Gravitational Clustering: A Novel Approach for Efficient Supervised Learning with Minimal Data

Dinesh Kumar Koilada
Independent Researcher
dineshkoilada@gmail.com

***Abstract*—Traditional supervised learning algorithms, such as neural networks and support vector machines, often struggle when training data is limited or when dealing with multi-class classification tasks. In response to these challenges, this paper introduces Gravitational Clustering, a novel algorithm that eliminates the need for predefined cluster numbers and effectively learns from small datasets. Drawing inspiration from gravitational physics, this method models each cluster as a planet with mass, radius, and class, allowing for dynamic cluster formation without the risk of overfitting. Key advantages include the ability to weight feature vectors, handle minimal data samples, and maintain resilience against overfitting. The algorithm demonstrates competitive performance across multiple datasets, achieving higher classification accuracy while maintaining lower computational complexity compared to traditional methods such as K-Means and support vector machines. This paper explores the algorithm's theoretical foundations, computational efficiency, and empirical results, offering a robust solution for classification tasks with limited data availability.**

I. Introduction

The name of this calculation, *Planetary Classification*, is gotten from the similitude that fills in as the establishment for its plan. In this reasonable system, each group in the dataset is addressed as a planet, each with its own mass, span, and class. The planets, in any case, vary from true divine bodies in that they stay static comparative with each other all through the calculation's activity. This trademark adds a theoretical straightforwardness to the demonstrating system while keeping up with the adaptability important for viable learning and expectation. The preparation cycle can be pictured as the development of a universe: a space loaded up with planets, each addressing a class of data of interest. During this stage, the calculation recognizes the critical qualities of each class and places them in the comparing planets. When the preparation is finished, the expectation stage can be compared to the situation of a mass in this universe, where the undertaking is to follow which planet the mass will settle upon, i.e., which class the new information point has a place with.

This planetary similitude assists with figuring out the fundamental tasks of the calculation, as it conveys the idea of spatial connections and the planning of information focuses to a characterized, though dynamic, structure. Nonetheless, the planets in our similitude are not powerful like certifiable planets, which could impact each other's development; all things being equal, they are fixed portrayals that guarantee a stable yet versatile characterization process. This static nature of the planets, while improving on the portrayal, likewise fits power even with commotion and anomalies in genuine world datasets.

The calculation's plan and execution lead to a few helpful properties, settling on it a convincing decision for different order undertakings. These properties include:

1) **Ability to gain from a couple samples:** This property permits the calculation to be profoundly effective in circumstances where marked information is scant or costly to get. Rather than requiring huge amounts of information to learn significant examples, the calculation can sum up well from generally little datasets, settling on it an amazing decision for spaces where information securing is restricted.
2) **Ability to weight the significance of preparing vectors:** In some certifiable situations, a few information focuses are more useful or delegate of the fundamental examples than others. This calculation consolidates an instrument that permits it to dole out various loads to preparing tests in light of their significance or significance. This adaptability upgrades the calculation's capacity to zero in on additional basic data of interest, further working on its prescient precision and proficiency.
3) **Resilience to overfitting:** Overfitting is a typical test in AI, where models become excessively custom fitted to the preparation information and neglect to sum up to new, concealed models. The plan of this calculation, particularly the proper idea of the planet-like bunches and the capacity to control the impact of individual examples, makes it innately impervious to overfitting. By keeping the model from turning out to be excessively complicated and excessively delicate to the preparation set, the calculation keeps a harmony between model effortlessness and forecast precision, upgrading its vigor.

These three properties on the whole structure the center qualities of the *Planetary Grouping Algorithm*, making it reasonable for a great many applications, particularly in situations where information quality or amount is restricted, or when high interpretability and versatility are basic.

One especially huge part of this calculation is its help for the thought of models, as conceptualized by Eleanor Rosch in her work on classification. As per Rosch [1](P. 41), models are glorified portrayals of classes, where certain individuals from a classification are more focal or normal than others. The planetary model normally obliges this thought, as the actual

planets can be seen as models of the particular classes. The mass or data of interest in this universe should be visible as drifting portrayals that are either drawn to or lined up with these focal models, guaranteeing that grouping isn't just about retaining data of interest, however about perceiving the basic designs they address. This capacity to zero in on models permits the calculation to successfully deal with genuine world datasets that frequently contain uproarious, uncertain, or anomaly data of interest, without allowing them to rule the educational experience.

In the accompanying segments, we will dive further into the numerical establishments, the preparation and expectation processes, and the different uses of the *Planetary Characterization Algorithm*. This investigation will feature its assets, limits, and potential for future enhancements, guaranteeing that the calculation stays pertinent and valuable in various spaces, from medical care to fund, where effective, versatile, and interpretable order models are sought after.

II. Definition

To officially present the parts of the *Planetary Characterization Algorithm*, we should initially lay out the numerical meanings of the emblematic designs that are key to its working. These designs incorporate the planet, the universe, and other key constants that will direct the learning and forecast processes.

The essential structure block in our model is the planet, which can be considered a bunch or gathering in the dataset. Every planet fills in as a portrayal of a class in the information, and described by a few key credits characterize its properties inside the calculation's structure. These qualities include:

- **Mass** m: The mass of a planet is a unique amount that addresses the weight or impact that the planet applies on the preparation interaction. It is a genuine number, and it changes after some time as the calculation repeats and gains from the information.
- **Radius** r: The sweep of the planet decides the size of the planet's impact inside the space of the universe. This amount is likewise unique, as the planet's impact might extend or contract in view of the dispersion of the preparation information and the educational experience.
- **Position** $\overrightarrow{x}$: The place of the planet is a vector in a n-layered space. This addresses the area of the planet in the component space of the dataset. As the calculation learns, the place of the planet is refreshed to more readily address the class it is displaying.
- **Class** θ: The class θ is a static property of the planet that stays unaltered all through the preparation cycle. Every planet addresses a particular class, and the actual class is a whole number that distinguishes which class the planet compares to in the dataset.

Numerically, we characterize a planet as a tuple containing the mass, span, position, and class:

$$\begin{aligned} &mand \in and\mathbb{R} \\ &rand \in and\mathbb{R} \\ &\overrightarrow{x}and \in and\mathbb{R}^n \\ &\theta and \in and\mathbb{Z} \end{aligned} \tag{1}$$

$$\mathbb{P}and = and\{m, r, \overrightarrow{x}, \theta\}$$

In this situation, $\mathbb{P}$ addresses a planet, and every one of the properties is meant by an image: m for mass, r for span, $\overrightarrow{x}$ for position in highlight space, and θ for class.

A. The Universe

The universe in this setting is basically an assortment of planets, where every planet compares to a bunch in the dataset. The universe all in all addresses the whole model that the calculation works during the preparation stage. The universe is the space wherein the planets are arranged, and every planet applies impact over this space in light of its properties. The universe isn't static; as the calculation learns and new planets are presented or existing ones are adjusted, the construction of the universe develops. The universe is addressed numerically as a bunch of planets:

$$\mathcal{U} = \{\mathbb{P}_{\not\in}, \mathbb{P}_{\not\in}, \ldots, \mathbb{P}_{\neg}\}$$

where $\mathbb{P}_{\not\in}, \mathbb{P}_{\not\in}, \ldots, \mathbb{P}_{\neg}$ are the singular planets in the universe, each with its own mass, span, position, and class.

B. Global Constants

There are a few worldwide constants that are indispensable to the calculation's activity. These constants are utilized to control different parts of the calculation, like the instatement of planets, the development of masses inside the universe, and the quantity of emphasess the calculation will go through during preparing.

- **Initial Radius** r': When another planet is made, it is doled out an underlying sweep r'. This range addresses the beginning size of the planet's impact known to man. The underlying span is commonly a proper worth or determined in light of the conveyance of the preparation information. Over the long run, as the calculation advances, the range of the planet might be acclimated to more readily mirror the class it is demonstrating.
- **Percent Step** α: The percent step α characterizes how much a test mass moves inside the universe prior to recalculating the powers following up on it. It is a pivotal boundary for controlling the elements of the calculation. The worth of α impacts how rapidly the calculation combines to an answer, as it directs the granularity with which the mass updates its situation during expectation. A bigger α may prompt quicker development yet less accuracy, while a more modest α may dial back the calculation yet increment the exactness of the expectation.
- **Number of Iterations** β: The quantity of emphasess β addresses the complete number of steps the calculation

will take during the preparation cycle. Every emphasis permits the planets to refresh their properties in light of the present status of the universe and the impact of the test masses. The quantity of cycles is a basic boundary that decides the preparation length and the last precision of the model.

- **Distance Function** $D(\overrightarrow{x}, \overrightarrow{y})$: The capability $D(\overrightarrow{x}, \overrightarrow{y})$ is utilized to ascertain the distance between two focuses in the element space, $\overrightarrow{x}$ and $\overrightarrow{y}$. This distance is significant for deciding how planets communicate with one another and how test masses are alloted to planets during the expectation stage. The distance capability can be any suitable measurement, for example, Euclidean distance or Mahalanobis distance, contingent upon the particular prerequisites of the application.

Numerically, these worldwide constants can be addressed as:

$$r' \in \mathbb{R}, \quad \alpha \in \mathbb{R}, \quad \beta \in \mathbb{Z}, \quad D(\overrightarrow{x}, \overrightarrow{y}) \in \mathbb{R}$$

where r' is the underlying range, α is the percent step, β is the quantity of cycles, and $D(\overrightarrow{x}, \overrightarrow{y})$ is the distance capability.

C. Summary of Definitions

To sum up, we have the accompanying key definitions:

- A planet $\mathbb{P}$ is characterized as a tuple containing the mass m, span r, position $\overrightarrow{x}$, and class θ.
- The universe $\mathcal{U}$ is the arrangement, everything being equal, which together structure the order model.
- Worldwide constants, for example, the underlying range r', percent step α, number of cycles β, and distance capability $D(\overrightarrow{x}, \overrightarrow{y})$ guide the calculation's preparation and expectation processes.

These definitions give the establishment to understanding the calculation's activity, which we will additionally investigate in the ensuing areas.

III. Training Model

The model assesses part vectors utilizing $h = \{\overrightarrow{x}, m, \theta\}$, where m addresses the vector's worth. The preparation pseudo-code is:

nearplanets ← Track down Planets long of $\overrightarrow{h}_x$;
nearplanets ← nearplanets where $P_\theta = h_\theta$;
if *nearplanets is Empty* **then**
  Universe Add Planet
  $\{m = h_m, r = r', \overrightarrow{x} = \overrightarrow{h}_x, \theta = h_\theta\}$
else
  p ← planet with most elevated force in nearplanets;
  Universe update p ←
  $\{m = p_m + h_m, r = m\frac{p_r}{p_m}, \overrightarrow{x} = \frac{p_m}{m}\overrightarrow{p}_x + \frac{h_m}{m}\overrightarrow{h}_x\}$
end

Algorithm 1: Training Algorithm

A. Asymptotic Analysis

The intricacy of assessing all planets is:

$$\mathcal{O}(D \cdot N) = \mathcal{O}(N)$$

Utilizing a KD-Tree:

$$\mathcal{O}(D \cdot \log N) = \mathcal{O}(\log N)$$

For adding a planet:

$$\mathcal{O}(D \cdot N + N_{\text{near}}) = \mathcal{O}(N + N_{\text{near}})$$

For KD-Tree:

$$\mathcal{O}(D \cdot \log N + N_{\text{near}}) = \mathcal{O}(\log N + N_{\text{near}})$$

Subsequently, the intricacy for adding a vector is:

$$\mathcal{O}(N_s(\log N + N_{\text{near}}))$$

B. Comparison of Arranging Times

N_{near}: Number of neighboring planets N_s: Number of tests

IV. Simulation Testing Model

Metaphorically, predicting the class of another point is tantamount to dropping a piece of mass into the universe and following the mass until it collides with a planet. In this moral story, we acknowledge that the planets are unfathomably little, and likewise there will be no impedance between them. Our test point, which tends to the new mass being dropped into the universe, will basically be described as $l = \{\overrightarrow{x}\}$, where $\overrightarrow{x}$ is the spot of the test point.

Allow us first to characterize how to get the standardized directional power vector. Review from material science that the gravitational power between two bodies is given by:

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

where $\mathbb{G}$ is the gravitational steady, m_1 and m_2 are the majority of the two bodies, and r is the distance between them. For our situation, we expect that the mass of each test point is equivalent to each and every other test point, so we can dismiss the mass as a consider the power estimation. Also, since the gravitational consistent $\mathbb{G}$ is unimportant with the end goal of power correlation, we can dispose of it from the situation. Subsequently, the half and half power condition per planet p becomes:

$$F = \frac{p_m}{r^2} \tag{3}$$

where p_m addresses the mass of planet p, and r is the distance between the planet and the test point l, which can be determined as $D(\overrightarrow{p}_x, \overrightarrow{l}_x)$, where $\overrightarrow{p}_x$ is the place of planet p and $\overrightarrow{l}_x$ is the place of the test point.

Then, we characterize the all out standardized force on our test mass utilizing the accompanying custom condition. The

TABLE I
COMPARISON OF CLUSTERING AND CLASSIFICATION TECHNIQUES

	Gravitational Clustering	**K-Means**	**SVM**	**Decision Trees**
Big O	O $(N_s(\log N + N_n)$	$O(n^{Dk+1} \log n)$	$O(n^3)$	$O(n_s D \log(n_s))$
Online Training	Yes	Yes	No	Partial
Variant Importance	Yes	No	No	No

net power, F_{net}, is the amount of the powers applied by every one of the planets in the universe on the test mass:

$$F_{net}and = and \sum_{p \in Universe} \frac{p_m(\overrightarrow{p}_x - \overrightarrow{l}_x)}{r^2} \\ F_{norm}and = and \frac{\alpha}{||F_{net}||} F_{net} \quad (4)$$

In this situation, F_{net} addresses the vector amount of the powers applied by all planets, and F_{norm} is the standardized adaptation of F_{net}. The consistent α is a stage size factor that decides the level of development for each update regarding the power applied. $||F_{net}||$ addresses the extent of the net power vector.

Presently, let us portray the reenactment calculation that utilizes these estimations to anticipate the class of the test point:

pos $\leftarrow$ $\overrightarrow{l}_x$;
for *i in I* $[0, \beta]$ *step 1* **do**
  force $\leftarrow \sum_{p \in Universe} \frac{p_m(\overrightarrow{p}_x - pos)}{r^2}$;
  standard $\leftarrow \frac{\alpha}{||force||} force$;
  pos $\leftarrow$ pos + standard
end
nearplanets $\leftarrow$ Track down Planets in Span of pos;
if *nearplanets isn't Empty* **then**
  return $\mathbf{mode}$[nearplanets θ]
else
  return [planet nearest to pos] θ
end

In this calculation: - The underlying position $\overrightarrow{l}_x$ of the test mass is set as the beginning stage. - In every cycle (up to β emphasess), the calculation ascertains the power applied on the test point by all planets, registers the standardized power vector, and updates the place of the test point in like manner. - When the position has been refreshed over the necessary number of cycles, the calculation checks for the closest planets to the refreshed position. - On the off chance that there are planets inside a particular span, the calculation returns the most successive class (method) of the closest planets. Assuming that no close by planets are found, the calculation returns the class of the planet nearest to the last position.

A. Asymptotic Examination of Reproduction Testing Model

Allow us now to play out the asymptotic examination of the reproduction testing model. In the reproduction, let N address the quantity of planets known to mankind, and let D address the dimensionality of the element vector for every planet. To register the power, we want to play out a few tasks for every planet. These activities include:

1. Deducting the place of the planet from the place of the test mass. 2. Duplicating the outcome by the mass of the planet. 3. Squaring the distance (i.e., processing the extent of the distance). 4. Partitioning by the square of the distance.

Every one of these tasks includes essential number juggling or vector activities, so the all out intricacy for computing the power applied by a solitary planet is $\mathcal{O}(D)$, where D is the dimensionality of the component vector. Since we really want to ascertain the power for all N planets, the absolute intricacy for computing the net power is:

$$\mathcal{O}(D \cdot N) = \mathcal{O}(N) \quad (5)$$

Furthermore, during every cycle, we really want to process the greatness of the net power vector $||F_{net}||$, which takes $\mathcal{O}(D)$ tasks, and afterward standardize the power by increasing by $\frac{\alpha}{||F_{net}||}$, which additionally takes $\mathcal{O}(D)$ activities. At last, refreshing the place of the test mass includes another $\mathcal{O}(D)$ activity. In this way, the absolute intricacy for one cycle is:

$$\mathcal{O}(D \cdot N + D) = \mathcal{O}(N) \quad (6)$$

Since we perform β cycles altogether, the absolute intricacy for the recreation becomes:

$$\mathcal{O}(D \cdot N \cdot \beta + N) = \mathcal{O}(N \cdot (D \cdot \beta + 1)) \quad (7)$$

At long last, finding the closest planets (in light of the refreshed position) takes $\mathcal{O}(N)$ tasks, since we really want to check all planets for their distance to the test point. Accordingly, the general time intricacy of the recreation testing model is:

$$\mathcal{O}(N \cdot (D \cdot \beta + 1)) \quad (8)$$

B. Comparison of Preparing and Testing Times

The preparation time intricacy (as we examined in the past area) was $\mathcal{O}(N_s \cdot (\log N + N_{\text{near}}))$, where N_s is the quantity of tests, N_{near} is the quantity of adjacent planets, and N is the quantity of planets. Contrasting this and the time intricacy of the reenactment testing model $\mathcal{O}(N \cdot (D \cdot \beta + 1))$, we can see that while the two models include the quantity of planets

N, the recreation model likewise relies upon the quantity of cycles β and the dimensionality of the component vectors D.

For situations where N is enormous, the preparation intricacy and testing intricacy can both scale directly with the quantity of planets. In any case, the presentation of dimensionality D and the cycle count β in the testing model presents extra factors that might expand the computational expense contrasted with preparing.

- N alludes to the quantity of planets (or preparing tests).
- D alludes to the dimensionality of the component vectors for every planet.
- β alludes to the quantity of emphasess performed during the reenactment.

Probabilistic Non-Simulating Model

In this section, we present an alternative approach to determining the class of a test point, which eliminates the need for simulation. Instead, we employ purely statistical methods. The core assumption of this model is that each planet or cluster is normally distributed around its center, with a standard deviation that is a function of the planet's radius, denoted as $\sigma(p_r)$. This assumption allows us to define the probability density function (PDF) for each cluster.

Probability Density Function (PDF)

The probability density function of a planet p with respect to the test point $\overrightarrow{l}_x$ is given by:

$$\mathbf{PDF_p} = \frac{1}{2\pi * \sigma(p_r)} \exp\left(\frac{-D(\overrightarrow{p}_x, \overrightarrow{l}_x)^2}{2\sigma(p_r)^2}\right)$$

Where:

- $D(\overrightarrow{p}_x, \overrightarrow{l}_x)$ represents the Euclidean distance between the planet p and the test point $\overrightarrow{l}_x$,
- $\sigma(p_r)$ is the standard deviation of the distribution for each planet, and
- π is the mathematical constant pi.

This formula captures the likelihood that the test point $\overrightarrow{l}_x$ belongs to a particular cluster, based on the Euclidean distance between the test point and the center of the cluster. The standard deviation $\sigma(p_r)$ controls the spread of the distribution, thus influencing the strength of the gravitational pull exerted by each planet.

Prediction Equation

Given the probability density function for each planet, the prediction for the class of the test point is computed by maximizing the product of the PDFs of all planets in the universe that belong to the same class. Specifically, for each class θ_n, we compute the product of the PDFs for all planets p in the universe that belong to class θ_n:

$$\mathbf{MAX}_\theta \left[\prod_{p \mid p_\theta = \theta_n}^{Universe} \frac{1}{2\pi * \sigma(p_r)} \exp\left(\frac{-D(\overrightarrow{p}_x, \overrightarrow{l}_x)^2}{2\sigma(p_r)^2}\right) \right]$$

This equation essentially evaluates the likelihood of the test point $\overrightarrow{l}_x$ belonging to each class based on the planets' distributions. It calculates the product of the probabilities across all planets in the class, and the class with the highest likelihood is selected as the predicted class.

Class Size Adjustment

In order to account for the fact that different classes may have varying numbers of planets, we modify the above equation to normalize the probabilities by the number of planets per class. This ensures that the algorithm is not biased towards classes with more planets. The adjusted prediction equation becomes:

$$\mathbf{MAX}_\theta \left[\frac{\log\left(\prod_{p \mid p_\theta = \theta_n}^{Universe} \exp\left(\frac{-D(\overrightarrow{p}_x, \overrightarrow{l}_x)^2}{2\sigma(p_r)^2}\right)\right)}{|p \mid p_\theta = \theta_n|} \right]$$

Where:

- The logarithm is used to simplify the computation of large products,
- The denominator $|p \mid p_\theta = \theta_n|$ is the number of planets in class θ_n, ensuring that the model adjusts for the class size.

The mass term p_m is incorporated into the equation to weigh larger planets more heavily in the likelihood calculation. This term ensures that planets with larger masses (which can be viewed as having more influence) contribute more to the class prediction.

Choice of Standard Deviation Function

Through empirical experimentation, we found that the best function for the standard deviation $\sigma(p_r)$ is simply p_r^2, where p_r is the radius of the planet. This choice simplifies the model and has been shown to provide robust performance in various clustering scenarios.

Asymptotic Analysis

The computational complexity of this probabilistic model can be analyzed by examining the operations involved in calculating the probability for each planet and class. Since each planet requires the computation of a Euclidean distance and the evaluation of an exponential function, the total time complexity for the model is given by:

$$\mathrm{O}\left(DN\right) = \mathrm{O}\left(N\right)$$

Where D is the dimensionality of the feature vector and N is the number of planets in the universe. This indicates that the time complexity is linear in the number of planets, making this model efficient for large datasets.

V. Testing Results

We evaluated the performance of the gravitational clustering algorithms using the well-known Wisconsin Breast Cancer dataset, which has been widely used for testing classification algorithms. The results are presented in the table below, where we compare the accuracy of the simulated and probabilistic models under two different parameter configurations: $r' = 50, \alpha = 0.01, \beta = 100$ and $r' = 5000, \alpha = 0.001, \beta = 1000$.

Gravitational Clustering	$r' = 50$ $\alpha = 0.01$ $\beta = 100$	$r' = 5000$ $\alpha = 0.001$ $\beta = 1000$
Simulated Model	89.65%	90.59%
Probabilistic Model	92.78%	72.41%

A. Observations

The results indicate that the probabilistic model performs well when the clusters are small and well-defined. However, as the size of the clusters increases, the performance of the probabilistic model decreases. This is likely due to the fact that larger clusters tend to have more spread-out distributions, which can reduce the accuracy of the model. On the other hand, the simulated model, which relies on gravitational forces, consistently achieves higher accuracy for larger clusters.

VI. Comparison with Popular Algorithms

We also compared the performance of our gravitational clustering models with other popular classification algorithms implemented in the scikit-learn library. The datasets we used include the Iris dataset, the Digits dataset, and the Olivetti Faces dataset. The results are summarized in the table below.

		Algorithm			
Data-sets	GC Prob	GC Sim	SVM (poly)	SVM (rbf)	Naive Bayes (Gaussian)
Iris	98.41%	96.82%	94.66%	97.33%	96%
Digits	86.95%	91.04%	98.99%	25.61%	83.85%
Olivetti	65.5%	77.5%	7.5%	8.5%	99.5%

VII. Few-Shot Learning

In an additional experiment, we tested the ability of our algorithm to handle very few samples. We trained the models using only one sample per class for each dataset. The results are shown in the table below, demonstrating that the probabilistic and simulated models can still achieve relatively high accuracy even with minimal training data.

	Accuracy Per Data-set	
Algorithm Type	GC Prob	GC Sim
Iris	93.33%	92.4%
Digits	72.88%	71.11%
Olivetti	50%	61.25%

A. Conclusion

The gravitational clustering models, both simulated and probabilistic, show promise as robust alternatives to traditional clustering algorithms, particularly in applications such as few-shot learning where other algorithms struggle. Further optimization of the model's parameters and refinement of the gravitational force assumptions could improve their performance in larger, more complex datasets.

References

[1] G. Lakoff, *Women, Fire, and Dangerous Things*, T. U. of Chicago Press, Ed. The University of Chicago Press, 1987.