Linear Models of Classification

Andres Ponce

March 25, 2020

The goal with **classification** is "to take an input vector x and to assign it to one of K discrete classes.

If we want to classify a certain input vector, we would have to map it to a certain region of a plane, defined by certain boundaries.[1]

If there are D varaiables we are trying to optimize, then our answer will exist in a D dimensional space.

[1] Is this similar to **linear programming**, where we had to optimize a linear combination of the parameters?

Discriminants

Linear discriminants can be of the form[2]

$$y(x) = w^T x + w_0$$

[2] Here, w is the parameters, and w_0 is the bias.

We also ahve a **decision boundary**, where we know which of the K categories our input belongs to given our discriminant function, and a certain cutoff value C_x.

We can also use a Bayesian approach to determine classification, if we calculate $p(C_k|x)$[3]

[3] The probability that given input x, it belongs in class k.

How do we extend a linear discriminant to other classes? We can have **One-vs-the-rest** or **one-vs-one** qualifiers.

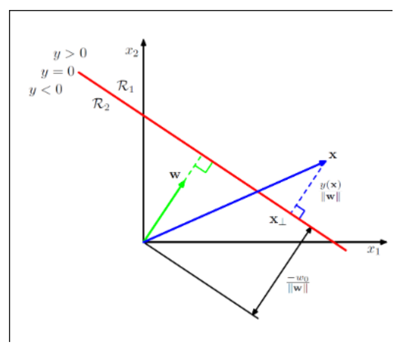

Figure 1: The dotted green and blue lines are the distance between the arbitrary point and the decision boundary.

One-vs-the-rest

With this type of classifier functions, we try to distinguish those inputs on a binary case by case basis. That is, we try to one at a time build a classifier that knows points in C_i from those points in other classes. However, when two points "not in their repsective classes" fall in the same area, this area would be defined differently by the areas we're actually testing.

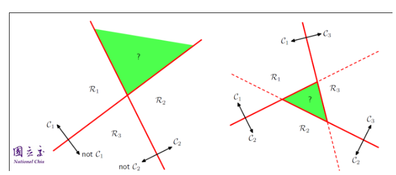

Figure 2: When a point is not in a certain region, it can be ambiguous where it falls, if the same point can be thought of differently by each region.

K-class discriminant

If we use K linear functions of the form

$$y_k(x) = w_k^T(x) + w_{k0}$$

Then point x goes in class k iff $y_k(x) > y_j(x) \forall j \neq k$ [4]

[4] essentially we describe every region by its own equation, and assign the points to the one that fits the best.

Least Squares for classification

Similar to the original least squares definition, we want to minimize the difference between the data points and the final values.

The actual formula for the least squares classification is

$$y(x) = \widetilde{W}^T \tilde{x}$$

However, the least squares solution usually does not have the best performance, and is sensitive to **outliers**.

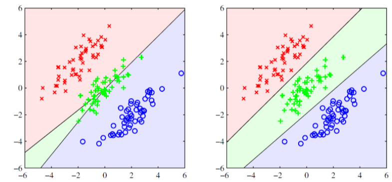

Figure 3: Logistic regression can sometimes have better performance than ordinary least squares.

Fisher's Linear Discriminant:2 classes

Fisher's Linear Discriminant can help with dimensionality reduction at the moment of solving a K-dimensionality problem. With these kinds of problems, we want to find a lienar combinations of elements that maximizes the distance between data points not in the same class.

The **mean vectors** between two classes is

$$m_1 = \frac{1}{N_1} \sum_{n \in C_1} x_n$$

and likewise for m_2. Then, we want a function that maximizes the distance between the two functions. We would like

$$m_2 - m_1 = w^t(m_2 - m_1)$$

to be maximized.We also want to minimize the variance within a single class.

So, there are a coupple ways to carry out classification:

- Determine a threshold:We can find a point x such that if $y(x) \geq -y_0$, C_1, else C_2.
- use the nearest-neighbor rule:When we project a training sample onto a one-dimensional space, we can sample some of the training data a certain distance away, and classifying the point based on the number of training points belonging to a certain class.

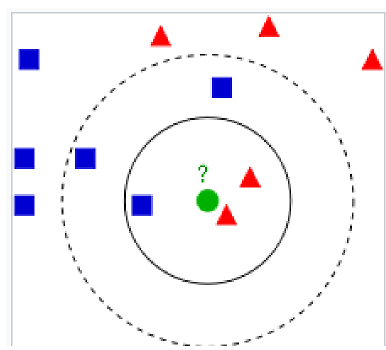

Figure 4: If we take the sample space to be the solid colored line, we say the green sample point is a red triangle, since the closest points are mostly red triangles(2-1). Using the dahsed circle, we would classify it as a blue square for the same reason.