February 2019

ML Project: Exoplanets Classification

Ferrario Marco and Ottolina Giorgio

Planetary habitability has become a very important topic during the last years, especially thanks to technological advancements and the latest space discoveries. We now know that there are way more planets around us than what we thought was possible. We are at an unique juncture of human history where within next few years we may be able to confirm earth like planets or earths twin around other stars where conditions are suitable for life to exists or even find a proof of life on those planets. NASA launched Kepler space telescope in 2009 which looked at a patch of sky and studied over 150,000 stars and found around 2500 confirmed planets around other stars using transit method. Scientists and researchers have identified several of these planets which could be suitable for life based on planet and parent star's features. Based on Kepler data, scientists have estimated that there could be as much as 40 billion earth size planet in our own Milky Way galaxy alone. In early 90s, scientists started discovering planets around other stars, called exoplanets. Kepler space telescope was launched by NASA in 2009 to find exoplanets in our stellar neighborhood. Kepler finds exoplanets by looking for tiny dips in the brightness of a star when a planet orbiting it crosses in front of it. Once detected, the planet's orbital size can be calculated from the period (how long it takes for the planet to orbit once around the star) and the mass of the star using Kepler's Third Law of planetary motion. The main objective of our analysis is to analyze the exoplanets data in order to identify which ones were really confirmed as being planets and which ones were not. To do so, we applied Classification Machine Learning algorithms. Further work that could be done from here would be to define which among these confirmed planets are effectively habitable.

CONTENTS

I. INTRODUCTION

The dataset we have been provided with consists of a list of potential exoplanets with several attributes. The selected dataset contains **3584 rows and 25 columns**. Characteristics of all detected and discovered exoplanets orbiting outside of the solar system since 1992. Data fields include planet and host star attributes, discovery methods, and (of course) date of discovery.

A. Dataset Attributes

We removed the other attributes for specific reasons, which we will describe in detail in the next related sections. The following list consists only of the attributes that were used for the preliminary analysis.

- **PlanetaryMassJpt** = exoplanet's mass in relation to Jupyter's
- **RadiusJpt** = exoplanet's radius in relation to Jupyter's
- **PeriodDays** = exoplanet's orbital period, that is the time it takes to complete one orbit around another object
- **SemiMajorAxisAU** = exoplanet's major axis
- **Eccentricity** = parameter that determines the amount by which its orbit around another body deviates from a perfect circle
- **SurfaceTempK** = superficial planetary equilibrium temperature in Kelvin
- **DistFromSunParsec** = distance from the Sun in Parsec
- **HostStarMassSlrMass** = mass of the host star in relation to the Sun
- **HostStarRadiusSlrRad** = radius of the host star in relation to the Sun
- **HostStarMetallicity** = metallicity of the host star in relation to the Sun
- **HostStarTempK** = temperature of the host star in Kelvin

- **ListsPlanetIsOn** = A list of lists the planet is on including "Confirmed planets", "Controversial", "Kepler Objects of Interest", etc.

For further informations, refer to Kaggle platform[1]. The thing about this dataset that immediately caught our attention was its huge number of missing values. Besides, in the dataset there are a lot of planets belonging to Solar system even if that is apparently a contradiction since "exoplanet" literally means "planet orbiting around a star different than the Sun".

II. DATA PREPROCESSING

A. Feature Selection

In the first step, with a column filter all columns were kept except for "PlanetIdentifier", "TypeFlag", "DiscoveryMethod", "DiscoveryYear", "LastUpdated", "RightAscension" and "Declination", since we thought they wouldn't help us finding a solution to the classification problem. Then, we removed the attributes which presented more than 80% missing values. The presence of all these missing values is due to the big difficulty that arises from studying characteristics of planets which orbit at thousands of light years from Earth. Besides, one attribute often happens to be linked to another one: for example, mass is useful to define the orbit[2], the temperature derives from the distance from the relative star. We can see how a missing attribute value highly influences the possible absence of another one. Finally, rows whose values were missing for more than half of the total. At this point, in order to handle the remaining missing values, which were still a lot, we searched for possible relations between attributes. We found that there was some kind of relation between PlanetaryMassJpt and DistFromSunParsec[3]. For this reason we decided to apply the most probable method to impute missing values, removing in the first place records that presented missing values in both columns. Missing values of the remaining columns were then removed using the method of Linear Interpolation. Linear interpolation is a form of interpolation, which involves the generation of new values based on an existing set of values. This is obtained by geometrically rendering a straight line between two adjacent points on a graph or plane. All points on the line other than the original two can be considered interpolated values.

B. Preparing the Dataset for Classification

The ListPlanetisOn was chosen as the column for the Classification Models; anyway, it was originally made of more than two categories. Since the model we decided to apply was based on classifying which exoplanets were actually confirmed as being so, all class' values were aggregated in order to make it binary. Inside two String Manipulation nodes, two different regex were used to group all similar Exoplanets into two categories. It was mandatory to classify, respectively:

- All the rows with the pattern **Confirmed Planets** or the value **Solar System** (since objects belonging to the Solar System are surely considered as confirmed exoplanets) as **Confirmed Planets**
- All the rows with the pattern **Controversial**, value **Kepler Object of Interest** (since objects belonging to this classification system are surely considered as unconfirmed) and/or **Retracted** as **Unconfirmed**.

We immediately saw that the Unconfirmed class consisted of way less instances than the Confirmed one and was therefore the minority class: only 4% of instances were Unconfirmed ones. In order to improve models' predictions and to prevent overfitting and underfitting, we decided to:

- adopt the two **classification algorithms** that gave the best results;
- apply further **feature selections**;
- use **cross validation** approach to partition the dataset.

III. CLASSIFICATION

A. Models

Our first step was deciding which Machine Learning Algorithms were the most appropriate given our data and our objective. For the Classification the following algorithms were chosen:

- **Support Vector Machine with SMO**
- **Naive Bayes**
- **Random Forest**
- **Decision Tree J48**
- **BayesNet**
- **Multi Layer Perceptron**

Since we considered the "Unconfirmed" class as the positive one (the one with smaller frequence), we had to pay significant attention to the Recall value. Trying to evaluate the habitability/non-habitability of a planet it's especially relevant to identify the exoplanets that really don't classify as that in order to remove them from further analysis without any doubt.

B. Class Imbalance Problem

When utilising orthodox machine learning algorithms the mining of skewed datasets can result in models that are strongly predictive for the larger class, while delivering performance which is poorly predictive for the minority class. This is due to the fact that orthodox classifiers will attempt to return the most correct predictions based

upon the entire dataset, this results in them categorizing all data as belonging to the larger class. This class is usually the class which is of least interest to the data-mining problem.

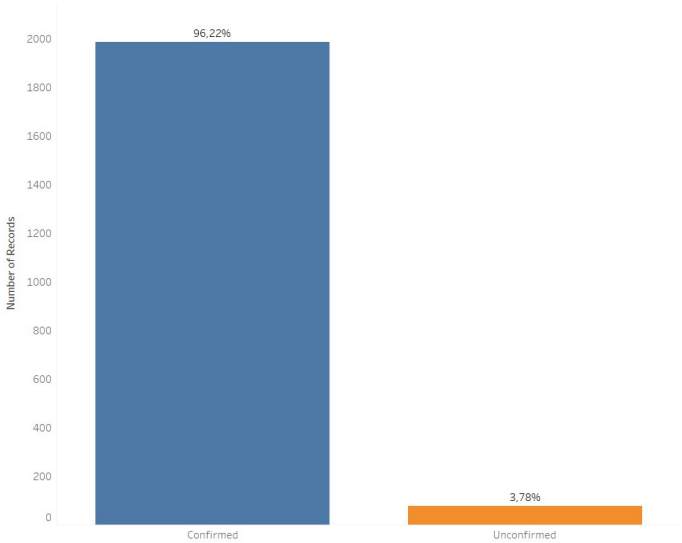

FIG. 1. Confirmed and Unconfirmed Classes - The Imbalance Problem

There are 5 type of problems that occur when learning from skewed datasets:

- **Incorrect evaluation measures**. One must be very careful when selecting the correct performance measure to evaluate a learner (**F-measure, G-mean or AUC values**);
- **Absolute Scarcity of data**. The number of items that belong to the minority class are small when compared to the total number of samples, this can make it difficult to observe patterns within the minority class;
- **Data fragmentation**. This can be a problem as most classifiers utilize a divide and conquer approach. This causes a continual division of the learning space. This results in patterns that are found in the data as a whole cannot be found in the resulting partitions formed by this divide and conquer strategy;
- **Inappropriate inductive bias**;
- **Noise**. While effecting the performance of a data mining system as a whole, noise has been found to have a greater effect on the minority classes

Because of the way the class was imbalanced, the possibility of using sampling techniques was considered. **Equal Size Sampling** (technique that removes records from the majority class in order to make it even with the minority one) didn't look like a good choice, since it would have reduced dataset's instances in a significant way ("Unconfirmed" value was found only 78 times). We considered to use the **SMOTE** technique (Synthetic Minority Oversampling Technique). It generates "synthetic" observations: starting from a minory class point, an arbitrary number of nearest neighbours are selected and connected to before mentioned point. The new observations are points on these segments, created by a randomic interpolation.

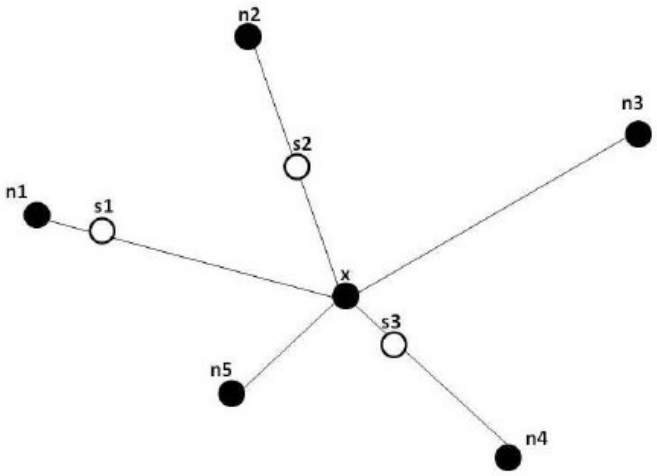

FIG. 2. Graphical representation of a SMOTE Model

C. Results and Performance Evaluation

In order not to alterate the testing set validity, the SMOTE node was applied only to the training set. For each one of the models we decided to create two different kind of workflows in order to evaluate the efficiency of such an approach. In the lower section of the KNIME project are located the workflows that don't include oversampling of the minority class through SMOTE, while in the upper section there are the workflows which include SMOTE nodes. The dataset was divided in both cases in two parts, according to the **Holdout Method**. 67% of the dataset was used for training and the remaining 33% for the test set with stratified sampling. We decided to proceed in this way to avoid further imbalancing of the dataset, which could have been caused by random sampling. Since we are dealing with an imbalanced classification class, we have to remember that accuracy is not the best metric to look at in order to decide which model is the best performing among all. In situations like this, other metrics such as F-measure are definitely a better call.

D. SMOTE models Results

SMOTE model with 5 nearest neighbors was applied to the training set. Below we can see an overview of the results we obtained:

We can see how the model with the highest accuracy among all is Random Forest, while the one with the highest value of AUC is Bayes Net.

Name	Accuracy	AUC
SMO	0.83	0.71
Naive Bayes	0.92	0.81
Random Forest	0.95	0.79
Decision Tree J48	0.9	0.72
Bayes Net	0.83	0.83
Multi Layer Perceptron	0.84	0.78

Name	Recall	Precision	F-measure
SMO	0.58	0.13	0.21
Naive Bayes	0.15	0.11	0.13
Random Forest	0.42	0.41	0.42
Decision Tree J48	0.46	0.19	0.27
Bayes Net	0.61	0.13	0.21
Multi Layer Perceptron	0.20	0.05	0.08

The model which correctly predicts the positive observations is Bayes Net, while the one with the best F-measure value (that is equal to the harmonic average of Precision and Recall) is Random Forest.

E. No-SMOTE models Results

Below we can see an overview of the new results we obtained:

Name	Accuracy	AUC
SMO	0.96	0.5
Naive Bayes	0.93	0.81
Random Forest	0.96	0.79
Decision Tree J48	0.96	0.64
Bayes Net	0.94	0.80
Multi Layer Perceptron	0.96	0.77

We can observe that the AUC values for the various models are almost the same as the ones of the SMOTE models. It is very interesting to observe how the SMO method didn't classify any True Positive value for the **Unconfirmed** class, obtaining a 0 value for the Recall parameter and making then impossible to evaluate the Precision. In both analyses the ROC curves appeared **jagged** for the majority of the models: this underlines the dependence on the details of the particular subset of data. In order to reduce this dependence we then used cross validation.

Name	Recall	Precision	F-measure
SMO	0	Nan	Nan
Naive Bayes	0.15	0.49	0.15
Random Forest	0.19	0.5	0.28
Decision Tree J48	0.19	0.39	0.26
Bayes Net	0.42	0.28	0.33
Multi Layer Perceptron	0.12	0.43	0.18

In both analyses, the obtained results revealed themselves to not be particularly useful. We could very easily find a distortion due to the sampling done in order to divide the observations in the two subgroups. For this reason, we decided to apply a different approach regarding to the two classifiers which produced the best results: Bayes Net and Random Forest.

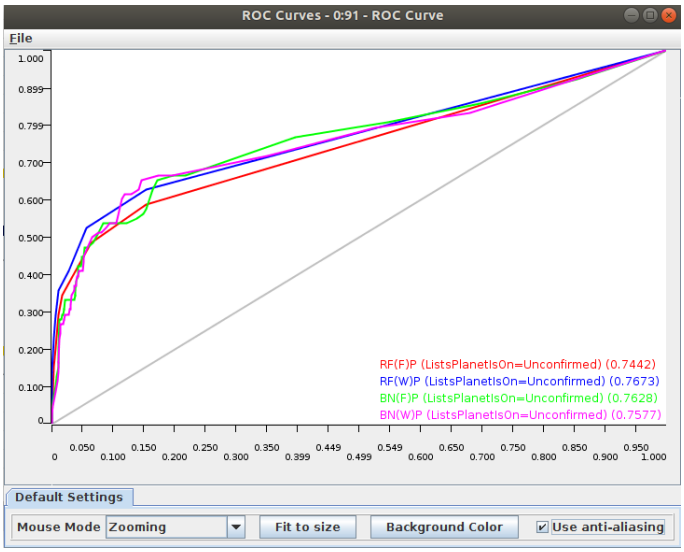

FIG. 3. ROC curve that highlihts models' performance.

IV. FEATURE SELECTION

Here we adopted two different techniques: **Filter method** and **Wrapper method**. As far as the Filter method is concerned, we compared uni variate and multi variate applied to Random Forest and Bayes Network algorithms. **In the first case scenario**, uni variate version reached higher F-Score values, by selecting the attributes: 1) HostStarTempK; 2) PlanetaryMassJpt; 3) DistFromSunParsec; 4) HostStarMassSlrMass; 5) HostStarMetallicity; 6) HostStarRadiusSlrRad.

In the second case scenario (filter with Bayes Net), the multi variate version has been more accurate and offered higher F-Score values by selecting the attributes: 1) DistFromSunParsec; 2) HostStarMetallicity; 3) HostStarTempK; 4) PlanetaryMassJpt.

Wrapper method: this method selected different attributes for each one of the two main related models. Random Forest: 1) SemiMajorAxisAU; 2) SurfaceTempK; 3) DistFromSunParsec; 4) HostStarMassSlrMass; 5) HostStarTempK; 6) PlanetaryMassJper.

Bayes Network: 1) DistFromSunParsec; 2) HostStarMassSlrMass; 3) HostStarMetallicity; 4) PlanetaryMassJpt.

The following box plots graphically represent F-Measure and Recall results obtained by using the Filter and the Wrapper method both for Random Forest and Bayes Net methods.

V. CROSS VALIDATION

By applying Cross Validation, two different classification models were tested with the attributes selected by the Filter (uni variate Random Forest and multi variate Bayes Network) and the Wrapper methods in comparison. **K** was chosen as a parameter with value equal to 5 after some attempts. SMOTE sampling on the contrary wasn't effective in this case scenario, so we decided not to implement it. Analyzing the results obtained for

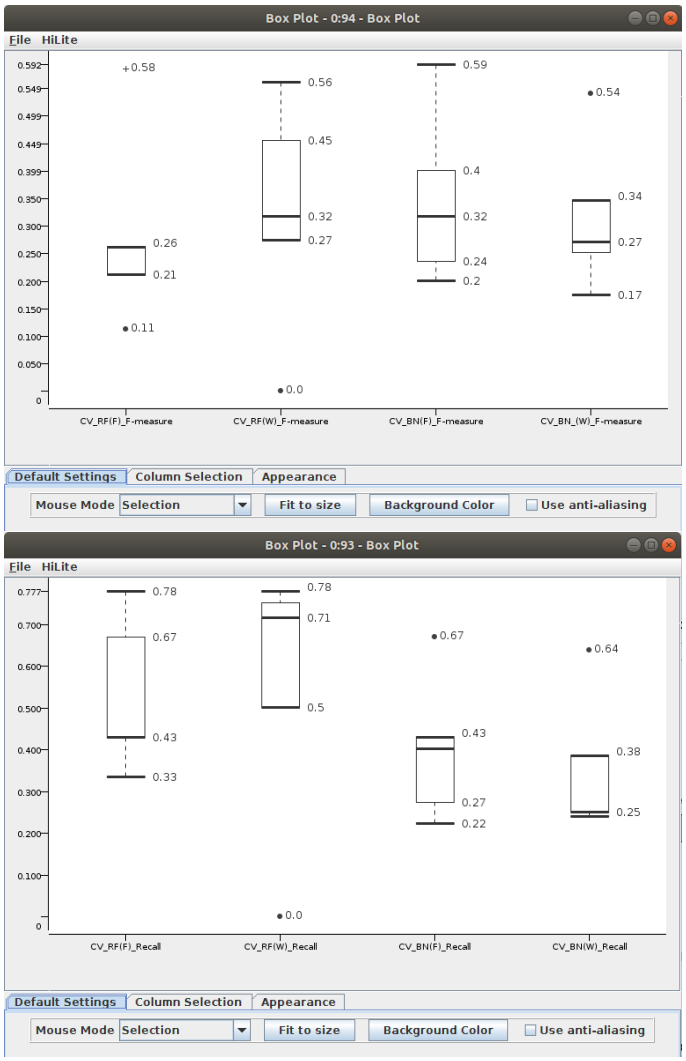

FIG. 4. F-Measure (first picture) and Recall (second picture) values for cross validation-random forest and cross validation-bayes net using Filter and Wrapper methods.

the 5 folds we can observe how Recall has in some cases scored higher than 75% for Random Forest model, while F-Measure has almost reached 60% for a fold with Bayes Net model. The ROC curves (for the "Unconfirmed class) appear less "jagged" and this underlines how models are correct in identifying 60% values of True Positive (AUC value around 0.75), with Wrapper methods models for attributes selection being slightly preferable.

VI. RECAP AND POSSIBLE DEVELOPMENTS

Our work's goal was to classify which potential exoplanets identified during the past years were surely confirmed as being so or not. The main obastacles we faced during the analysis have been imbalanced class and the high number of missing values. By applying cross validation and feature selection techniques we reached the following results:

- **Average Recall** for the 5 folds resulted higher by using Random Forest as classification algorithm: with Wrapper method, it reached 55%;
- **Average F-Measure** resulted higher by using Bayes Network as classification algorithm: with multi variate filter method it reached 35%;
- **AUC values** remained almost the same as the ones encountered in previous experiments.

Because of the extremely low frequence of the Unconfirmed value, in these experiments the **Train/Test set schema** was adopted: it is ideal for the Filter method but not so much for the Wrapper one. A possible way to improve performances could then be to apply the Train/validation/test set schema. Besides, it is auspicable that in the future the data related to single exoplanets will be more complete and precise so that handling an enormous quantity of missing values won't be necessary anymore.

VII. REFERENCES

1 https://www.kaggle.com/mrisdal/open-exoplanet-catalogue

2 http://www.danielegasparri.com/Italiano/articoli/astrofisica/Massa.pdf

3 https://www.kaggle.com/nelnour123/hidden-reationships