** *Derive the convex hull of the adversary's feasible set*

$$Z = \left\{ z \in \mathbb{R}^N : \sum_{i=1}^{N} z_i = k,\ z_i \in \{0,1\}\ \forall i \in [N] \right\}, \tag{7}$$

where k is an integer. Prove that your formulation is indeed the convex hull. Hint: Two sets A and B are equivalent if $A \subseteq B$ and $B \subseteq A$.

Question 7 (25 points: 10+10+5)**.**

1. *Show that the MinMax problem* (6) *is equivalent to the following linear program. Hint: Use the insights from Sections* 3 *and* 4*.*

$$\begin{aligned} \min_{\theta,\alpha,\beta_i,b_i} \quad & k\alpha + \sum_{i=1}^{N} \beta_i + \lambda \sum_{j=1}^{d} b_i \\ \text{s.t.} \quad & \alpha + \beta_i \geq (y^{(i)} - \theta^\top x^{(i)})/k \quad \forall i \in [N] \\ & \alpha + \beta_i \geq -(y^{(i)} - \theta^\top x^{(i)})/k \quad \forall i \in [N] \\ & \beta_i \geq 0 \quad \forall i \in [N] \\ & -b_j \leq \theta_j \leq b_j \quad \forall j \in [d] \end{aligned} \tag{8}$$

   Give an interpretation for β*. Explain how one can extract the validation set from a solution of this problem.*

2. *Implement the linear program* (8) *in* MATLAB *or* Python *using the skeleton code we provide on Moodle and solve it for 50 logarithmically spaced values of* λ *in the interval* $[10^{-5}, 10^{-1}]$*. For each of these values compute the MAE of the corresponding estimator on the validation set, and select the best robust estimator. To assess the benefits of robust cross validation over standard cross validation, compare the MAE of the robust estimator against the MEA of the estimator found in Question* 2.3*. In both cases the MEA should be computed on the test set.*

3. *For the robust estimator found above, compare also the MEA on the test set against the MEA on the training set. Is there a significant difference to what you observed in Question* 2.3*.? Interpret your observations?*

6 No More Toy-Examples/Optimization Prize

So far, you have applied different regression techniques to the "Diabetes" dataset. In this open question, we ask you to apply one of the regression techniques seen in Sections 2 and 5 to a regression problem of your liking.

Question 8 (10 points)**.** *Apply one of the regression techniques seen in Sections* 2 *and* 5 *to a dataset of your choice. Hint: Good sources of datasets include Kaggle and the UCI archive. In answering this question, put emphasis on interpreting your results and on justifying the chosen optimization scheme.*

Question 9 (Voluntary bonus question, 10 points)**.** *If you like, prepare a short presentation of your answer of Question* 8 *(about 5 minutes) for the exercise session on May 20th. This will allow you to present your dataset and insights to your fellow students. We will give you bonus points for volunteering to do the presentation, but we will not grade its contents. Everyone present in the exercise session will be able to vote on the best presentation. The group attracting the most votes will receive an optimization prize.*