UNIVERSITAT POLITÈCNICA DE CATALUNYA

BARCELONA SCHOOL OF INFORMATICS

Official Master on Data Science

Project Course

Machine Learning

Maria Paraskeva
Martin Tazón

Barcelona, June 2023

Abstract

The discovery of exoplanets has revolutionised our understanding of the universe, igniting a surge of research in exoplanetary science. With the advent of advanced telescopes like Kepler and observational techniques like the transit method, vast amounts of data on potential exoplanetary systems have been collected. In this study, we employ a range of machine learning algorithms to predict whether a given observation represents an exoplanet or not.

We begin by utilising a comprehensive exoplanets dataset, encompassing various astrophysical features and labelled exoplanet classifications. Logistic regression, decision trees, random forest, support vector machines, and neural networks are employed as the primary classification models. These models are trained and evaluated on a subset of the dataset, employing appropriate cross-validation techniques to assess their performance.

Results demonstrate that all models achieve considerable accuracy, with logistic regression achieving an accuracy of 76%, decision trees at 79%, random forest at 82%, support vector machines at 74%, and neural networks at 83%. Additional evaluation metrics such as precision, recall, and F1 score are also considered to comprehensively assess the model performances.

The next step involves selecting the best model among the ensemble of classifiers. A comparative analysis is conducted, considering factors such as accuracy, interpretability, computational complexity, and generalizability. Based on the evaluation metrics, the neural network model emerges as the top-performing classifier, demonstrating superior predictive capabilities, robustness, and efficiency.

The chosen model is then used to make predictions on unseen observations from the dataset. The performance of the selected model is compared against the other models to validate its superiority. The results showcase the neural network's ability to effectively distinguish exoplanets from non-exoplanets, yielding an accuracy of 83% on the test set.

In conclusion, this study provides a comprehensive analysis of various classification models for exoplanet detection. The findings underscore the effectiveness of machine learning techniques in identifying exoplanetary systems and highlight the neural network as the most promising model. The insights gained from this research can aid astronomers and astrophysicists in accelerating the discovery and understanding of exoplanets, thus advancing our knowledge of planetary systems beyond our own.

1 Introduction

In this course project we have selected a dataset and applied the methods learnt during the course, made appropriate adjustments, and reached some conclusions. This has given us the opportunity to put in practice the theoretical knowledge of the semester and test our skills.

The dataset of our choice comes from the NASA Exoplanet Science Institute (KOI) and can be found here: NASA Exoplanet Dataset | Kaggle. It is a very interesting dataset that has gathered data from the Kepler mission, revealing thousands of planets outside of our solar system. The dataset consists of 49 columns and 9564 rows, each row corresponds to a specific light source in the visible universe (presumably a star) that the Kepler telescope has analysed (Kepler Object of Interest) during a fixed period of time hoping to find evidence that there exists a planet revolving around the observed star. In order to identify possible exoplanets, the Kepler mission studies properties of light signals that arrive from each individual star. In a nutshell, the Kepler mission expects to capture transits of a planet that partially cover the observed star surface from out frame of reference. The columns of the dataset correspond to properties of the light signals analysed, some of these properties include periodicity of the dims in the brightness of the star and the intensity of the dims among others. Since these measurements have been recorded during an extended period of time (more than four years) the dataset includes the average value of the metric together with the statistical error of the measurement, both for the positive and negative tails of the distribution.
This method produced a large list of stars that were candidates for hosting exoplanets, and further efforts have been made to confirm if these candidates were in fact exoplanets.

The objective of this analysis is to predict, based on the Kepler indicators alone, if a candidate is going to be confirmed as a planet or not and then evaluate the accuracy of the results.

2 Data Exploration

2.1 Target Variable

After taking a closer look at the dataset and its meaning, we have decided to use the variable *koi_disposition* as our target variable, which is the data observed by humans. We can see that it is a categorical variable containing three classes; False Positive, Confirmed, and Candidate, with the following number of observations:

```
FALSE POSITIVE    4840
CANDIDATE         2367
CONFIRMED         2357
```

In order to simplify this analysis, we are going to remove the False Positive cases since they can be the objective of another type of exercise. Thus we are trying to predict the Confirmed and Candidate cases, transforming our analysis into a binary classification problem.

On another note, we used *kepoi_name* as the index to identify the rows, as it shows the scientific name of the Kepler object of interest.

2.2 Feature Selection and Engineering

2.2.1 Drop Variables

We proceed to the removal of the variables *kepid* and *kepler_name* since they only serve to identify the points and add no significant information. We also removed *koi_score* since this

is the confidence on the machine's prediction and it shall not be used as a predictive variable. Other variables we removed were *koi_teq_err1* and *koi_teq_err2* since they only contained missing values.

From the data statistics, we noticed that *koi_fpflag_co* and *koi_fpflag_ec* are always equal to 0, so we decided to drop them. The first variable represents the Centroid Offset Flag while the second the Ephemeris Match Indicates Contamination Flag.

Regarding the first variable, the statement shows that the source of the signal is from a nearby star, as inferred by measuring the centroid location of the image both in and out of transit, or by the strength of the transit signal in the target's outer (halo) pixels as compared to the transit signal from the pixels in the optimal (or core) aperture. Either way, a false positive suggests the source isn't coming from an exoplanet, so we drop it.

The scientists can evaluate this by measuring the centroid location of the image both when the exoplanet is in front of the star (in transit) and when it is not. The centroid is essentially the centre of the image. If the centroid shifts significantly when the exoplanet is in transit, it suggests that the transit signal is not coming from the exoplanet, but from a nearby star. In the other case, they can look at the strength of the transit signal in the outer pixels (halo) of the image compared to the signal from the core pixels (optimal aperture). If the transit signal is stronger in the outer pixels, it suggests that it is not coming from the exoplanet, but from a nearby star.

When it comes to the second variable, the statement means that the KOI in question has the same period and epoch as another object. This suggests that what was detected may not actually be a planet, but rather a result of "flux contamination" or "electronic crosstalk". Flux contamination refers to the interference from other sources of light, such as nearby stars, while electronic crosstalk refers to the interference caused by the electronics of the telescope itself.
Therefore, the statement suggests that the detection of the potential exoplanet is not valid, as it is likely a false positive caused by technical issues with the telescope or other sources of light, hence it is dropped.

Another variable that we decided to drop was *koi_fpflag_nt*, which stands for Non Transit-Like Flag. It has only three entries that are non zero while all the rest are. This means that the light curve of these three objects is not consistent with that of a transiting planet and could be anything from an instrumental artefact to spurious detections.

2.2.2 Missing Values

Continuing our feature engineering procedure, we completely removed any rows who had missing values. These rows were only 194 in total so removing them did not severely affect the size of our dataset. *Figure 1.* shows a heatmap of the missing data.

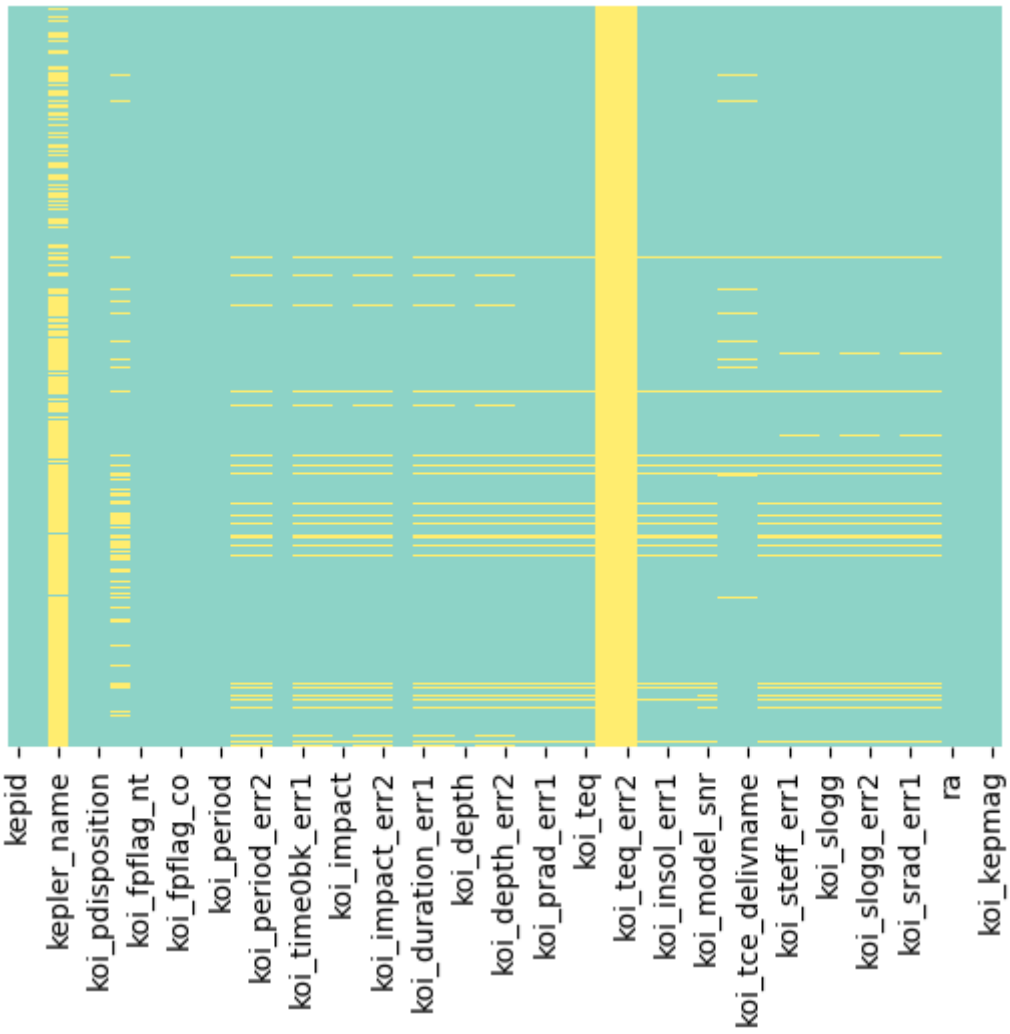

Figure 1.

We also observed that eleven of our variables have respective variables named *err1* and *err2*. Many of the *err2* columns are highly correlated to their *err1* column. We decided to drop the ones that have a correlation higher than 70%, since their contribution in our analysis will not be significant.

2.2.3 Categorical Values

Regarding the conversion to categorical values, we converted the following variables: *koi_disposition, koi_tce_plnt_num, koi_tce_delivname, koi_fpflag_ss.* Later we can see that we changed the categorical variables into dummy variables, as they are easier to work with and are best suited for fitting machine learning models.

2.2.4 Outliers

After the outlier treatment, the percentage of each variable that previously had significant outliers decreased as shown in *Table 1.*, with the total percentage of considered outliers reaching only 2.76%.

Variable name	Percentage of values considered outliers
koi_period	0.044150110375269946 %
koi_period_err1	0.0883002207505541 %
koi_time0bk	0.02207505518764208
koi_time0bk_err1	0.1766004415011082
koi_impact	0.11037527593819618
koi_impact_err1	1.081677704194263
koi_impact_err2	0.11037527593819618
koi_duration	0.22075055187637815
koi_duration_err1	0.11037527593819618
koi_depth	0.15452538631346613
koi_depth_err1	0.1766004415011082
koi_prad	0.24282560706400602
koi_prad_err1	0.2869757174393044
koi_teq	0.11037527593819618
koi_insol	0.11037527593819618
koi_insol_err1	0.19867549668873608
koi_model_snr	0.1766004415011082
koi_steff	0.13245033112583826
koi_steff_err1	0.15452538631346613
koi_slogg	0.0883002207505541
koi_slogg_err1	0.02207505518764208
koi_slogg_err2	0.02207505518764208
koi_srad	0.0883002207505541
koi_srad_err1	0.0883002207505541
koi_kepmag	0.02207505518764208

Table 1.

The scatterplots for all pairs of variables are shown below in *Figure 2*. The two distinct colours represent the values “Confirmed” (orange) and “Candidate” (blue) of our target variable. These plots can give us a more general idea of the correlation between each pair of variables. For example, looking at the plots of the variable *koi_model_snr*, the data points for each colour are visibly distinguishable from the rest. This means that this specific variable is strongly correlated with all the other variables, and probably plays an important role in the dataset.

What caught our attention were the plots of the variables *ra* and *dec.* The first one represents the KIC right ascension and the second one the KIC declination, both measured in degrees. The scatterplot of *ra* and *dec* in the field of astronomy often exhibits a cross-like shape due to the celestial coordinate system and the orientation of the sky. This pattern is known as the "celestial cross."

The celestial cross arises from the way right ascension and declination are defined and measured. Right ascension is measured along the celestial equator, which is an imaginary line projected onto the sky, corresponding to Earth's equator. It is divided into 24 hours, with 1 hour equivalent to 15 degrees. Thus, the range of right ascension goes from 0 hours (0 degrees) to 24 hours (360 degrees), forming a complete circle around the celestial sphere.

Declination, on the other hand, is measured along the celestial meridian, which is an imaginary line that extends from the north celestial pole to the south celestial pole. Declination is measured in degrees, ranging from -90 degrees at the south celestial pole to +90 degrees at the north celestial pole. The celestial equator has a declination of 0 degrees.

The cross-shaped scatterplot is a result of the spherical nature of the celestial coordinate system and the specific orientation of the celestial equator and celestial meridian.

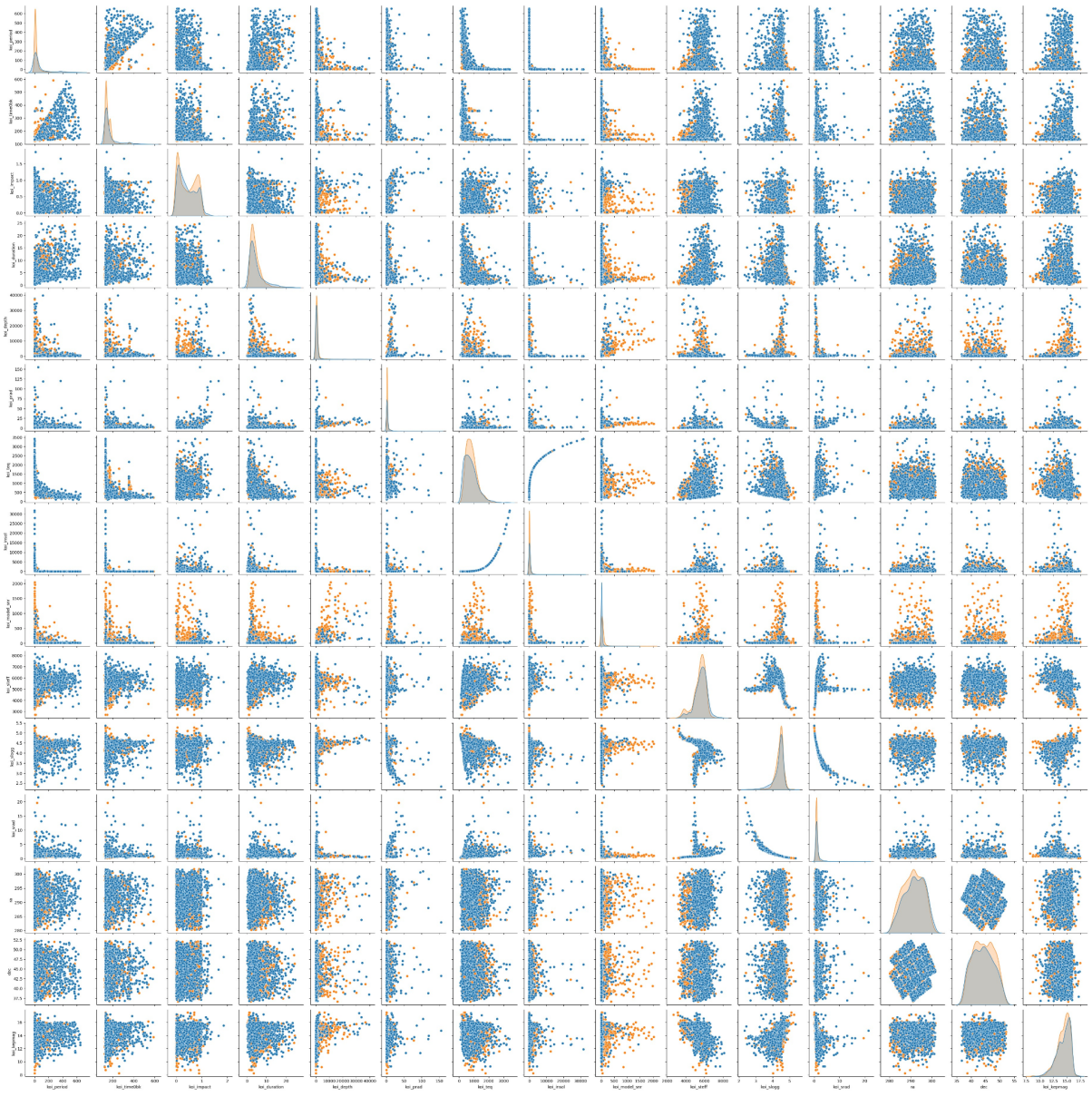

Figure 2.

3 Model Selection

Before splitting our preprocessed dataset into a training and a testing dataset, we isolated 10% of it as validating part of the whole dataset. We then proceeded to split the rest of the data set into train and test with a 70%-30% size respectively.

In order to avoid some features from dominating others and ensure each feature contributes equally to the learning process of each model, we then proceeded to perform min-max scaling to all numerical features. This will provide a stable and consistent input in all the algorithms while also preventing numerical instability or slow convergence caused by features with significantly different scales. Finally, it will make interpreting and visualising the data easier, as the data range will be strictly defined.

3.1 Model Fine-tuning

For each of the following models, we created a fine-tuning function which selects the best model for all the given parameters. In each section we will specify the parameters that result in the highest F1 score and give the respective interpretation.

Also, in order to compare all models and gain insights on the ones which perform better, we are going to focus on the F1 score instead of the macro average. The reason behind this decision is that the macro average focuses on evaluating performance class-wise and averaging the results, while the F1 score combines precision and recall to provide a single metric that represents the overall performance of the model.

For each of the following models, we also computed confusion matrices that can be found in the notebook. Due to the page limit, we decided not to include them in this document.

3.2 Logistic Regression

Our first logistic regression model gave the results shown in *Table 2*. The precision of the observations that could be an exoplanet appears higher by 0.14 than the ones that are, meaning that they have more correct positive predictions relative to the total positive predictions. On the other hand, the recall of the confirmed exoplanets are higher with a difference of 0.27, indicating that the model has more correct positive predictions relative to the total actual positives. This results in a higher F1 score for the confirmed observations and a total F1 score of 0.76.

	precision	recall	f1-score	support
CANDIDATE	0.85	0.62	0.72	585
CONFIRMED	0.71	0.89	0.79	605
accuracy			0.76	1190
macro avg	0.78	0.76	0.75	1190
weighted avg	0.78	0.76	0.75	1190

Table 2.

3.3 Decision Trees

The second model we created used the Decision Tree algorithm. The criteria to choose from were three; entropy, gini impurity, and logarithmic loss. Entropy and gini impurity are both measures of the impurity or disorder within the dataset. The goal in both of them is to minimise their index by selecting the feature that produces the lowest impurity after the split. Our fine-tuning algorithm selected the logarithmic loss (or log loss) criterion, which is a measure of the performance of the classification model that provides a continuous and differentiable value. It evaluates the quality of the tree split and the goal is to also minimise the log loss, which means choosing the feature that results in the lowest log loss after the split.

We first executed the algorithm with no maximum depth limit and the result was an overfitting model which can be seen in *Figure 3.*

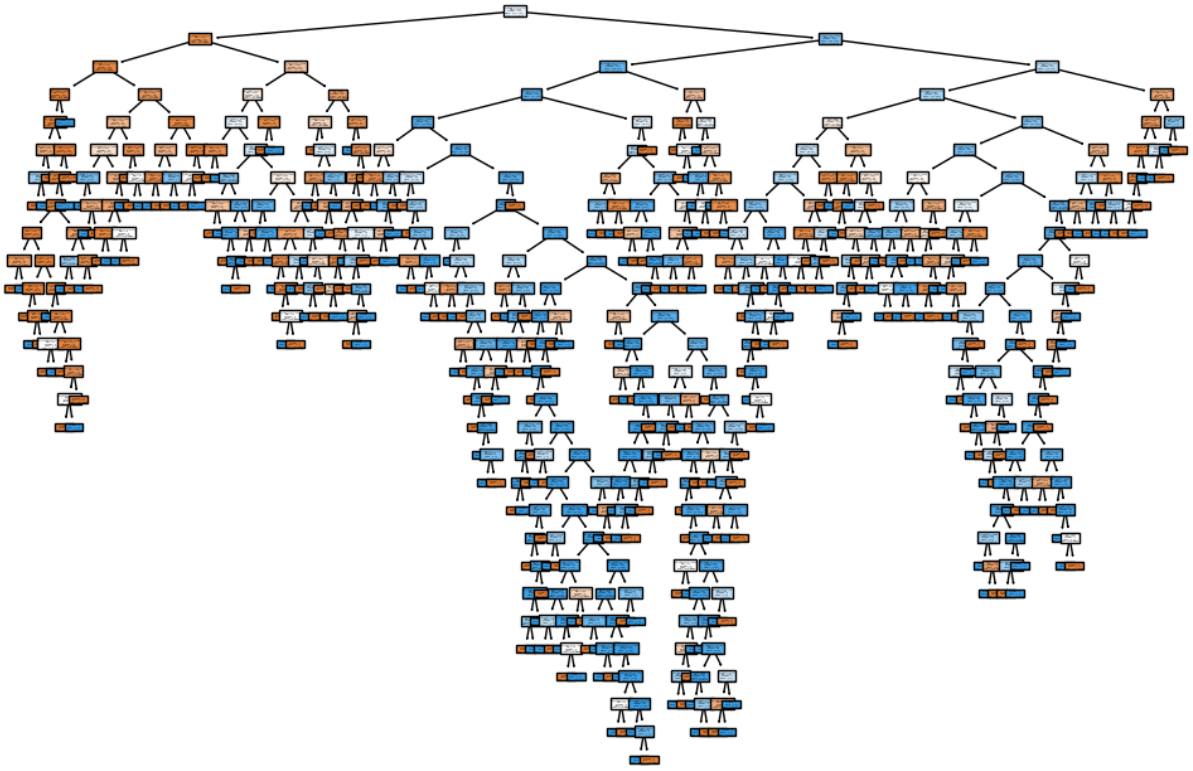

Figure 3.

We then gave three values (3,5,10) as an example to our fine-tuning algorithm and the optimal depth among them was proved to be 5. The solution can be seen in *Figure 4*. There is a possibility that this tree overfits as well, so, in order to overcome this, we decided to produce many smaller trees and choose the best one among them. This action was completed by our next model using Random Forests.

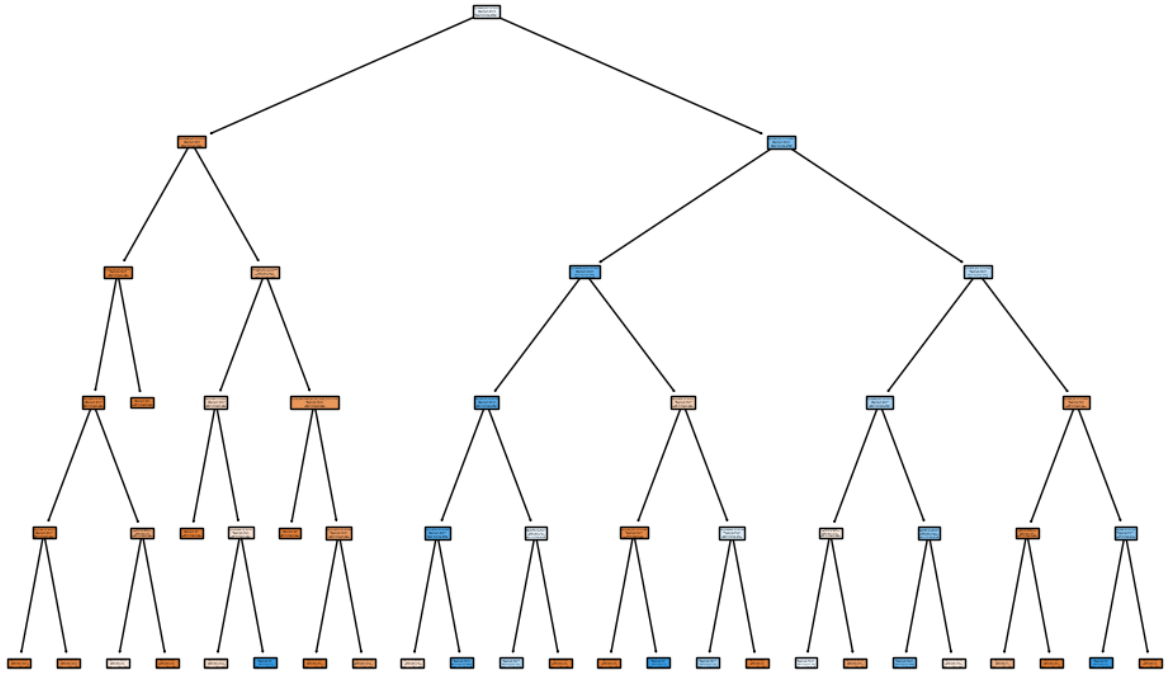

Figure 4.

Nevertheless, the confusion matrix for the best decision tree gave us an F1 score of 79%, which was better than the log model by 0.03%. The recall of the confirmed values stayed the same, while the rest of the metrics slightly improved as can be seen in *Table 3.*

	precision	recall	f1-score	support
CANDIDATE	0.86	0.69	0.76	585
CONFIRMED	0.75	0.89	0.81	605
accuracy			0.79	1190
macro avg	0.80	0.79	0.79	1190
weighted avg	0.80	0.79	0.79	1190

Table 3.

3.4 Random Forest

As explained above, this ensemble method was chosen in order to avoid the decision tree's tendency to overfit. The fine-tuning algorithm kept the log loss criterion but changed the maximum depth to 10. However, this is not a fixed result as we saw during our analysis. The F1 score improved by 0.3, and similar improvement was observed in all the other metrics apart from the recall of the confirmed observations, which was kept unchanged as can be seen in *Table 4.*

	precision	recall	f1-score	support
CANDIDATE	0.87	0.76	0.81	585
CONFIRMED	0.79	0.89	0.84	605
accuracy			0.82	1190
macro avg	0.83	0.82	0.82	1190
weighted avg	0.83	0.82	0.82	1190

Table 4.

Looking at the variable importance plot in *Figure 5.*, it is clear that the most important variable appears to be *koi_model_snr.* It is very interesting to note that this variable also stood out in our pairwise scatterplot in *Figure 1.*, as it was the one whose colours were most easily distinguishable.

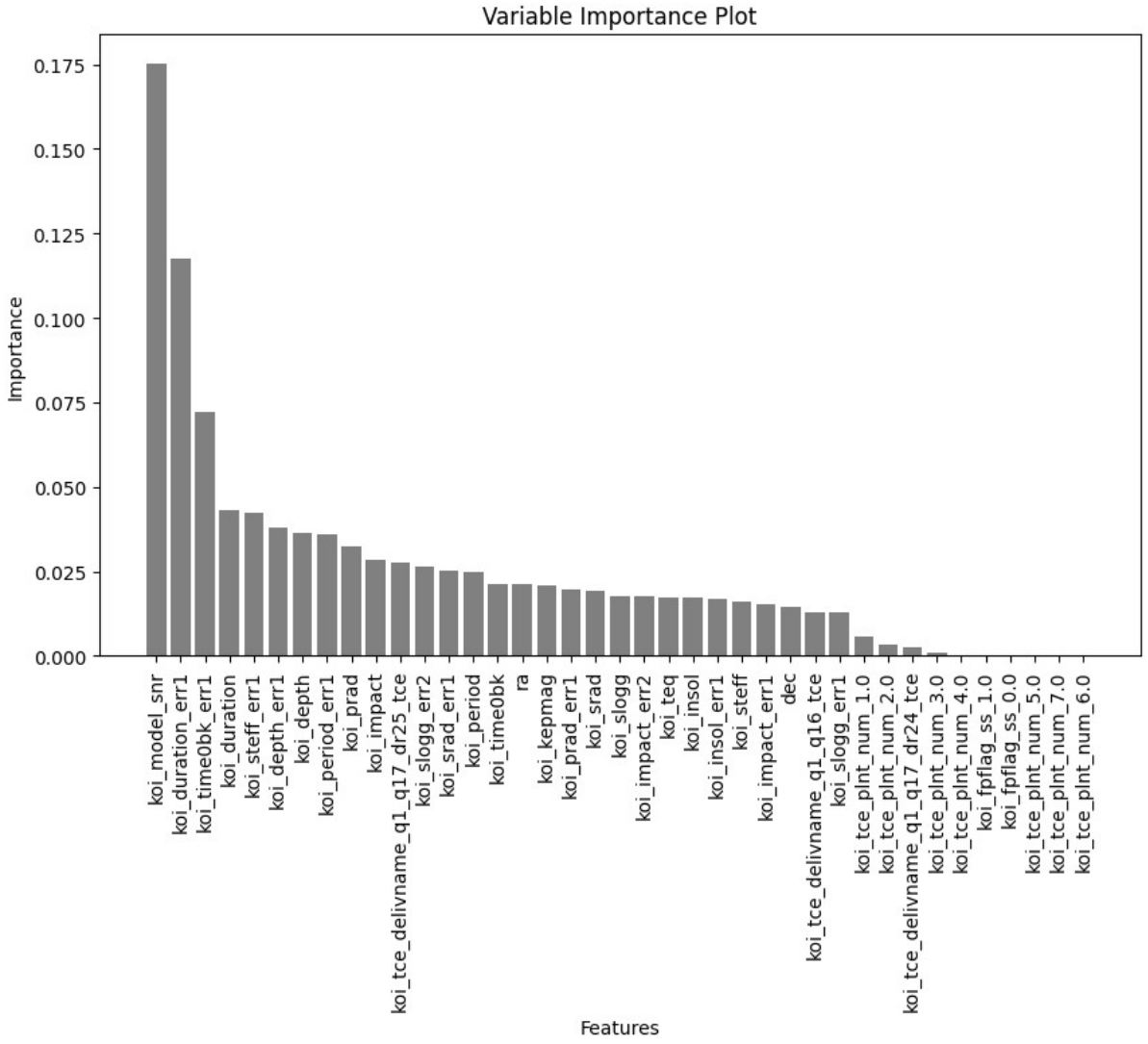

Figure 5.

3.5 Support Vector Machines

The last algorithm we tried before passing on to Neural Networks was the Support Vector Machine algorithm. SVM based classification works by finding the best possible decision boundary between the different classes. Since this algorithm can work with a large number of dimensions, said boundary is commonly a hyperplane, and it is determined by maximising the margin, which is the distance between the hyperplane and the nearest data points of each class.

When the given data is not linearly separable SVM can use different kernels to transform the feature space into a higher-dimensional one where the data may become linearly separable. We do not expect our data to be linearly separable so we decided to test multiple kernels in order to achieve the best parameters configuration for this algorithm.

After applying cross validation among different kernel configurations we concluded that the best option is to use a Radial Basis Function (RBF) with a gamma factor of 0.1. This factor determines the smoothness of the boundary layer.

This model has unexpectedly produced a lower F1 score at only 74%, dropping a significant 8% from the random forest model. There can be various reasons behind this, such as hyperparameter tuning or feature representation. For example, the features in the dataset may not exhibit clear linear separability, so SVM may struggle to find an optimal decision boundary. On the other hand, Random Forest models are more flexible and can handle complex relationships between features, leading to better performance.

	precision	recall	f1-score	support
CANDIDATE	0.87	0.55	0.68	585
CONFIRMED	0.68	0.92	0.78	605
accuracy			0.74	1190
macro avg	0.78	0.74	0.73	1190
weighted avg	0.77	0.74	0.73	1190

Table 5.

3.6 Neural Networks

Lastly, we wanted to try using a Neural Network as this is a state of the art ML model that is known to perform very well when dealing with complex functions. In order to find a good NN architecture we have conducted several tests where we have modified the number of layers, the amount of neurons per layer, the activation functions used, and the metrics that are optimised during the learning process.

We have restricted ourselves to use only the *Dense* layer type since considering the nature of our variables we can not find any argument to use more complex structures like *Convolutional* or *Recurrent* neural network layers.

The basic structure that is needed for our problem is to have an input layer with input dimension equal to the number of variables followed by several intermediate layers, and lastly a final dense layer with just one neuron that will give as an output a value between 0 and 1 representing the confidence the NN has of each example having a positive or negative outcome.

One very important aspect to consider when building a neural network is the activation function used on each layer. For classification tasks it is recommended to use a *sigmoid* function on the last layer but the rest of the layers can use other functions like, for example, *Relu* or *tanh*.

Figure 6. shows the architecture that we found produced the best results. It consists of an input layer, five hidden layers, and the single neuron output layer. For all layers except the last one, *tanh* activation function has been used.

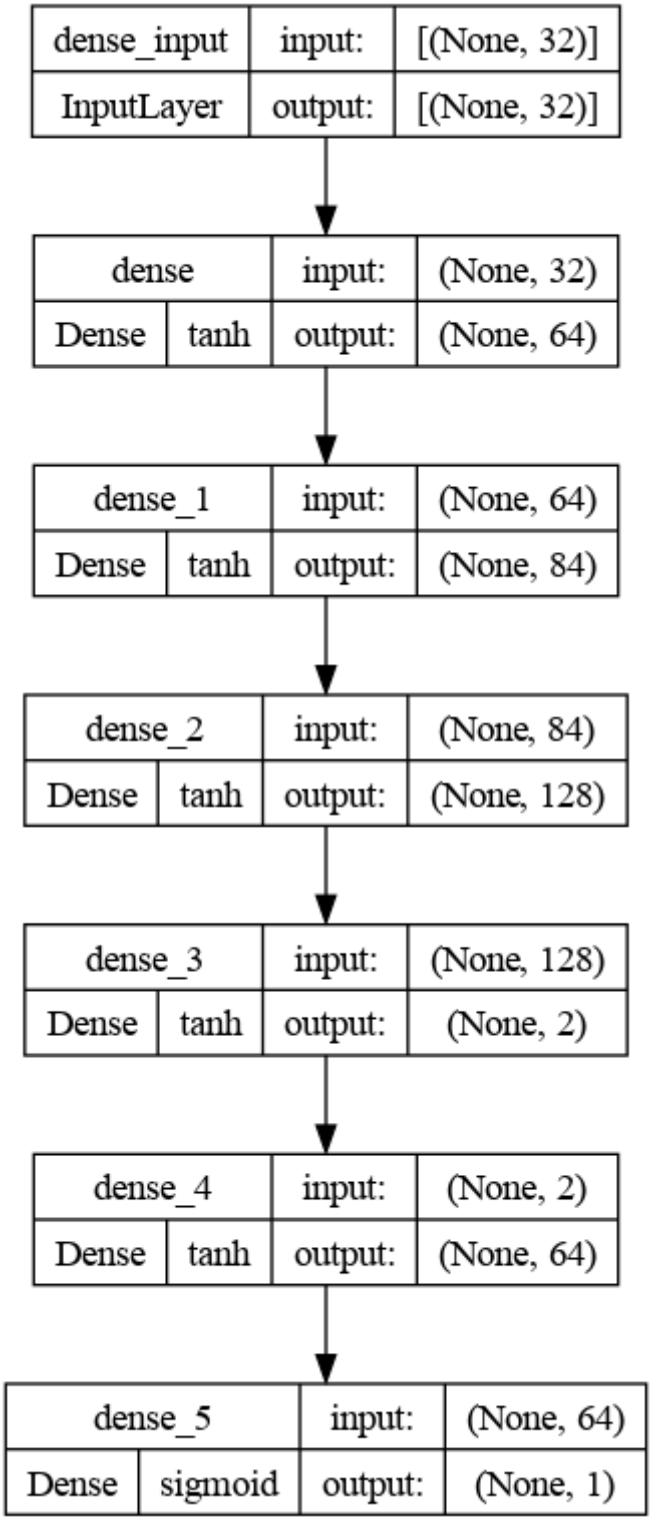

Figure 6.

The results obtained when running the above NN on the test dataset are shown in *Table 6*. We can see that this model achieved a slight improvement in performance (+0.01) with respect to the random forest results becoming the best performing model so far.

	precision	recall	f1-score	support
0	0.82	0.83	0.83	585
1	0.84	0.82	0.83	605
accuracy			0.83	1190
macro avg	0.83	0.83	0.83	1190
weighted avg	0.83	0.83	0.83	1190

Table 6.

4 Validation Results

4.1 Final model selection

After having trained, fine-tuned, and evaluated several models we are in a good position to decide what would be the best model to use as a final model. *Table 7.* shows the best scores achieved for each kind of model that we have trained sorted by their final F1-Score.
This results seem to indicate that the best option would be to use a Neural Network since the performance is the highest, however, considering that the difference is quite small compared to the second best (Random Forest) and that neural networks have a tendency to overfit, together with the fact that neural networks are much more complex and heavier than random forests, we have decided to test both of these models with the validation datasets to ensure that we select the model that generalises the best.

Model	F1-Score (accuracy)
Neural Network	83 %
Random Forest	82 %
Decision Trees	79 %
Logistic Regression	76 %
Support Vector Machines	74 %

Table 7.

The validation results for the random forest are shown in *Table 8* while the results for the NN are in *Table 9*. By comparing the two tables one can conclude that both models are equally good for solving the problem at hand and have a very good generalisation on unseen data. With this in mind we incline ourselves to recommend picking the random forest since it is a simpler model.

	precision	recall	f1-score	support
CANDIDATE	0.83	0.82	0.82	187
CONFIRMED	0.87	0.87	0.87	254
accuracy			0.85	441
macro avg	0.85	0.85	0.85	441
weighted avg	0.85	0.85	0.85	441

Table 8.

	precision	recall	f1-score	support
0	0.80	0.85	0.83	187
1	0.88	0.85	0.87	254
accuracy			0.85	441
macro avg	0.84	0.85	0.85	441
weighted avg	0.85	0.85	0.85	441

Table 9.

In *Figure 7.* we can see the confusion matrix of the Random Forest model, which shows the TP, TN, FP, and FN. Overall the model looks good, as the majority of the observations are placed in the diagonal. That means that the FP and FN take the lowest values when compared to the previous models.

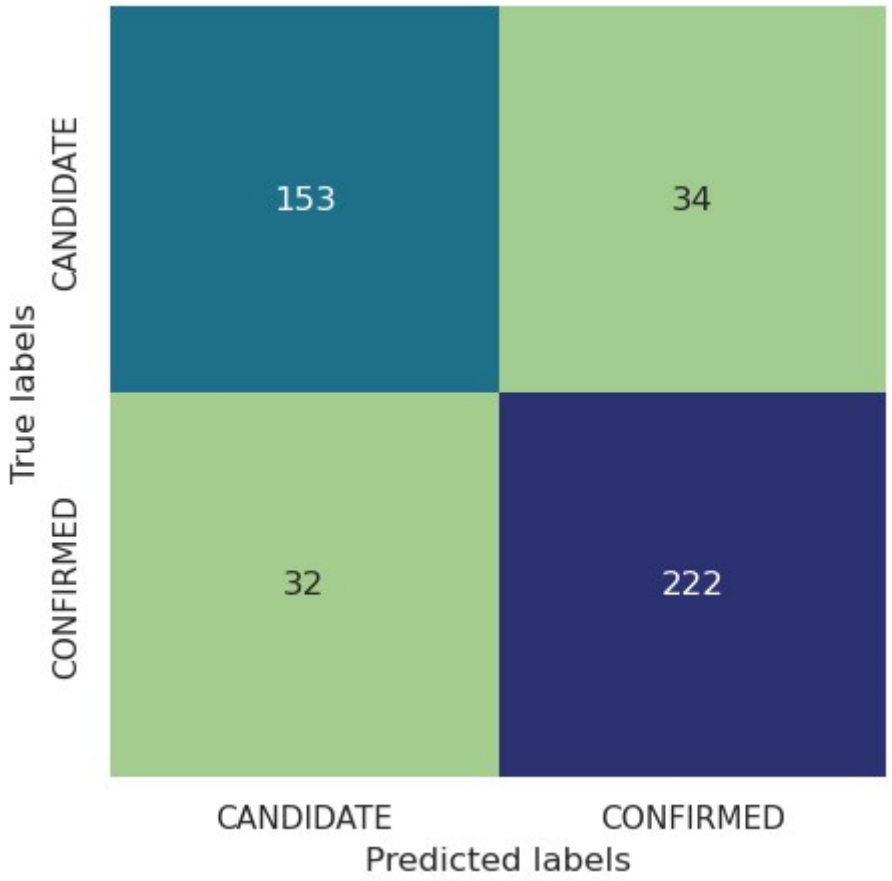

Figure 7.

5 Conclusions and Future Work

Through this analysis we have explored a very interesting dataset in order to predict whether the light sources captured by the Kepler telescope were exoplanets or not. We have learned a lot about how Kepler metrics work and the meaning behind them, and how they are all combined in order to calculate distances, degrees, as well as flags in the most efficient way. Apart from that, we had the opportunity to put in practice some of the methodologies seen during the lectures, and, through trial and error, get the best results.
Further efforts could be made in order to find a more complex Neural Network structure that is capable of outperforming the Random Forest, but this will translate in a heavier and slower model and the improvements may not be worth the costs.
Going in a different direction in the future, we could analyse this dataset without having removed the False Positives from our target variable, making it a multi-class classification. The multi-class classification algorithms to be used are the ones specifically designed to handle problems with multiple classes, such as Multinomial Logistic Regression, Random Forests, or Naive Bayes.

References

1. *Data columns in Kepler objects of interest table* (2021) *Data columns in Kepler Objects of Interest Table*. Available at: https://exoplanetarchive.ipac.caltech.edu/docs/API_kepcandidate_columns.html#FWDa (Accessed: 28 May 2023).