Gravity Based Multicatigorical Classification

Barrett Duna

September 14, 2020

1 Introduction

We borrow the concept of gravitational pull from Physics to form a classifier that classifies based on maximal gravitational pull measured against each class. We define the mass of each point in the data set as the reciprocal of the squared distance from the dataset point to our new data point to be classified. Gravitational pull is calculated separately for each class and the class that exerts the strongest gravitational pull is the prediction.

2 The Gravitational Pull Formula

Suppose we have m classes $C_1, C_2, \ldots, C_m$ and dataset X containing numeric features with each data point in X corresponding to a class C_i. Then our formulation of the gravitational pull of class C on vector $\vec{x}$ is...

$$F(\vec{x}, C) = \frac{\Sigma_{i=1}^{n_C} \frac{1}{\|\vec{x} - x_C^i\|^2}}{\|\vec{x} - C_{centroid}\|^2}$$

where n_C is the number of data points in class C, x_C^i is the i^{th} data point of class C and $C_{centroid}$ is the centroid of class C

3 Background on the Gravitational Pull Formula

The gravitational force pulling two objects together is given by the following equation...

$$F = G\frac{m_1 m_2}{r^2}$$

Here G is the gravitational constant which we drop because each calculation of the gravitational pull on $\vec{x}$ by class C_i involves the constant so it's safe to omit it. The constants m_1 and m_2 are the masses of the two objects and r is the distance between the two objects. We see that as mass increases and the distance between the two objects decreases the gravitational force increases.

4 Interpreting our Gravitational Pull Formula

$F(\vec{x}, C)$ can be interpreted as an application of the gravitational force equation. Suppose we are calculating the gravitational pull of class C on $\vec{x}$.

We have n_C data points in class C which spread across the input feature vector space. Instead of having two objects, we have n_C class C data points representing our first object and one vector $\vec{x}$ representing our second object.

To calculate the mass of class C we sum the masses of each data point in C. If m_C is the total mass of class C and m_C^i is the mass of the i^{th} data point, then the total mass is...

$$m_C = \Sigma_{i=1}^{n_c} m_C^i$$

In our gravitational pull formula the masses of each data point in each class change depending on the location of $\vec{x}$...

$$m_C^i = \frac{1}{\|\vec{x} - x_C^i\|^2}$$

Thus, as the distance between $\vec{x}$ and x_C^i grows, the mass of the i^{th} data point in C decreases leading to a lower gravitational pull holding all else constant. This means points further away from x are weighted less.

Combining the above two equations we see...

$$m_C = \Sigma_{i=1}^{n_c} m_C^i = \Sigma_{i=1}^{n_c} \frac{1}{\|\vec{x} - x_C^i\|^2}$$

Reviewing the gravitational force equation and substituting m_C for m_1, dropping the constant G and denoting the mass of $\vec{x}$ as m_x...

$$F(\vec{x}, C) = \frac{m_C m_x}{r^2}$$

We hold the mass of $\vec{x}$, i.e. m_x constant and note that this constant is multiplied by every calculation of the gravitational pull for each class. Thus, we can safely set $m_x = 1$ producing...

$$F(\vec{x}, C) = \frac{m_C}{r^2}$$

To calculate r we substituted the distance between $\vec{x}$ and the centroid of C because the centroid of C is the center of mass of the class when all points are equally weighted. It is also the point that minimizes the sum of squared distances between the calculated centroid and all class data points.

Technically, to be true to the gravitational force calculation, we should weight all points in the class by their mass which isn't the same for every point. However, we found better classification performance by simply weighting each point equally. The equally weighted centroid is a better representation of the center of the class. The formula we used for calculating the centroid is...

$$C_{centroid} = \frac{\Sigma_{i=1}^{n_C} x_C^i}{n_C}$$

Combining this we get...

$$F(\vec{x}, C) = \frac{m_C}{\|\vec{x} - C_{centroid}\|^2}$$

Giving us the original calculation...

$$F(\vec{x}, C) = \frac{\Sigma_{i=1}^{n_C} \frac{1}{\|\vec{x} - x_C^i\|^2}}{\|\vec{x} - C_{centroid}\|^2}$$

5 Deriving the Class Point Masses

While it's intuitive to weight each class point by the reciprocal of it's distance to the point to be classified, there is mathematical justification for this.

Intuitively, we'd like to minimize the point masses while penalizing decreases in gravitation pull. Larger point masses lead to stronger gravitational pull which biases our predictions to the class with the largest point masses. Smaller gravitational pull biases our predictions towards classes with stronger gravitational pull. Thus, we need to optimize an objective function that considers the trade off between the two competing forces and treats all classes equally.

Consider the objective function $J(\vec{m_C})$ where $\vec{m_C}$ is the vector of point weights from class C...

$$J(\vec{m_C}) = \Sigma_{i=1}^{n_C} [\frac{1}{2} {m_C^i}^2 - \frac{m_C^i}{\|\vec{x} - x_C^i\|^2}]$$

We will minimize $J(\vec{m_C})$. Since each term in the sum only involves a single point mass, differentiating with respect to m_C^i gives...

$$\frac{J(\vec{m_C})}{dm_C^i} = \frac{d}{dm_c^i} [\frac{1}{2} {m_C^i}^2 - \frac{m_C^i}{\|\vec{x} - x_C^i\|^2}]$$

Intuitively, we see minimizing this function optimizes the trade off between point mass and gravitational pull. The first term minimizes point mass of m_C^i while the second term is the gravitational pull between $\vec{x}$ and x_C^i which penalizes decreases in gravitational pull between $\vec{x}$ and the i^{th} point in class C.

Completing the derivative we see...

$$\frac{J(\vec{m_C})}{dm_C^i} = m_C^i - \frac{1}{\|\vec{x} - x_C^i\|^2}$$

Setting the derivative equal to zero produces the desired point mass calculation...

$$m_C^i = \frac{1}{\|\vec{x} - x_C^i\|^2}$$

Taking the second derivative...

$$\frac{J(\vec{m_C})}{d^2 m_C^i} = 1 > 0$$

Thus our function is concave up and we have reached a minimum.

6 Identities and Inequalities of the Gravity Classifier

We will begin by calculating the expected squared distance $E[\|\vec{x} - \vec{x_C}\|^2]$ between the point $\vec{x} \in R^n$ to be classified which we fix and the random variable $x_C \in R^n$ from class C with probability distribution P_C.

Note..

$$\|\vec{x} - \vec{x_C}\|^2 = \|\vec{x}\|^2 - 2\vec{x} \cdot \vec{x_C} + \|\vec{x_C}\|^2$$

Thus...

$$E[\|\vec{x} - \vec{x_C}\|^2] \quad = \quad E[\|\vec{x}\|^2] - 2E[\vec{x} \cdot \vec{x_C}] + E[\|\vec{x_C}\|^2]$$

Observe...

$$\begin{aligned} E[\vec{x} \cdot \vec{x_C}] &= E[\Sigma_{i=1}^n x^i \cdot x_C^i] \\ &= \Sigma_{i=1}^n E[x^i \cdot x_C^i] \\ &= \Sigma_{i=1}^n x_i E[x_C^i] \\ &= \vec{x} \cdot < E[x_C^1], ..., E[x_C^n] > \\ &= \vec{x} \cdot E[\vec{x_C}] \\ &= \vec{x} \cdot C_{centroid} \end{aligned}$$

Since $\|x\|^2$ is a constant, $E[\|x\|^2] = \|x\|^2$.

Combining all of this, we obtain...

$$E[\|\vec{x} - \vec{x_C}\|^2] = \|\vec{x}\|^2 - 2\vec{x} \cdot C_{centroid} + E[\|\vec{x_C}\|^2]$$

Continuing...

$$\|\vec{x} - C_{centroid}\|^2 \quad = \quad \|\vec{x}\|^2 - 2\vec{x} \cdot C_{centroid} + \|C_{centroid}\|^2$$

7 Classifying Multivariate Gaussian Clusters

In this section, we look at a special case where we assume a cluster is an n-dimensional multivariate Gaussian random variable, i.e. $X_C \sim N(\mu, \Sigma)$.

We make use of the following fact about multivariate Gaussians...

$$E[\|x_C\|^2] = \|\mu\|^2 + tr(\Sigma)$$

Note, μ is the expected value of the distribution and hence the centroid.

Thus...

$$E[\|x_C\|^2] = \|C_{centroid}\|^2 + tr(\Sigma)$$

and...

$$E[\|\vec{x} - x_C\|^2] = \|x\|^2 - 2\vec{x} \cdot C_{centroid} + \|C_{centroid}\|^2 + tr(\Sigma)$$

considering...

$$\|\vec{x} - C_{centroid}\|^2 = \|\vec{x}\|^2 - 2\vec{x} \cdot C_{centroid} + \|C_{centroid}\|^2$$

we see...

$$E[\|\vec{x} - x_C\|^2] = \|\vec{x} - C_{centroid}\|^2 + tr(\Sigma)$$

F

8 Gravity Classification Boundary Area Plots

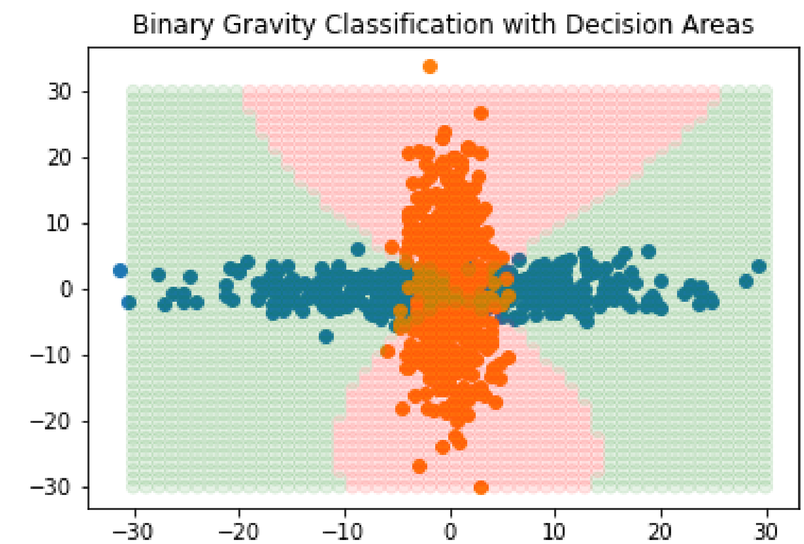

Figure 1: Non-Linear Decision Areas Produced By Gravity Algorithm

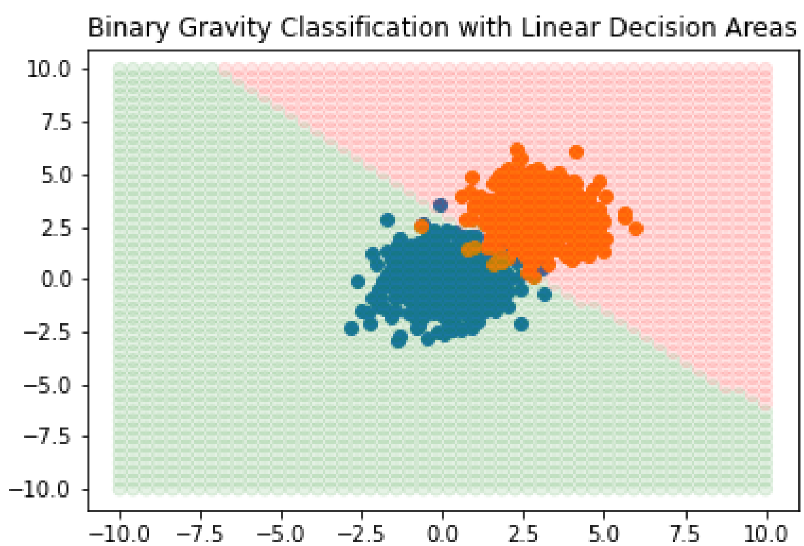

Figure 2: Linear Decision Areas Produced By Gravity Algorithm

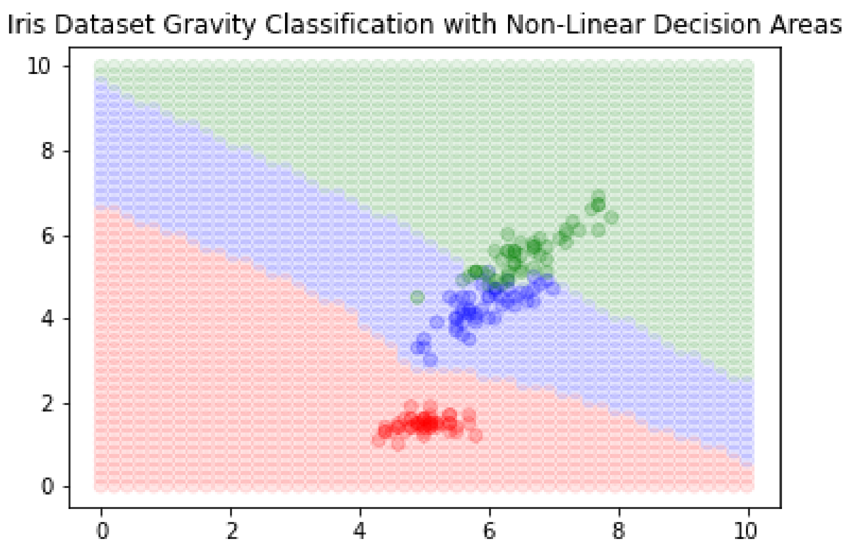

Figure 3: Non-Linear Decision Areas Produced By Gravity Algorithm on Three Category Iris Dataset