CS 337

Parth Pujari
210100106

31st August 2023

1 Regularized Logistic Regression

$$\mathbf{w}_{\mathbf{R}}^{*} = \text{argmin} L_{CE}(\mathbf{w}, \mathcal{D}) + \lambda \|\mathbf{w}\|_2^2$$

Suppose we have the following equation of hyperplane :

$$x_1 - 2x_2 = 0$$

The following are acceptable values of w_1 and w_2, where

$$\mathbf{w} = [w_1, w_2]^T, \text{and} \mathbf{w}^{\mathbf{T}}\mathbf{x} = 0$$

1. $w_1 = 1, w_2 = -2$
2. $w_1 = 10, w_2 = -20$
3. $w_1 = 10^5, w_2 = -2 \times 10^5$

While all these values seem to satisfy the equation, it is important to note that points which are far from the decision boundary(in this case, the hyperplane), on its positive side, get assigned a higher probability by the classifier.

$$\mathcal{P}(y|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x}) = \frac{1}{1 + e^{(-\mathbf{w}^T\mathbf{x})}}$$

For a constant $\mathbf{x}$, $\mathbf{w}^T\mathbf{x}$, if positive increases with the scaling in $\mathbf{w}$. In this case, $\mathcal{P}(y|\mathbf{x}, \mathbf{w})$ will tend towards 1. Thus, classifier will prefer higher values of $\mathbf{w}$ but won't be able to differentiate between a point near the hyperplane from one which is farther away from it. We need to regularize $\mathbf{w}$ and make sure that it does not become arbitrarily large, so that we have a better calibration. Hence, regularization is used. We are in a way making the variance smaller by limiting $\mathbf{w}$.

Logistic Regression provides us with a satisfactory method for feature interpretability. It is fairly intuitive to understand and interpret decision boundaries. However, it is not easy to do so in the case of non-linear decision boundaries. Although we can use feature transformation along with Logistic regression, it still leaves us wanting.

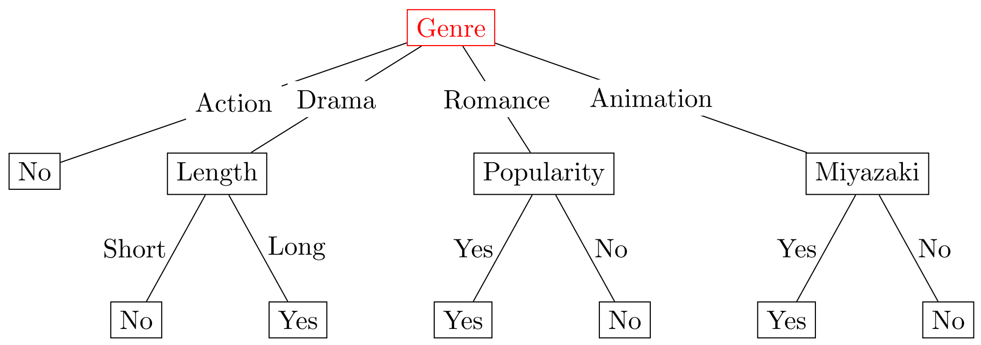
Desiderata

To overcome these shortcomings of Logistic Regression, we need :

- The ability to learn complex decision boundaries.
- Interpretability of the model, to be able to understand how useful each of the features/attributes is.

2 Decision Tree Classifiers

Decision Trees are an interpretable model whose final prediction can be written as "a disjunction of conjunctions", based on attribute values over training instances.

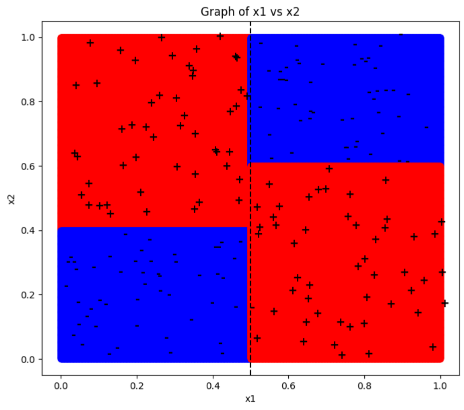


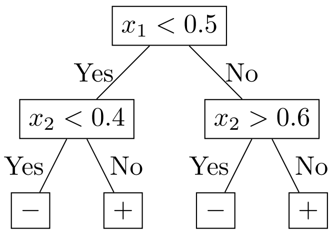
Each path in the tree is a conjunction of attribute-based constraints. The complete tree is a disjunction of these conjunctions. The expression for the above tree is :

$$(G = \mathit{Drama} \wedge L = \mathit{short}) \vee (G = \mathit{Romance} \wedge \mathit{Popularity} = Yes)$$
$$\vee (G = \mathit{Animation} \wedge \mathit{Miyazaki} = Yes)$$

3 Decision Tree Boundaries

- At each node, a decision tree divides the feature space into hyper-rectangles, when the condition on that node contains a single variable. The resulting decision boundaries are axis-parallel hyperplanes. [1]
- Finding the smallest "optimal" Decision Tree w.r.t some metric involving all attributes is NP-Hard.
- Decision Tree estimation is largely **Greedy**. We will be recursively building the tree.





[1]The space can be divided into other shapes by suitable conditions containing multiple variables. For example, $ax1 + bx2 < 0.2$ divides the region using non axis parallel hyperplanes

4 Simple Decision Tree Template

1. Start from an empty tree with all instances.
2. Pick the **best** attribute to split on.
3. Repeat Step 2, recursively, for each new node until a **stopping criterion** is met.

5 Two Important Questions

1. What is the notion of a "**best**" attribute and how to find it?
2. What is a **good** stopping criterion?

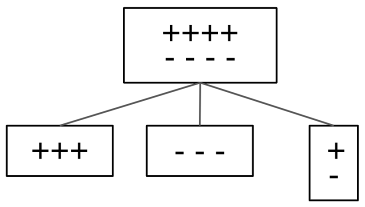


Figure 1: Homogenous split

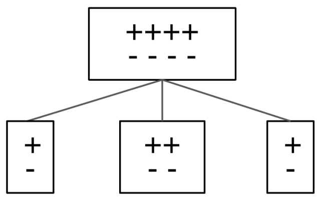


Figure 2: Non homogenous split

Here, the first tree is better, because the nodes are relatively more homogeneous, compared to the second one. If we are not able to separate + and − from each other, we are not gaining any new information by increasing the depth of the tree. It is redundant. Also, we don't want the tree to have too much depth, or be too detailed, as that leads to overfitting. It should be as small as possible.

INTUITION : A good split for an attribute results in subsets that are nearly all positive or all negative.

6 Information Gain

$$\boxed{H(X) = -\sum_{x\in X} P(X=x)\log_2 P(X=x)}$$

Here H(X) is the **entropy** of a random variable X. The unit of entropy is bits (taking the base as 2).

A higher entropy represents a nearly uniform distribution, which means higher uncertainty. On the other hand, lower entropy implies that the underlying distribution has a well-defined mode. This means that it has lower uncertainty and higher predictability. Entropy of a dataset S is :

$$\boxed{H(S) = -\sum_{i=1}^{c} P_{i,s} log P_{i,s}}$$

where $P_{i,s}$ is the relative count of instances in S with label i.

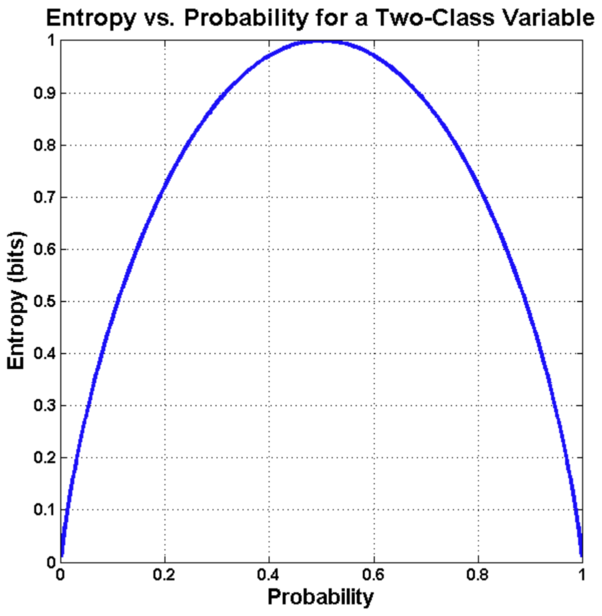


Figure 3: *Entropy of a fair coin*

$$\boxed{Gain(S,a) = H(S) \;-\; \sum_{v\in vals(a)} \frac{|S_v|}{|S|} H(S_v)}$$

where S_v is the subset of S whose instances all have attributes 'a' taking the value 'v'.