Classification and discovery of Earth-alike exoplanets

Tommaso Farneti[1], Luca Spezia[1], Tariq Baghrous[1]

Abstract
The Kepler Space Telescope is a NASA-build satellite launched in 2009 dedicated to searching for exoplanets in star systems besides our own, with the ultimate goal of possibly finding other habitable planets.
By analyzing a dataset collecting the properties of approximately 10.000 exoplanet candidates the Kepler telescope has taken observations on, we first classify each exoplanet as ”candidate”, ”confirmed” or ”false positive” exoplanet and then, by analyzing the ”confirmed” ones, we perform a cluster analysis to try finding Earth-alike environments.
The classification was performed using four different models: Logistic, Random Forest, Naive-Bayes and SVM. The best one was the Random Forest with a F-measure of 0.86. The cluster analysis was performed using Hierarchical and K-means algorithm. The best clustering was obtained by the Hierarchical one, which divided the data into 7 groups and allowed us to partition those exoplanets which are more likely to have a rocky composition and maintain surface liquid water (cluster number 7).

Keywords
Machine Learning — Exoplanets — Classification — Cluster

[1] *Data Science, Università degli Studi di Milano Bicocca*

Contents

Introduction

The discovery of exoplanets, planets orbiting stars outside our solar system, has revolutionized our understanding of the cosmos and opened new avenues for investigating the possibility of habitable worlds beyond Earth. By measuring exoplanets' features, like diameters and masses, we can see compositions ranging from very rocky (like Earth and Venus) to very gas-rich (like Jupiter and Saturn). Some planets may be dominated by water or ice, while others are dominated by iron or carbon. The first exoplanets were discovered in the 1990s and since then we've identified thousands using a variety of detection methods[1]. However, one of the greatest challenges we face when observing new objects in space is determining whether they truly qualify as exoplanets. Therefore, observations are often categorized as ”confirmed” if they are indeed planets, labeled as ”candidate” if further evaluation is required, and designated as ”false positive” if an initial classification as a planet is proven to be incorrect.
The faster we are able to recognize if an observation is actually a planet, the more time we can spend trying to find new Earth-alike environments where life may exist or could have existed. Thus, in this project we first try to build a classifier which is able to predict whether a given observation is a ”candidate,” ”confirmed,” or ”false positive” planet (Research Question #1). Then, we aim to cluster the confirmed exoplanets to identify Earth-alike candidates within this group (Research Question #2).

1. Dataset

The dataset chosen for answering our research questions is the *Kepler Exoplanet Search Results*, which is a cumulative record of all observed Kepler ”objects of interest” — basically, all of the approximately 10,000 exoplanet candidates Kepler has taken observations on, available on the Kaggle Platform

[1] What is an Exoplanet - exoplanet.nasa.gov

[1]. For each planet are reported many features. The ones interesting for our project are:

- koi_disposition: CANDIDATE or FALSE POSITIVE or CONFIRMED;
- koi_fpflag_nt: a KOI[2] whose light curve is not consistent with that of a transiting planet.
- koi_fpflag_ss: a KOI that is observed to have a significant secondary event, transit shape, or out-of-eclipse variability, which indicates that the transit-like event is most likely caused by an eclipsing binary.
- koi_fpflag_co: the source of the signal is from a nearby star;
- koi_fpflag_ec: the KOI shares the same period and epoch as another object and is judged to be the result of flux contamination in the aperture or electronic crosstalk;
- koi_period: the interval between consecutive planetary transits (koi_period_err1 , koi_period_err2);
- koi_time0bk: Transit Epoch (time corresponding to the center of the first detected transit in BJD minus a constant offset of 2,454,833.0 days (koi_time0bk_err1 , koi_time0bk_err2);
- koi_impact: the sky-projected distance between the center of the stellar disc and the center of the planet disc at conjunction, normalized by the stellar radius (koi_impact_err1 , koi_impact_err2);
- koi_duration: the duration of the observed transits (koi_duration_err1 , koi_duration_err2);
- koi_depth: the fraction of stellar flux lost at the minimum of the planetary transit (koi_depth_err1 , koi_depth_err2);
- koi_prad: the radius of the planet (koi_prad_err1 , koi_prad_err2);
- koi_teq: approximation for the temperature of the planet;
- koi_insol: insolation flux is another way to give the equilibrium temperature (koi_insol_err1 , koi_insol_err2);
- koi_tce_deliv_name: TCE delivery name corresponding to the TCE data federated to the KOI;
- koi_steff: the photospheric temperature of the star (koi_steff_err1 , koi_steff_err2);
- koi_slogg: the base-10 logarithm of the acceleration due to gravity at the surface of the star (koi_slogg_err1 , koi_slogg_err2);
- koi_srad: the photospheric radius of the star (koi_srad_err1 , koi_srad_err2);
- ra: KIC Right Ascension;
- dec: KIC Declination;
- koi_kepmag: Kepler-band (mag).

[2]Kepler Object of Interest

For the Research Question #2, we enriched our dataset with three additional features found in the Nasa Exoplanet Archive [2]:

- koi_sma: Orbit Semi-Major Axis - half of the long axis of the ellipse defining a planet's orbit, and also the mean distance between exoplanet and his star;
- koi_smass: the mass of the star;
- koi_smet: the stellar metallicity.

For the classification task, we regard the *koi_disposition* as the dependent variable since our objective is to forecast the nature of a planet.

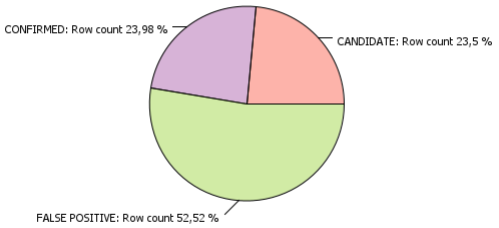

Figure 1. data objects by *koi_disposition* value

As we can see, majority of the observations are "false positive" (roughly 53%) and 23% are "candidate" exoplanets. All KOIs marked as "confirmed", which make around 24% of the observed objects, are also listed in the Exoplanet Archive Confirmed Planet table [2].

2. Preprocessing

2.1 Preprocessing (I Task)

Out of all the 50 features present in the dataset, we have discarded the ones that intuitively would have less importance in the classification task, such as *rowid, kepid, kepoi_name, kepler_name, koi_pdisposition* and *koi_score*, mainly because we have to classify without these information.
We have also removed *koi_teq_err1* and *koi_teq_err2* attributes since they have only *nan* values. The chosen variables are the one described in the previous section.

Then we treated the **missing values**.
We selected all the observations with *koi disposition* equal to "confirmed" and we treated the missing value as follow:

- missing string values replaced by most frequent string
- missing integer values replaced by most frequent value
- missing double values replace by mean value

The same process is repeated for *koi disposition* equal to "candidate" and "false positive".
In addition, we applied a binarization to the categorical variable *koi_tce_delivname*, which can assume 3 values. Specifically we created 3 dummy variables, each with one of the values the *koi_tce_delivname* can assume.

We tried performing an **additional features selection** based on a **J48** classifier (*koi_fpflag_nt, koi_fpflag_ss, koi_fpflag_co, koi_fpflag_ec, koi_prad_err1, koi_teq, koi_model_snr, koi_steff_err1, ra, delivname_q1_q17_dr24_tce*), but as we're going to see in the Results section this did not improve our classifiers.

2.2 Preprocessing (II Task)

In order to cluster the "confirmed" exoplanets to identify Earth-alike candidates we processed the data differently compared to what we did for the classification task. After joining the original dataset with the one coming from the Nasa Exoplanet Archive described in the Dataset section, we first treated the missing values as we did for task one. Obviously, we removed all the rows with *koi_disposition* different from "confirmed". Then we started computing new features using the ones present in the joined table, in order to capture more information which could lead us to the "habitability" decision of each exoplanet.

Computing new features. We computed the following two features:

- **surfaceGravity** - we multiply the koi_slogg by the square root of the stellar radius (koi_srad). This can give us an index that combines the effects of surface gravity and stellar radius:

$$\text{surfaceGravity} = \text{koi\_slogg} * \sqrt{\text{koi\_srad}}$$

- **insRelEarth** - we divide the koi_insol feature by the insolation flux value measured for Earth (1.361). This can give us a relative measure of the insolation received by the exoplanet compared to Earth:

$$\text{insRelEarth} = \frac{\text{koi\_insol}}{1.361}$$

The selected features for the cluster analysis are then: *surfaceGravity, insRelEarth, koi_sma, koi_smass, koi_smet, koi_srad, koi_steff, koi_teq, koi_prad*.
Lastly, we treated the outliers removing them.

3. *RQ* #1 - NonBinary Classification

3.1 Models

For the Research Question #1, several classification techniques have been implemented in order to identify the most suitable one based on the available data:

1. Regression Models: Logistic Regression.
2. Heuristic Models: Random Forest, a classifier that utilizes multiple decision trees and combines their results.
3. Probabilistic Models: Naïve Bayes, based on Bayes' theorem.
4. Separation Models: SVM (Support Vector Machine)

Before running each model we partitioned randomly the dataset into train and validation set. Then, we removed some rows from the train dataset such that the values in the koi_disposition column were equally distributed (**equal size sampling**). This can be useful, for instance if a learning algorithm is prone to unequal class distributions and you want to downsize the data set so that the class attributes occur equally often in the data set [3].

Each model was evaluated using **cross validation**. The dataset was split in ten subsets (10-fold cross validation) and the models were trained ten times, using, at each iteration, nine parts as training set and one part as test set. This was done in order to have a clearer understanding of the model performances and to better compare the performances of all models.

3.2 Results and Discussion

3.2.1 Performance evaluation

Since accuracy is not always the best choice when there is the need of evaluating the performance of a classifier, we decided to compute three different metrics:

- **recall**, also known as sensitivity, measures the proportion of actual positive instances that are correctly identified by the classifier:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **precision**, also known as positive predictive value, measures the proportion of predicted positive instances that are actually true positives:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **F-measure**, also called F1 score, is a combined measure of both recall and precision, providing a single metric to evaluate the classifier's performance:

$$\text{F} - \text{measure} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

In our case, which is a non-binary classification problem with three classes, we can calculate recall, precision, and F-measure for each class individually, and then compute the average of these metrics across all classes using different averaging strategies, such as Micro-average, Macro-average and Weighted-average. We decided to use the **Macro-average**, so we calculated the metrics for each class separately and then we averaged them, giving equal weight to each class.

Table 1. Performance metrics

Classifier	Recall	Precision	F-measure
Logistic	0.84	0.842	0.837
RandomForest	0.86	0.86	0.86
NaiveBayes	0.516	0.528	0.437
SVM	0.798	0.801	0.794

As we can notice from the Table 1, the Naive-Bayes classifier had poor performance in terms of correctly identifying positive instances (recall) and accurately predicting positive instances (precision), compared to the other classifiers, which in general are performing well on both metrics.
Taking into account both the recall and precision metrics simultaneously, we considered the F-measure to decide which model had the best performance. The best classifier was the Random Forest, with a F-measure of 0.86.

Classifiers after features selection. As we mentioned earlier in Section 2.1, we attempted additional feature selection and then ran two additional models to assess if we could achieve better performance. We employed a Logistic classifier and a Multilayer Perceptron model for this purpose. Unfortunately, the results we obtained were inferior compared to those achieved with the previous feature selection. Therefore, we made the decision not to pursue this approach any further.

3.2.2 ROC-curve

We compared our four models also by exploiting the AUC, which is the area under the ROC curve. Since we are performing a non-binary classification, we decided to use one of the many multiple approaches to adapt the ROC curve for evaluation. We used the *One-vs-All ROC*: we calculated multiple ROC curves, each considering one class as positive and the rest as negative. This way, we could evaluate the classifier's performance for each class individually [4].

As we can see in the three figures, the random forest model performs the best in classifying both the "candidate" and the "confirmed" categories. The SVM model demonstrates superior performance in classifying the "false positive" category. By analyzing the results of the Random Forest, it is possible to observe that the **most significant variables** for classifying whether an exoplanet can be labeled as "confirmed", "candidate" or "false positive" are: koi_fpflag_nt, koi_fpflag_ss, koi_fpflag_co, koi_fpflag_ec, koi_period, koi_prad, koi_model_snr and koi_steff_err1.

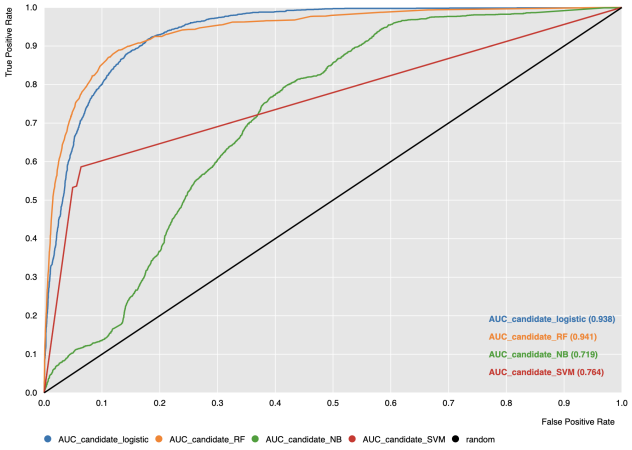

Figure 2. AUC, "candidate" koi_disposition

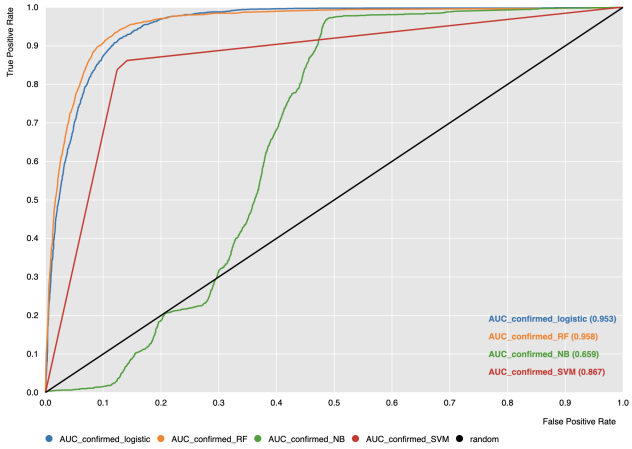

Figure 3. AUC, "confirmed" koi_disposition

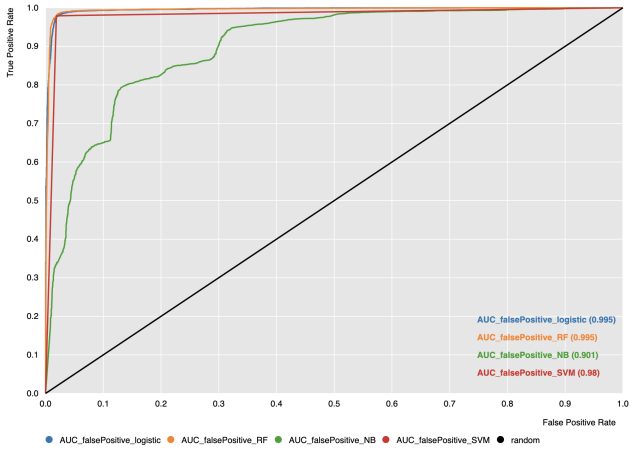

Figure 4. AUC, "false positive" koi_disposition

4. *RQ* #2 - Cluster Analysis

For the Research Question #2, after computing the new features as described in Section 2.2, we performed a cluster analysis to detect which exoplanets could be considered habitable or similar to the Earth.
After standardizing the data we proceed with the cluster analysis using two different aggregation methods: K-means and hierarchical.

4.1 Algorithms

4.1.1 Hierarchical clustering

It can be categorized into two main types:

- **agglomerative**: this method starts with each data point as an individual cluster and then progressively merges clusters based on their similarity, forming a hierarchical structure. At each step, the most similar clusters are merged until a single cluster containing all data points is formed.

- **divisive**: this method starts with a single cluster containing all data points and recursively divides it into smaller clusters based on dissimilarity. It repeatedly splits clusters into subclusters until each data point is assigned to its own individual cluster.

We decided to perform a hierarchical clustering with the **average linkage method**, which is a technique where we define the distance between two clusters to be the average distance between data points in the first cluster and data points in the second cluster. On the basis of this definition of distance between clusters, at each stage of the process we combine the two clusters that have the smallest average linkage distance [5].

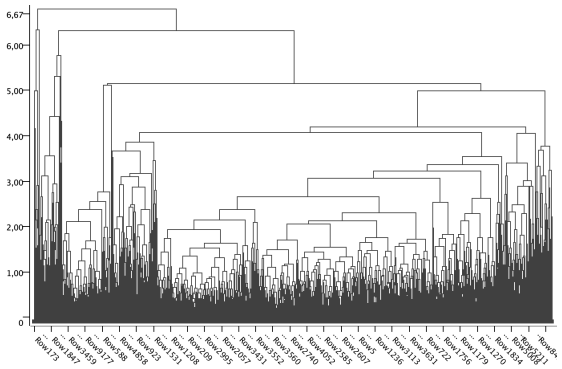

Figure 5. Dendrogram

The result is a dendrogram, which is a tree-like structure illustrating the hierarchical relationship between clusters. The height at which clusters merge in the dendrogram reflects the level of similarity between them.

For the hierarchical algorithm we decided the number of clusters equal to 7 by looking at the dendrogram, shown in Figure 5, which reports on the Y-axis the distance between clusters and on the X-axis the single observations.

4.1.2 K-means

K-means is a popular clustering algorithm used to partition a dataset into K distinct clusters. It aims to minimize the total within-cluster variance by assigning data points to the cluster whose mean is closest to them. First, the algorithm randomly select K data points as the initial cluster centroids. Then, it assigns each data point to the nearest centroid based on a distance metric, in this case the Euclidean distance. Then, it recalculates the centroids of the clusters by taking the mean of all data points assigned to each cluster. Finally, it repeats the assignment and update steps iteratively until convergence. Convergence occurs when there is minimal change in the assignment of data points or the positions of the centroids.
For the k-means algorithm we choose the optimal number of clusters equal to 6 by looking at the value of the Silhouette and Dunn indexes (4.2) for different number of clusters (Figure 6).

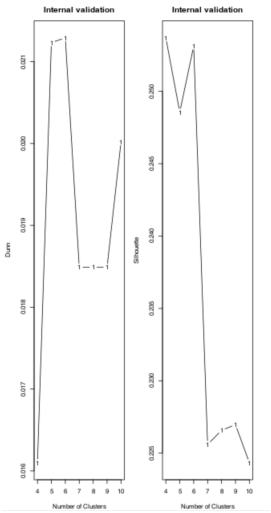

Figure 6. Silhouette and Dunn indexes for different clusters

4.2 Internal Validation

Since every algorithm identify clusters, even when there is no inherent structure in the data, it is necessary to evaluate the quality of the results. For this purpose, we considered two internal indices, which are evaluation metrics used to assess the quality and compactness of clusters without relying on external information or ground truth labels. The two internal indices that we used for clustering evaluation are the Silhouette index and the Dunn index.

4.2.1 Silhouette index

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). It can be used to study the separation distance between the resulting clusters [6]. For each data point, the Silhouette index considers the average distance to other data

points within its cluster (a) and the average distance to data points in the nearest neighboring cluster (b). The index value ranges from -1 to 1, where a higher value indicates a better clustering. A value close to 1 implies well-separated and compact clusters, while values close to 0 suggest overlapping clusters. Negative values indicate data points assigned to the wrong clusters.

$$s_i = \frac{b_i - a_i}{max(a_i, b_i)} \in [-1, 1]$$

where a_i represents the average distance between the i-th observation and other observations belonging to the same cluster; b_i represents the minimum value of the average distances between the i-th observation and observations belonging to different clusters than the one to which the i-th observation belongs.

4.2.2 Dunn index

The Dunn index is an internal evaluation scheme, where the result is based on the clustered data itself. It computes the ratio between the minimum inter-cluster distance and the maximum intra-cluster distance in the dataset. Higher the Dunn index value, better is the clustering. The number of clusters that maximizes Dunn index is taken as the optimal number of clusters k.

4.3 Validity Test

In order to verify the presence of an actual structure within the data, a test is conducted where the null hypothesis H0 states that the positions of observations in a region of an n-dimensional plane are equiprobable (Random Position Hypothesis).

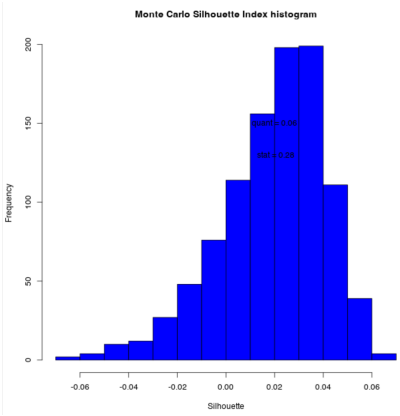

Figure 7. Empirical distribution of the Silhouette index obtained with the Monte Carlo method

The Silhouette coefficient obtained from the previously calculated Hierarchical algorithm is used to test H0. Specifically, by employing the Monte Carlo computational method, an empirical distribution is generated based on 1000 simulations. The quantile of this distribution, which depends on the chosen significance level ($\alpha = 0.01$), is compared to the test statistic value (i.e., the Silhouette coefficient).

Since the quantile (equal to 0.06) is significantly lower than the actual value of the Silhouette coefficient (0.285 - Table 2), we reject the null hypothesis of no structure within the data and conclude that the proposed clustering solution is meaningful.

4.4 Results and Discussion

The results are summarized in the Table 2.

Table 2. Internal Validation

Method	Silhouette	Dunn
K-means	0.253	0.021
Hierarchical	0.285	0.043

By comparing the results we can notice how the best clusters were obtained through the Hierarchical clustering method. We proceed interpreting its results.

Results interpretation

We obtained seven clusters, each one with a different number of exoplanets: the cluster with the majority of exoplanets is the cluster 4, while cluster 1 contains only one planet. We can now focus our attention on some scatterplots to understand which are the main characteristics of each group of exoplanets.

In Figure 8 we can notice how the exoplanets of cluster 7 are the ones which orbit around the ”coldest” stars and also that have the lowest equilibrium temperature. This could be an indication that the temperature might not be too high for water to vaporize.

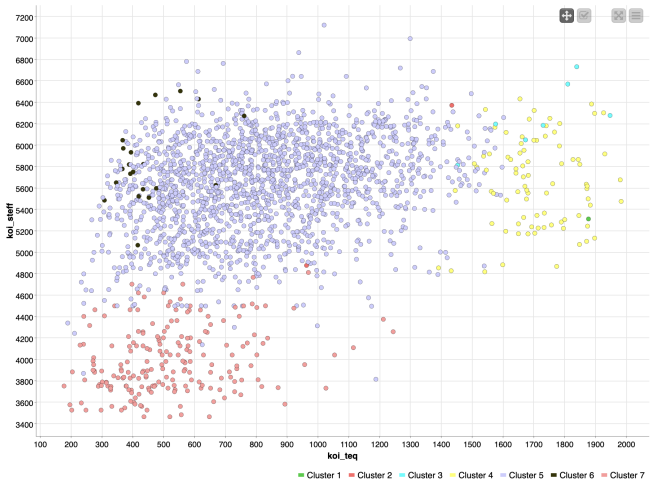

Figure 8. Stellar effective temperature vs Planet equilibrium temperature

In Figure 9 we can outline an interesting feature of the exoplanets included in cluster 7. The koi_sma indicates the ”Orbit Semi-Major Axis in Astronomical Units (AU)”, which indicates the average distance of the exoplanets from its sun during its orbit. Since it is given in AU we can compare the orbit of the exoplanets with the one of the Earth (which is 1

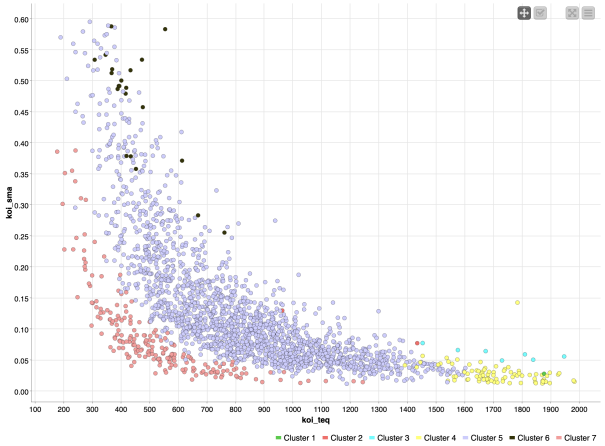

Figure 9. Orbit of a planet vs Planet equilibrium temperature

AU). This means that, if an exoplanet and its sun are similar to the Earth and our Sun, the optimal koi_sma would be 1 in order to hope for the presence of water on its surface. But since we have noticed that the exoplanets in the cluster 7 have colder stars compared to our Sun, and also lower equilibrium temperature compared to the Earth, we have reason to think that a rather smaller orbit around their sun can be compatible with the presence of water on its surface (or at least they have an acceptable temperature).

The scatterplot in Figure 10 suggests us another crucial aspect in order to understand which exoplanets are inside the cluster 7. We know that koi_prad indicates the planets radius compared to the radius of the Earth (if equal to 1, the exoplanet has the same radius of the Earth). This scatterplot indicates us that the exoplanets inside the cluster 7 are really close to the value of the radius of 1 and, more important, have a lower stellar effective temperature compared to all the other exoplanets of the other 6 clusters.

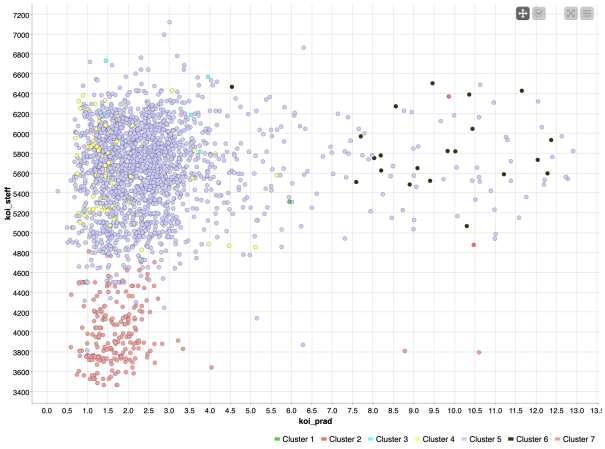

Figure 10. Stellar effective temperature vs Planet radius

Another important feature is the *insRelEarth*, which is the insolation received by the exoplanet compared to the Earth. Smaller values indicate a similarity with the insolation received by the Earth.
In Figure 11 we can notice that the exoplanets of the cluster 7 are the ones which orbit around the smallest stars (koi_srad is the stellar radius compared to the Sun) and which receive the most similar insolation flux compared to the Earth.

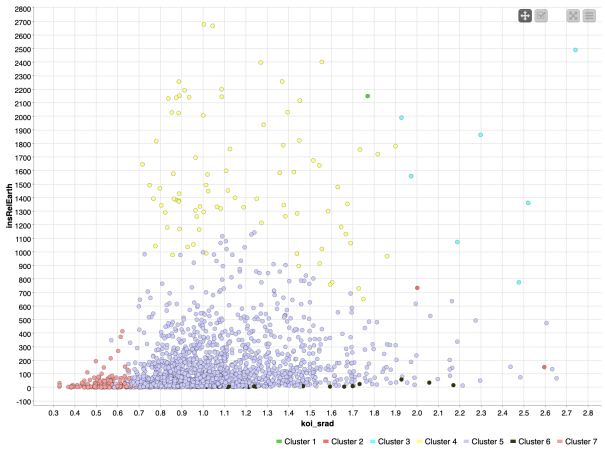

Figure 11. Insolation received by the exoplanet vs Stellar radius

Lastly, to better comparing these exoplanets with the Earth we group every cluster and we compute the mean for every features. Also we created two more variables:

- **tempRatio:** ratio beetwen the equilibrium temperature of Earth, and the star temperature;
- **ratioOrbit:** ratio between the semi-major axis (koi_sma) and the star radius (koi_srad).

By examining the scatterplot in Figure 12, we can observe that cluster 7 is the closest to the origin (0,0), which in our case represents the Earth's condition. However, there is an exception: cluster 6, which consists of Neptunian and Giant gas exoplanets, appears to be the closest to Earth's condition. That is because these type of exoplanets have the shortest distance to their respective stars, which happen to be some of the smallest stars. Therefore, someone could conclude that this cluster represents terrestrial planets, but it doesn't and that's because, by inspecting additional characteristics such as soil composition and atmosphere, it becomes evident that they are incapable of supporting life.

Some of the exoplanets that belong to the Terrestrial class and that are in the cluster 7 are:

- kepler 138 b
- Kepler-1351 b
- Kepler-779 b
- Kepler-1308 b
- Kepler-125 c

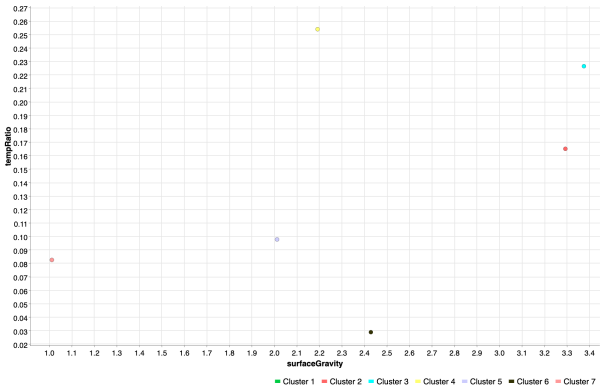

Figure 12. tempRatio vs surfaceGravity

5. Conclusion

In conclusion, the primary objective of this study was to identify a classifier which best predicted whether a given exoplanet was a ”confirmed,” ”false positive,” or ”candidate” exoplanet. To achieve this, we employed 4 different classification algorithms, including Random Forest, Logistic Regression, Naive Bayes, and Support Vector Machines (SVM). We highlighted that the Random Forest classifier is the optimal choice for this type of prediction. The Logistic classifier also performed well, particularly in predicting false positives, where it appeared to exhibit similar performance to the Random Forest.

The second task showed us how imprecise and difficult the Earth-alike exoplanets recognition is. This difficulty arises from both the imprecision of the data and our limited understanding as humans regarding the specific characteristics we should be seeking when searching for signs of life. However, since we think that a potentially habitable planet implies a terrestrial planet within the circumstellar habitable zone and with conditions roughly comparable to those of Earth [7], we could identify those planets which have these characteristics. In fact, cluster 7 is mostly composed of terrestrial planets, which are a type of exoplanet that share similar characteristics with the terrestrial planets of our solar system, such as Earth, Venus, Mars, and Mercury. Terrestrial exoplanets are often referred to as ”rocky” or ”Earth-alike” planets due to their composition and physical properties. Not all the terrestrial exoplanets have the potential to support life; this depends on various factors. However, among the gas giants, Neptunian-like planets, and super-Earths, terrestrial planets have the highest probability of hosting life. In conclusion, we can state that the exoplanets in Cluster 7 have the highest probability of being able to support life (or at least to be similar to Earth).

References

[1] Kaggle. Nasa - kepler exoplanet search results. *https://www.kaggle.com/datasets/nasa/kepler-exoplanet-search-results*.

[2] Nasa. Nasa exoplanet archive - koi table cumulative list. *https://exoplanetarchive.ipac.caltech.edu/docs/data.html*.

[3] Knime. Equal size sampling. *https://hub.knime.com/knime/extensions/org.knime.features-base/latest/org.knime.base.node.preproc.equalsizesampling-EqualSizeSamplingNodeFactory*.

[4] Vinícius Trevisan. Multiclass classification evaluation with roc curves and roc auc. *https://towardsdatascience.com/multiclass-classification-evaluation-with-roc-curves-and-roc-auc-294fd4617e3a*.

[5] The Pennsylvania State University Eberly College of Science. Agglomerative hierarchical clustering. *https://online.stat.psu.edu/stat505/lesson/14/14.4*.

[6] soumya7. Silhouette index – cluster validity index. *https://www.geeksforgeeks.org/silhouette-index-cluster-validity-index-set-2/*.

[7] Wikipedia. List of potentially habitable exoplanets. *http://scienceclass3000.weebly.com/uploads/5/4/5/9-5459088/list-of-potentially-habitable-exoplanets-wikipedia.pdf*.