1 Introduction

This is an investigation into alternative loss functions for linear regression. The traditional loss function to use is squared Euclidean distance. This loss can lead to problems in some situations though, such as when overestimation and underestimation are not equally costly.

Here we consider linear regression with loss function chosen by an adversary from a particular class of Bregman loss functions. Since the loss function is chosen adversarially, the regression algorithm must minimize loss in the worst case.

2 Bregman loss functions

Bregman loss functions are derived from a “helper” function called a link function.

The Bregman loss function associated with a non-decreasing link function $h : \mathbb{R} \to \mathbb{R}$ is given by

$$L_h(\hat{a}, a) = \int_a^{\hat{a}} h(z) - h(a)dz$$

$$= H(\hat{a}) - H(a) + (a - \hat{a})h(a),$$

where $H(z) = \int h(z)dz$. In pictures, the loss is given by the area of the shaded regions:
We can also think of this definition of \( L_h \) as the difference between \( H \) and its linear (first-order Taylor) approximation at \( a \):

The traditional loss function, squared Euclidean distance, is a Bregman loss function. It is derived from the link function \( h(a) = a \).

In this paper, we will be employing the restriction that \( h(a) \in [0, 1] \) for all \( a \in \mathbb{R} \). This restriction is reasonable because otherwise the adversary could choose arbitrarily “harsh” loss functions. Note that squared Euclidean distance does not meet this criterion.

3 The problem statement

Let \( x_1, \ldots, x_n \in \mathbb{R}^m \) be a sequence of examples with labels \( a_1, \ldots, a_n \in \mathbb{R} \). Assume without loss of generality that the labels are sorted, so if \( i \leq j \) then \( a_i \leq a_j \). Then compute

\[
\min_{w \in \mathbb{R}^m} \max_h \sum_{i=1}^{n} L_h(w \cdot x_i, a_i)
\]

where \( h : \mathbb{R} \to [0, 1] \) is a non-decreasing link function.

Note that the adversary is only choosing the link function (and hence the loss function). In particular, the adversary is not generating the data: the examples and labels are held fixed and are known to both the algorithm and the adversary.
4 Concrete example

The feature values and labels are given by the following table:

<table>
<thead>
<tr>
<th>$x$</th>
<th>$a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

So the data consists of two examples, each with the same feature value but with different labels.

As we shall see, the optimal link function for the adversary to choose is

$$h(a) = \begin{cases} 
0 & \text{if } a \leq 1 \\
\frac{1}{2} & \text{if } 1 < a < 2 \\
1 & \text{if } a \geq 2 
\end{cases}$$

and the optimal choice for the regression algorithm is $w = \frac{3}{2}$.

With these choices of $w$ and $h$, we find that the total loss is

$$L_h \left( \frac{3}{2}, 1 \right) + L_h \left( \frac{3}{2}, 2 \right) = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

by referencing this picture:

![Diagram](image)

We can see that $w = 1.5$ is optimal for this $h$ because decreasing the width of one shaded region will increase the width of the other region commensurately. Likewise, this $h$ must be optimal for $w = 1.5$ because increasing the height of one region will decrease the height of the other.

One might wonder whether the adversary would do better to pick something more complicated than a step function. The picture suggests otherwise (recall that the link function is required to be non-decreasing), and in fact we will now prove that the adversary only needs to consider step functions.
5 Step functions are optimal

In fact we can restrict our attention to link functions that are everywhere constant except at the labels $a_i$.

Proof. The key idea is to take an optimal link function $h$ and build a new one $\tilde{h}$ by replacing any non-constant portion on $[a_i, a_{i+1}]$ in the following way:

such that the shaded regions have equal area. This works because the loss

$$L_h(\hat{a}, a) = H(\hat{a}) - H(a) + (a - \hat{a})h(a)$$

has only one term (the first) that depends on the portion between $a_i$ and $a_{i+1}$, and this term can only increase.

To see that $\tilde{H}(\hat{a}) \geq H(\hat{a})$, we invoke the fact that $H$ is convex (because $h$ is non-decreasing). $\tilde{H} = \int \tilde{h}$ is a secant line approximation to $H$ on $[a_i, a_{i+1}]$, as shown in this picture:

This result is extremely important because it means that the space of options available to the adversary is actually a simplex of $2n$ dimensions. In particular, we can parameterize the space of feasible link functions by $c_1, \ldots, c_{2n}$ as shown in this diagram ($n = 2$):
These parameters are obviously non-negative. Furthermore, they clearly sum to 1 due to the obvious fact that optimal link functions should be identically 0 for inputs $a < a_1$ and identically 1 for inputs $a > a_n$.

## 6 Pure strategy for the adversary

Consider the inner problem

$$\max_h \sum_{i=1}^n L_h(w \cdot x_i, a_i)$$

where $w$ is kept fixed and is known to the adversary selecting $h$. It turns out that the objective function of this problem is linear in the parameters $c_1, \ldots, c_n$ (for fixed $w$ only). To see this, consider this picture:

Since $w$ is fixed, the width of each rectangle is fixed. The parameters $c_1, \ldots, c_n$ control only the heights of the rectangles.

The strategy space of the adversary is a simplex, and the objective function being maximized is linear. This means that the inner problem is a linear program, and therefore the solution must be one of the corner points of the simplex where $c_i = 1$ for some $i$. This means that the optimal link function has just a single step!
To borrow terminology from game theory, we can say that the adversary has an optimal pure strategy. It is pure in the sense that exactly one of the parameters is non-zero.

This result is somewhat surprising in light of the example problem shown earlier. The fact of the matter is that this result depends crucially on the adversary’s knowledge of the weight vector $w$ – without this knowledge, the adversary must sometimes choose a mixed strategy (where more than one parameter is non-zero), and the example shown earlier illustrates this situation. A link function with just a single step does not give rise to a very interesting loss function, so for the purposes of studying loss functions, we would like to avoid these pure solutions by forcing the adversary to choose without knowledge of $w$.

7 Future work

To prevent the adversary from selecting a pure strategy, we must study the dual problem:

$$\max_h \min_{w \in \mathbb{R}^m} \sum_{i=1}^n L_h(w \cdot x_i, a_i).$$

This problem is slightly harder to study than the original because the inner problem is not linear, even when $h$ is kept fixed. It is piecewise linear, however, and so it may be possible to apply the standard tricks for converting a piecewise linear problem into a linear one.

An alternative approach would be to compute the dual of the original inner problem – every linear program has a dual linear program. The dual variables in this problem will not be the weights in $w$, so this computation would require adaptation before being applied to the max-min problem.