Exoplanet Identification using Machine Learning on the NASA's Kepler Data

A Project Report
Submitted By

Chandramani

under the supervision of

Dr. Kemal Davaslioglu

Principle of Machine Learning

2022

Since many ages ago, only astronomy professionals and researchers have been able to identify planets with the aid of sophisticated tools. This pattern has changed with the development of computational techniques and the availability of satellite data from space missions. For example, the Exoplanet Exploration [1] program of NASA has given us access to a wealth of information on celestial bodies to aid in space exploration. The Kepler mission is one such intriguing mission. Since the mission's inception in 2007, more than 4000 such transiting exoplanets have been found, recognized, and cataloged. It concentrated on learning about planets and planetary systems.Four fundamental models have been explored. Specifically, XgBoost, Deep Neural Networks, Support Vector Machines, and Random Forest Classifiers. The XgBoost classifier, which produced an F-1 score of 98% on the dataset, was chosen as the best machine learning model.

1 Introduction

Astronomy has long fueled our curiosity and caused us to wonder about and consider the enigma surrounding our existence and place in the universe. Given that we live on one planet and may soon need to discover another, it is a natural human predisposition to wonder if the astronomical bodies we are aware of are planets or not. The challenge of detecting exoplanets, which are essentially planets distinct from the ones we know, far beyond our solar system, has led to novel discoveries. This question has engaged the imaginations of scholars for generations.
Up until now, this has been a time-consuming and exhausting task that has only been performed by astronomers with extensive domain knowledge and involved the use of specialist tools. These specialists look at pictures that satellite-based telescopes like the Hubble have taken. A new window has opened up for exoplanet exploration with the introduction of a brand-new generation of modern astronomy, technologically advanced telescopes, and satellites like the Kepler, with the ultimate goal being to automate the analysis of scientific observations and data generation pertaining to exoplanet identification. In addition to producing pictures, these satellites and telescopes also use astronomical and mathematical techniques to analyse the pictures, creating data with a range of attributes for identifying these exoplanets. The previously difficult process of exoplanet investigation among data scientists, engineers, and statisticians has been democratized by the availability of this data. Exoplanet data can be used with contemporary intelligent algorithms and techniques like Machine Learning and Deep Learning to find missed and incorrectly classified exoplanets and items of interest in data archives or automate the workflow. I investigated Support Vector Machines, Random Forest Classification, XGBoost, and Deep Neural Networks for the classification. They were used to categorize objects in the Kepler Cumulative Object of Interest (KCOI) table. The KCOI table contains 50 Kepler data-recorded features.

2 About the Dataset

NASA launched the Kepler Space Telescope in 2009. It has been the most successful telescope in aiding the discovery of exoplanets to date. It has discovered thousands of objects of interest, including over 4300 confirmed exoplanets. The catalog has a high level of reliability, averaging 85%-90% across the radius plane. It continues to improve as more observations are made. While Kepler was officially decommissioned in October 2018 due to a lack of fuel, the statistical data it gathered can provide insights for new exoplanet discoveries for years to come.

3 Workflow

My project focuses on a binary classification of Objects of Interest as "FALSE POSITIVE" or "CONFIRMED" exoplanets. NASA uses the label of "FALSE POSITIVE" to indicate the satellite incorrectly tracked an object. I did not consider the observations labelled as "CANDIDATE" since these are yet to be labelled by NASA and hence, are unknown to me. For my analysis, I have used four models, each with its own unique characteristics to tackle the problem at hand from different angles. The four models used are Support Vector Machines (SVM), Random Forest, XgBoost and a Feed-Forward Neural Network.

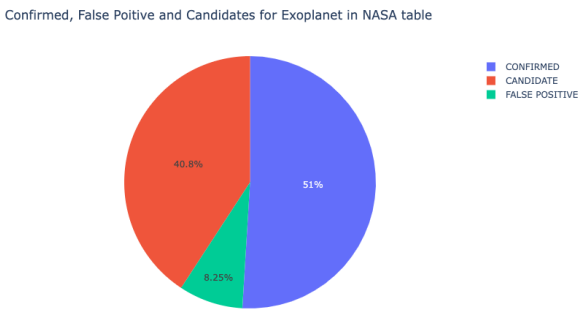


Figure 1: *Class Distribution on the data*

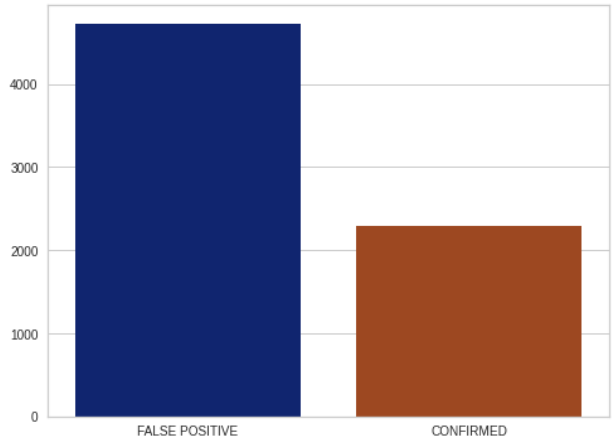


Figure 2: *Binary Classes for our models (Imablanced Dataset)*

The Cumulative Kepler Object of Information dataset is not without faults. The dataset is a raw one, obtained directly from the readings measured by Kepler after assigning labels, and hence has not only missing values but a lot of unnecessary columns as well. Some of these can be dropped since they are ID attributes assigned but a majority of them are crucial to our analysis need to be filled in appropriately. This step helps us tune our model's performance and remove redundant features which do not add any information to our analysis.

3.1 Data Preprocessing

I begin by looking at the columns that have missing values. I noticed that, while many of these columns have missing values, the majority of these missing values correspond to errors associated with attribute values.
I also noticed that two error attributes - the positive and negative errors associated with Equilibrium Temperature - have missing values. As a result, I decided to drop them entirely because their values cannot be imputed. I see a highly skewed distribution of a few attributes for the remaining error attributes. It would be unwise to replace these values with their average, so we decide to fill in the gaps with the median error value.

My analysis excludes the attributes *kepler name, koi tce delivname, and koi tce plnt num*. These are ID attributes that have been assigned to the objects of interest after analysis and labeling (into exoplanets and non-exoplanets), and thus do not play a role in the classification model's construction. As a result, these attributes are excluded from the analysis. I completed the data pre-processing step by standardizing our data, ensuring that all attributes are on the same scale. This improves performance while also reducing the computational effort required by the models, speeding up the classification process.

3.2 Features Correlation and Variance

I examined the association between the attributes and track the outcomes in order to decrease redundancy in the number of attributes selected. While there is a correlation between the attributes chosen for the project, the majority of the correlations are not of a high order, as can be shown. Furthermore, only a small number of characteristics show strong connection with the dependent variable. The majority of characteristics are seen to only have a medium or low connection. Additionally, we see that most attributes have negative correlations with the dependent variable, indicating that the likelihood of an object being an exoplanet is increased when these attributes have lower values.

I did Principal Component Analysis (PCA) on the dataset and select the number of Principal Components as the number of chosen attributes in order to observe the variance that the attributes added to the dataset. I find that only after accounting for every single attribute or component does the total variation reach 100%, and that the first principal component only accounts for 15% of the total variance. The steady drop in the curve demonstrates how each individual component significantly increases the dataset's variance. As a result, I choose to continue using our current set

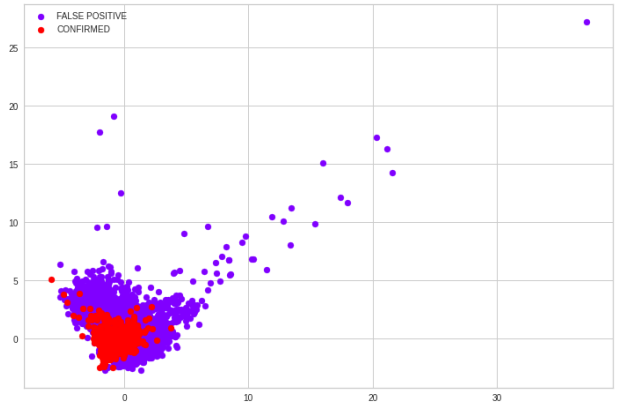


Figure 3: *PCA Decomposition*

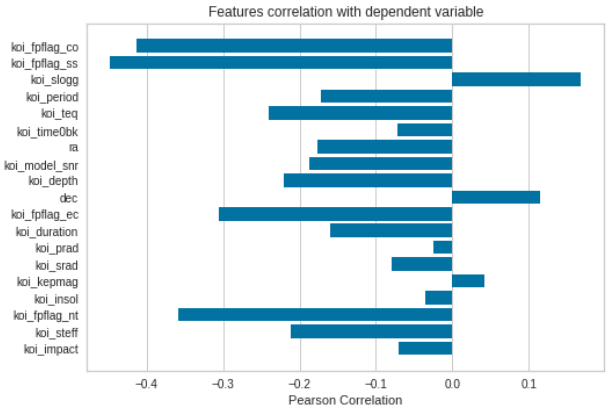


Figure 5: *Pearson Co relation*

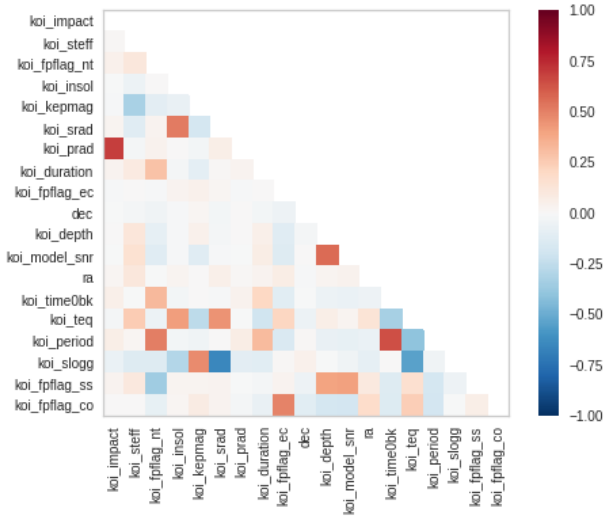


Figure 4: *Pearson Corelation HeatMap*

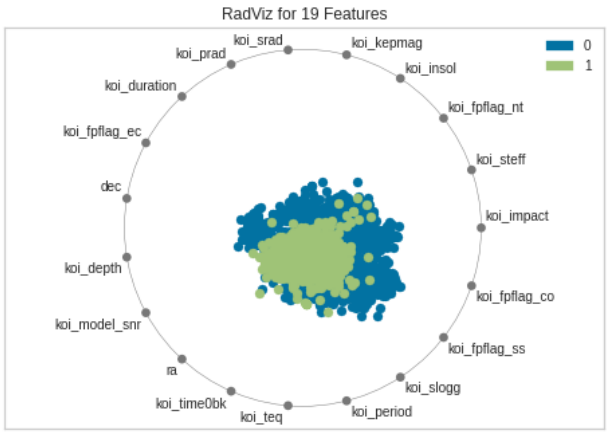


Figure 6: *Radial Visualisation without Error Attributes*

of qualities without changing any of them.

Final observations also indicate that the spread of values for non-exoplanets is far greater than the spread of values for confirmed exoplanets. This shows that the magnitude of attributes for exoplanets is lesser compared to other objects of interest.

3.3 Evaluation Metrics and Cross Validation

To compensate for the dataset's imbalance, I used various evaluation metrics that take the imbalance into account. These are the F1 Score, the Balanced Accuracy Score, and the Confusion Matrix Furthermore, we use K-Fold cross-validation across our entire dataset to ensure that we are not underfitting our classifier by introducing high bias when testing it on different sets. Because the dataset is imbalanced, we use both a non-stratified and a stratified splitting to ensure that the number of positive and negative examples within each fold is equal. We measure our classifier's performance across each split and then take the mean of the results.

3.4 Classification Models

Preliminary visualizations revealed that the dataset is skewed. Specifically, the distribution of examples within each class is skewed toward the negative class. This is due to the greater abundance of non-exoplanet bodies in the universe than actual exoplanets.

There is also a lot of noise in the dataset, as well as inter-mixed classes. To see how different classifiers approach this problem, we propose using traditional classifiers like the **SVM**, Bagging Ensemble models like the **RandomForest**, Boosting Classifiers like the **XGboost**, and finally a **FeedForward Neural Network**. In addition, we develop an Ensemble majority vote classifier, which combines all of these to take the majority vote and return the predicted label.

The attributes considered in our study include all attributes except the ID attributes, which assign names and IDs to each object of interest, as well as any other attributes assigned by NASA after studies on these data points have been performed.

3.4.1 Support Vector Machine

For SVM I used Gaussian or an RBF kernel to tackle the issue of my classes being non-linearly separable. This kernel performs a Gaussian transformation to map the input feature space to a higher dimensional space. I then tune my model by performing cross-validated GridSearch and RandomSearch CV on it, allowing it to run through multiple combinations of parameters and finally choosing the best model based on its performance.

Table 1: *Chosen GridSearch Parameters for SVM*

Attribute	Chosen Value
shrinking	true
gamma	scale
C	1.15
class_weight	None, balanced
tol	1
class_weight	None

3.4.2 Random Forest

An ensemble-based bagging classifier built on a decision tree serves as our second classifier. In order to classify the input test example, the RandomForest trains many separate Decision Trees on each subset of the dataset before averaging the results. The hyperparameters of each Decision Tree in the Random Forest allow it to be modified individually. Because the Random Forest is so powerful, creating an accurate model requires less fine tuning. I use RandomSearch to adjust each Decision Tree's hyperparameters in order to select the most ideal set that produces the greatest results.

3.4.3 Neural Network

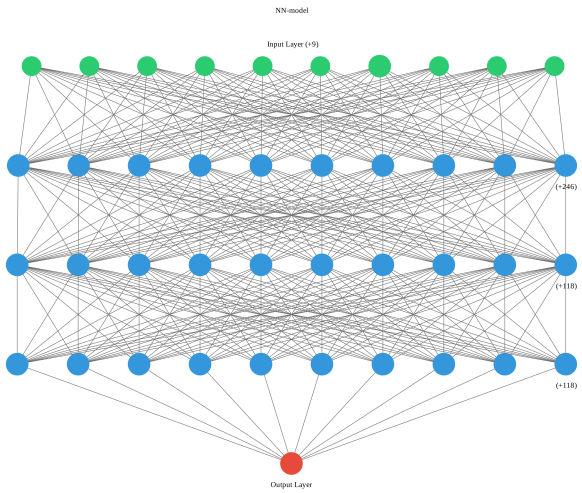
I train a typical Neural Network architecture to perform classifications as my next model. I change the number of layers and the number of neurons in each layer to fine-tune our neural network. Since

Table 2: *Chosen RandomSearchCV Parameters for Random Forest*

Attribute	Chosen Parameters
n-estimators	400
min-samples-split	10
min-samples-leaf	1
max-features	auto
max-depth	50
bootstrap	True

I are performing binary classification, I fix my activation functions for all layers except the final output layer to be ReLu and the final layer to be Sigmoid. Finally, I select Binary Cross Entropy as the Loss Function and Adam as the Optimizer.

Figure 7: *Neural Network Model*



3.4.4 XGBoost

I used XGBost as my last classifier. XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.
I tuned XGBoost using the following parameters using RandomSearchCV

Table 3: *HYPER PARAMETERS FOR NEURAL NETWORK*

Hyperparameters	Chosen Parameters
Layer Dimension	256, 128, 128, 1
Optimizer	Adam
Loss	binary_crossentropy
Metrices	binary_accuracy
Epochs	20

Table 4: *Chosen RandomSearchCV Parameters for XGboost*

Attribute	Chosen Parameters
n-estimators	200
min-child-dweight	1
max-depth	3
learning-rate	0.1
gamma	0.1
col-sample-bytree	0.5

4 Results

I tested using 10-Fold cross-validation on the entire set of observations. Cross-validation helps analyse the model's performance against each subset of the dataset by training the model on n-1 folds and testing the model on the nth fold. Cross-validation also creates stratified splits, thus creating a pipeline to estimate the model's accuracy even on imbalanced datasets.
Each of the aforementioned models are subjected to crossvalidation. I use the F-1 Score to mark a model's performance, since the dataset is imbalanced and a plain accuracy score would paint the wrong picture I observe that on a dataset without the error attributes, all the models work really well. However, in terms of numbers the XGBoost classifier obtains the best performance with an average F-1 score of 98.16%. The Neural Network Model was in 97.34%.

4.1 Analysis of Feature Importance

Different machine learning algorithms operate in different ways. They all differ in terms of classification method, sampling, and variable initialization for modeling. This results in differences in the features being analyzed, as well as the priority or importance assigned to each feature when performing a prediction or classification. I investigated the differences in feature importance by observing the features that have the greatest impact on the model's performance. These features are then plotted in order of importance to demonstrate how each model ranks the features. However, because the features are transformed into a higher dimensional feature space, this is not possible with the Support Vector Machine.
The XgBoost and Random Forest classifiers assign the highest importance to flag variables namely the Centroid Offset flag and Non-Transit like Flag. We also observe a steep decline in the importance of attributes, beyond the set of flag attributes thus suggesting that the XGBoost and Random Forest classifiers are extremely powerful while working with categorical variables and uses them to make the core decisions while splitting the dataset into decision nodes. This also proves that categorical variables help perform classifications faster since unlike continuous variables, they do not have to be binned.

Table 5: *MODEL PERFORMANCES*

Model	Stratified F-1 Score	Non-Stratified F-1 Score
SVM	97.7%	97.75%
Random Forest	98.0%	97.07%
Neural Network	97.7%	97.34%
XGBoost	98.16%	98.32%

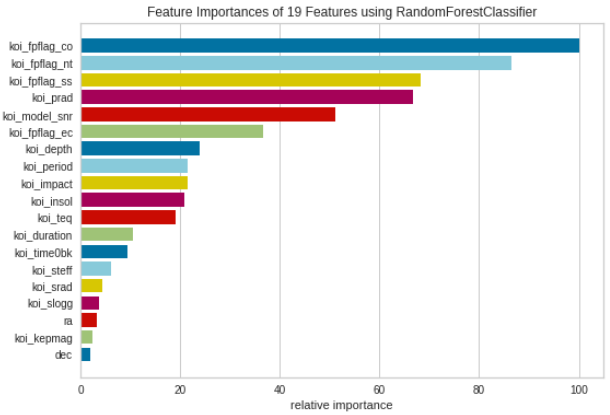


Figure 8: *Feature Importance for Random Forest Classifier*

The XgBoost classifier displays a different order of feature importance, even though it uses a De-

cision Tree, just like the Random Forest classifer. Flag variables although exhibiting a high level of importance, do not take the top spot. This is accounted by the difference between Boosting and Bagging. Unlike Random Forest, I also observe that the feature importance tends to show a pattern, with the magnitude of importance taking a discrete set of proportions. Additionally, we also observe that the XgBoost classifier can assign an importance of zero to any feature.

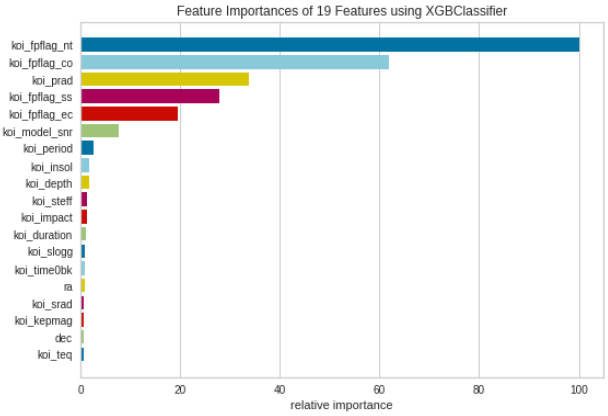


Figure 9: *Feature Importance for XGBoost Classifier*

On the other hand the, Neural Network returns a completely different set of ranked features. However, similar to the Random Forest classifier, the flag attributes are ranked highest. This confidently illustrates that categorical variables drive the process of making well-informed decisions that lead to accurate predictions in the classification of observations. Categorical variables speed up decision making and reduce overall entropy in a set of observations regardless of the algorithm used.

5 Observations

There are numerous criteria that can be used to group planets. Planets are classified based on factors such as mass, orbital range, and composition, to name a few. Planets in the same group or cluster often have similar properties to their neighbors and are used to classify or group newer planets. Scientists have attempted to analyze readings to group together such planets over the years in an effort to make analysis of these heavenly bodies simpler and more organized. Today, we can divide exoplanets into four broad categories.

- Gas Giant - similar to Jupyter or Saturn
- Neptune Like - similar to Uranus or Neptune

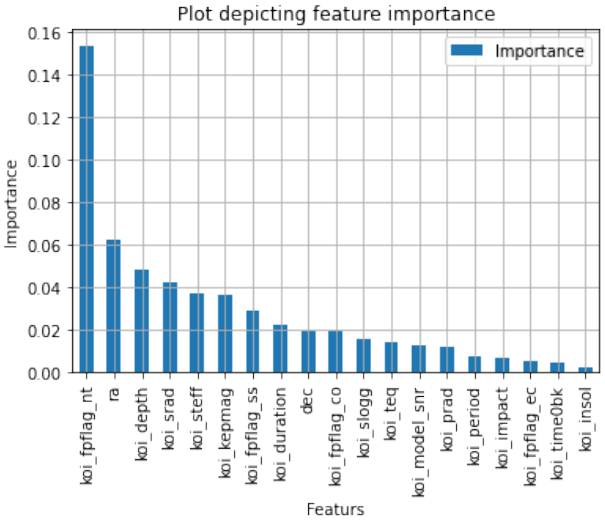


Figure 10: *Feature Importance for Neural Network*

- Super Earth - more massive than Earth but lighter than Neptune
- Terrestrial - Rocky and Earth-like

  To facilitate the study of these exoplanets, they have recently been divided into eight categories. Because these additional categories are branching off from the main ones, they have a lot of overlap. *Gas Giant, Mesoplanet, Mini Neptune, Planemo, Planetar, Super Jupyter, Super Earth, and Sub Earth* are the eight categories.

  A cluster analysis can be used to examine the distribution of exoplanets in the dataset. This will allow us to see the differences in these exoplanets' characteristics as well as their distribution into categories.

5.1 K Means Clustering

K-Means Clustering is an iterative clustering algorithm that is used to group similar objects together and identify patterns. Clusters are formed by assigning and recalculating centroids until convergence is reached. These centroids are highly representative of a cluster and are frequently used to represent group characteristics as well.
K-Means requires the number of clusters (or categories) to be fed into the model as a hyperparameter. The Sum of Squares Error is the most commonly used technique for determining the optimal number of clusters (SSE). This method computes and plots the SSE for each batch of clustering across a variable number of clusters. At this point, both the SSE value and the number of clusters are

balanced so that the set of observations is neither underfit nor overfit.

I observed that 7 clusters are returned optimally

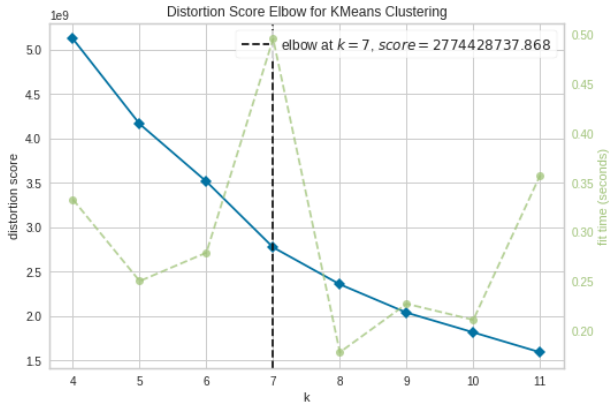


Figure 11: *SSE Method to Find Optimal Number of Clusters*

rather than 8. This is due to noise in the set of observations in the form of overlaps between classes. A distribution visualization reveals that the dataset contains four primary groups of planets, with additional groups formed by breaking down one of the classes.

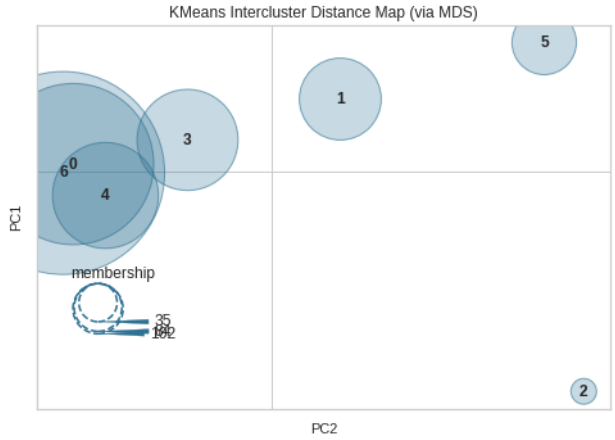


Figure 12: *Inter Cluster Distances*

5.2 Agglomerative Hierarchical Clustering

Another clustering algorithm is Agglomerative Hierarchical Clustering, which creates clusters by iteratively combining observations that are similar to each other into a single cluster until only one cluster remains. Unlike K-Means, however, Hierarchical Clustering does not require the number of clusters to group together observations. The number of clusters required can be calculated using a dendogram, which is a hierarchical tree that stores the order in which the clusters formed.

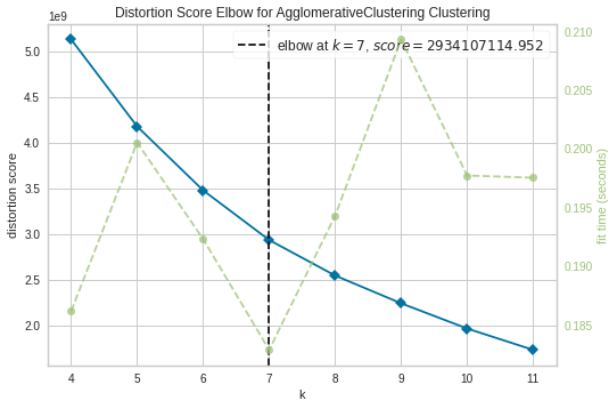


Figure 13: *Optimal Number of Clusters Using Agglomerative Hierarchical Clustering*

However, we can still find the optimal number of clusters recommended using a similar approach. I again obtain the optimal number as 7, thus proving without doubt about the overlap between the classes.

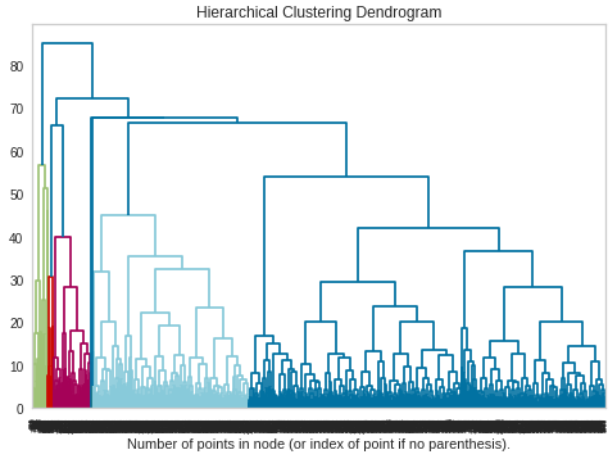


Figure 14: *Dendogram of Merges Obtained*

6 Lessons Learned

Some attributes, such as Insolation Flux and Equilibrium Temperature, are vastly different across classes compared to the remaining attributes, whereas others, such as Right Ascension and Kepler Band Magnitude, are very similar across classes.

This demonstrates that there is no significant trend between attributes, as exoplanets of varying

sizes and compositions can have a similar set of attributes. Because of the variation in characteristics, it is becoming increasingly difficult for researchers and scientists to identify planets and declare them habitable.

The analysis was thorough and provided extremely useful insights into exoplanets and the methodologies used to classify them.I did a lot of literature research to get background information on exoplanets, which was extremely helpful in planning and building my models. I also developed the hypotheses to be tested and worked on Deep Learning-based approaches.

References

[1] NASA, Kepler and k2 mission..

[2] CalTech *Cumulative kepler object of interest data*
https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nphtblView?app=ExoTblsconfig=cumulative,2020

[3] N. M. Batalha *Exploring exoplanet populations with nasa's kepler mission*
https://www.pnas.org/content/111/35/12647, 2014

[4] A. V. Christopher J. Shallue
Identifying exoplanets with deep learning: A five planet resonant chain around kepler-80 and an eighth planet around kepler-90
https://arxiv.org/abs/1712.05044

[5] M. J.-M. Vicente Alarcon-Aquino
Transiting exoplanet discovery using machine learning techniques
shorturl.at/mG069

Appendix

Table 6: *Attributes Chosen for Analysis*

dec	KIC Declination
koi depth	Transit Depth (parts per million)
koi duration	Transit Duration (hours)
koi fpflag co	Centroid Offset Flag
koi fpflag ec	Ephemeris Match Indicates Contamination Flag
koi fpflag nt	Not Transit-Like Flag
koi fpflag ss	Stellar Eclipse Flag
koi impact	Impact Parameter
koi insol	Insolation Flux [Earth flux]
koi kepmag	Kepler-band
koi model snr	Transit Signal-to-Noise
koi period	Orbital Period (days)
koi prad	Planetary Radius
koi slogg	Stellar Surface Gravity
koi srad	Stellar Radius
koi steff	Stellar Effective Temperature
koi teq	Equilibrium Temperature (Kelvin)
koi time0bk	Transit Epoch
ra	KIC Right Ascension

Confusion Matrix

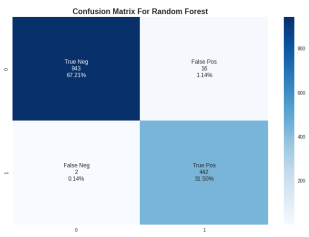


Figure 15: *Random Forest*

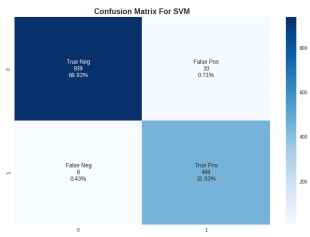


Figure 16: *SVM*

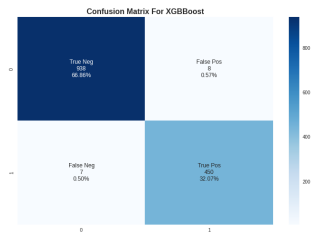


Figure 17: *XGBoost*