Course Final Project

Supervised Machine Learning: Classification
IBM Machine Learning Professional Certificate

Emily Kendall

Contents

1 Project Summary and Main Objectives

In this work we analyse the Kepler Objects of Interest (KOI) dataset, made available through the NASA Exoplanet Archive (Christiansen et al., 2025). This is a catalogue in which each row corresponds to a star which was observed by the Kepler space telescope to show signs of periodic dimming, possibly attributable to the transit of an orbiting exoplanet.

The detection of exoplanets is a notoriously difficult problem in astronomy, and after further investigation, many signals initially thought to be due to exoplanets are subsequently discovered to be due to other astrophysical phenomena or noise Almenara et al. (2009). It is therefore of interest to the scientific community to develop tools which are able to more readily distinguish between plausible candidates and spurious signals, so that time and resources may be dedicated to follow-up studies of only the most promising candidates.

Our goal in this work is to utilise machine learning tools to find underlying patterns in the KOI data to enable rapid and reliable identification of signals likely to be due to true exoplanet transits. Thanks to extensive follow-up studies and statistical analysis, the KOI dataset contains labels for 9564 observations. Where an observation has been found to correspond to a true exoplanet, the entry is labelled “CONFIRMED”. Conversely, signals which have been found to be attributable to other astrophysical phenomena or instrumental noise are labelled ”FALSE POSITIVE”. Any observations which have yet to be conclusively categorised are labelled ”CANDIDATE”.

We will use a subset of the labelled data to train various classification models to distinguish between the classes ”CONFIRMED” (Class 1) and ”FALSE POSITIVE” (Class 0). We will evaluate the performance of each our learned models on a hold-out set. Finally, we will apply our models to the remaining ”CANDIDATE” observations to generate a list of predictions of the most promising signals. This list may then be used by research teams to guide the allocation of resources for follow-up studies. Our analysis may also help to identify features which are most important in the identification of exoplanets, accelerating future research in this field.

2 Description of the KOI Data

The entire KOI cumulative dataset contains 9564 labelled entries. Each row contains a target value (‘koi_disposition’) as well as 154 corresponding features. The target value represents whether the given set of features corresponds to a true exoplanet (‘CONFIRMED’), or not (‘FALSE POSITIVE’). Some rows have yet to be classified, and these are labelled ’CANDIDATE’.

The 154 features include a range of measured properties of stars for which the Kepler Space Telescope observed periodic brightness changes. However, many of the features in the raw dataset are unrelated to whether or not the star hosts an exoplanet, such as unique identification numbers and sky location. Other features may have limited impact on the target variable, or exhibit significant mutlicolinearities. We therefore restrict our attention to the features described in Table 1.

The selection of these features is based upon domain expertise and knowledge of the data processing pipeline. Notably, we omit from our table the 'koi_score' feature, as this value is often derived from the same vetting process that determines the target label itself ('koi_disposition'). Therefore, while very predictive, it is not independent of the target label and is derived through a lengthy statistical analysis which we wish to obviate through our machine learning approach.

Feature name	**Type**	**Feature description**	**Relevance to target**
`koi_model_snr`	float64	Transit signal-to-noise ratio	High SNR indicates a robust signal, increasing confidence that this is likely due to an astrophysical source.
`koi_fpflag_nt`	int64	Non-transit-like signature flag	If set, the observed signal does not strongly resemble a typical transit light curve.
`koi_fpflag_ss`	int64	Stellar eclipse signature flag	Detects signs of an eclipsing binary star (i.e. eclipsing object unlikely to be an exoplanet).
`koi_fpflag_co`	int64	Centroid offset during transit	If brightness centre shifts during transit, the dimming may be due to a nearby star.
`koi_fpflag_ec`	int64	Ephemeris match with known binary	Indicates that signal may be due to a known eclipsing binary.
`koi_depth`	float64	Depth of transit in ppm	Extremely deep transits are more indicative of stars than planets.
`koi_duration`	float64	Duration of transit (hours)	Uncharacteristically long or short durations may signal false detections.
`koi_period`	float64	Orbital period (days)	Certain periods may be associated with instrument artifacts or transient phenomena.
`koi_prad`	float64	Planet radius	Smaller bodies are likely to be exoplanets while larger objects are typically stellar.
`koi_insol`	float64	Insolation flux	Anomalously high or low values may indicate physical implausibility.
`koi_steff`	float64	Stellar effective temperature	Affects light curve features. Certain stellar types are more prone to noise.
`koi_srad`	float64	Stellar radius	Essential for interpreting transit depth and inferring planet size.

Table 1: Descriptions of selected KOI features, their data types, and relevance to predicting `koi_disposition`.

Using this subset of features, we will develop a series of machine learning models to predict whether a given example corresponds to a true exoplanet or a spurious signal.

3 Data Exploration, Cleaning, and Feature Engineering

As a first step in our data cleaning process, we drop all rows in which any of the features listed in Table 1 are null or missing. The resulting complete dataset has a distribution of target values as shown in Table 2.

We note that, while there are more examples in Class 0 ('FALSE POSITIVE') than Class 1 ('CONFIRMED'), the distribution is not extremely imbalanced, and we are therefore unlikely to need to make use of re-weighting, oversampling, or downsampling to correct for underrepresentation of Class 1.

We now separate the entire database into two subsets. The first subset ('koi_tt'), which will be used for model training and validation, contains all entries with target label 'FALSE POSITIVE' or 'CONFIRMED'.

koi_disposition	**Count**
FALSE POSITIVE	4582
CONFIRMED	2743
CANDIDATE	1876

Table 2: Distribution of KOI Dispositions

The second subset ('koi_p'), upon which we will apply our machine learning pipeline to generate our final recommendations, contains all entries with target label 'CANDIDATE'.

Because we will be making use of models which are sensitive to distances between points in feature space, it is important that we scale our features. The integer features in Table 1 are already one-hot encoded, so we apply scaling only to the float columns. In particular, we use standard scaler to fit and transform the float features in 'koi_tt'. We then use the same transformation (without re-fitting) to transform all the float features in 'koi_p'. This ensures that the same scaling is used on both the training and real application data.

After scaling, we visually analyse the float features using pair plots of a small subsample of the data. The results are shown in Figures 1 and 2 (we have separated this into two separate pair plots for ease of visual analysis). While the two target categories are not easily separable along most axes, it is clear that the distributions associated with Class 0 ('FALSE POSITIVE') are substantially broader than the distributions of Class 1 ('CONFIRMED'). This suggests that any given example belonging to Class 0 will likely have at least one feature which lies outside the distribution of typical Class 1 values, and will therefore be distinguishable by our models.

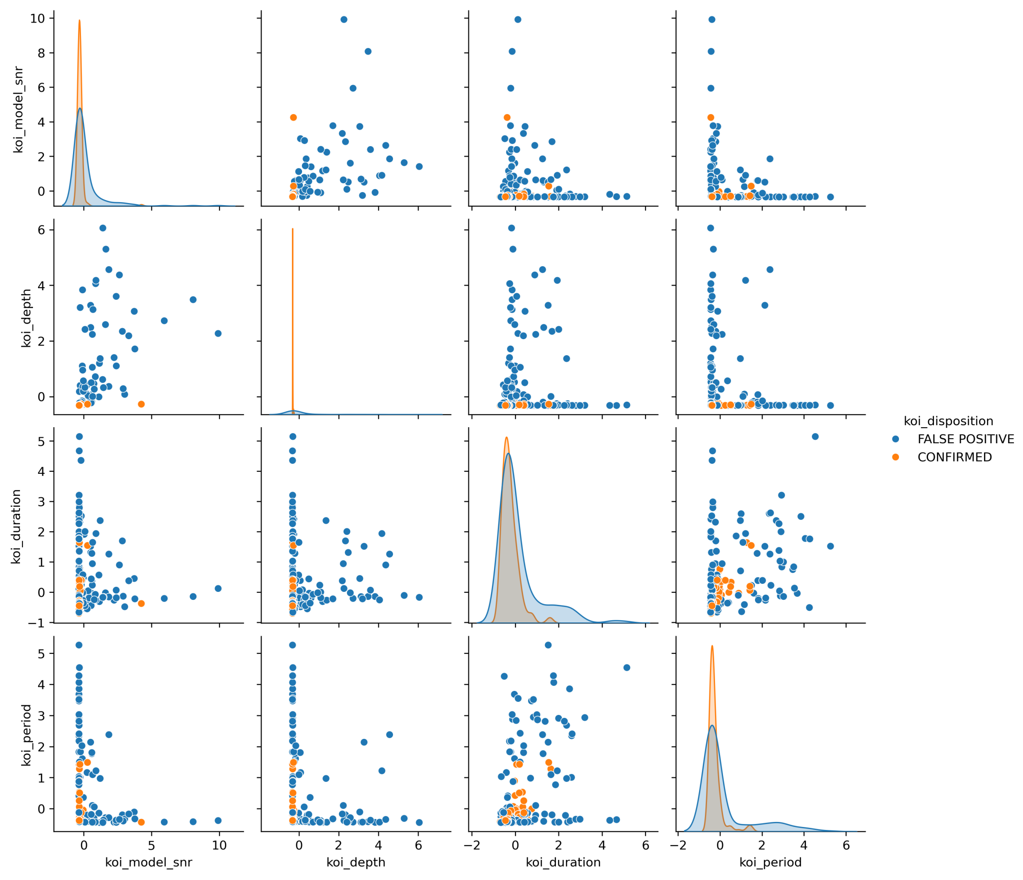

Figure 1: Pair plot of selected float columns, coloured by target label.

In Figure 3, we illustrate the Pearson correlation coefficient for each pair of float-valued features. while there are reasonable correlations between `'koi_depth'` and `'koi_model_snr'`, as well as between `'koi_insol'` and `'koi_srad'`, these are not so high to cause concerns of excessive multicolinearity. We therefore conclude that the chosen features are suitable for this analysis. We do not create any composite features at this stage, but will discuss in Section 5 other possible feature choices.

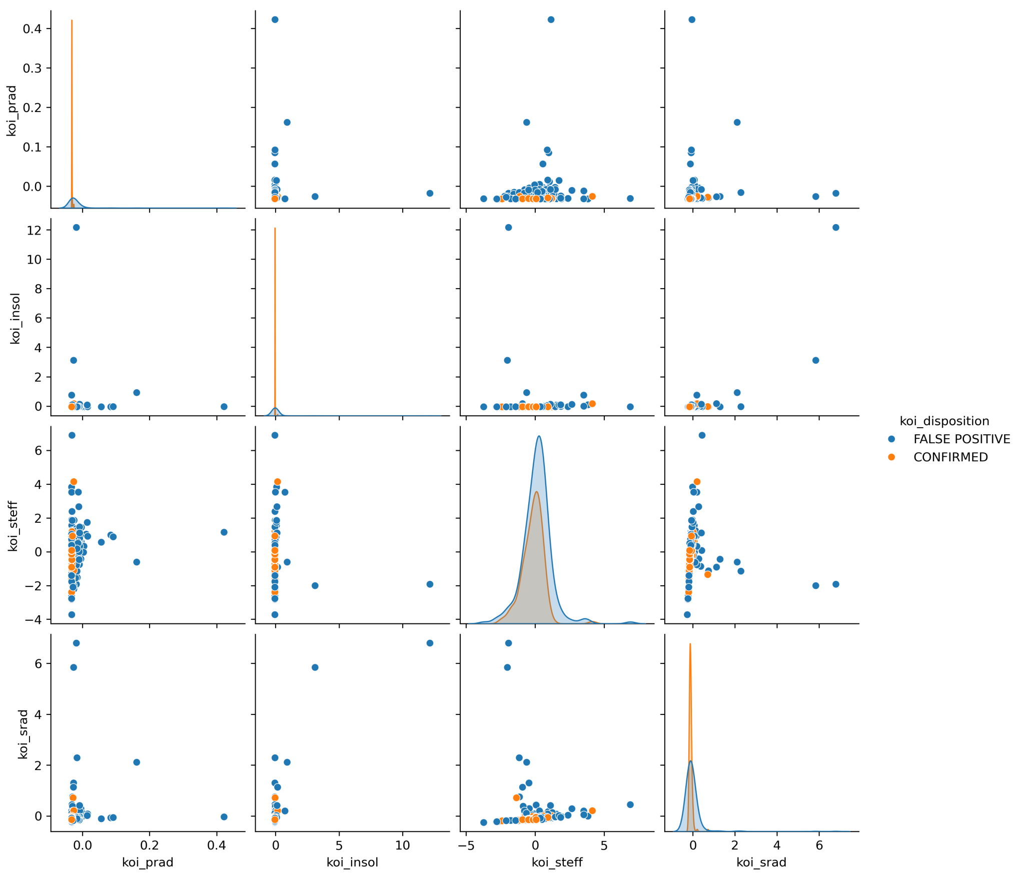

Figure 2: Pair plot of selected float columns, coloured by target label.

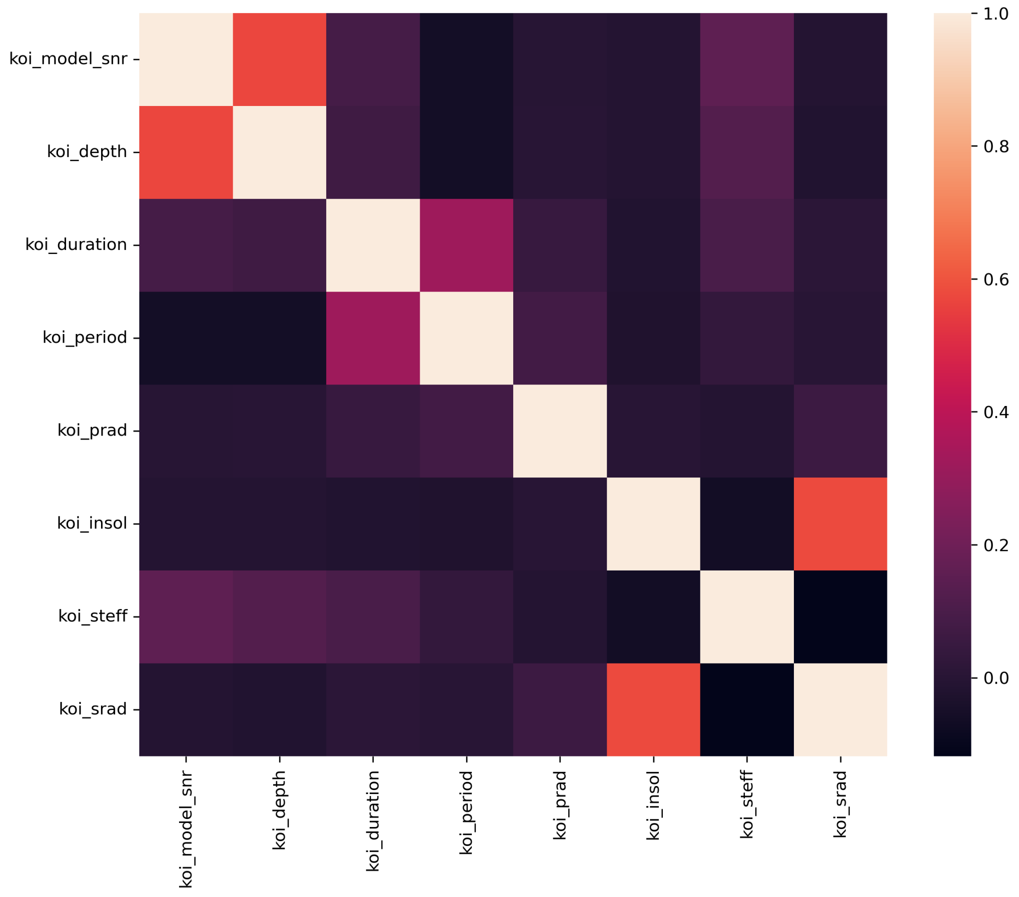

Figure 3: Pearson correlations for float-valued features.

Finally, before proceeding to build our machine learning models, we encode our target with integer values (0 = 'FALSE POSITIVE', 1 = 'CONFIRMED'), and separate the 'koi_tt' table into train (70%) and test (30%) sets, stratifying on the target to ensure similar distributions in both sets. In the following section, we train a variety of classifiers on this data.

4 Classifiers

4.1 Logistic Regression

We begin by training a simple logistic regression model. We use `GridSearchCV` to test hyperparameters for this model, and find the following to be optimal in terms of F1 score: `C=10, class_weight=None, penalty='l1', solver='liblinear'`. When we evaluate the performance of this model on the hold-out set, we obtain the following metrics:

Metric	**Class 0**	**Class 1**
Precision	0.986	0.915
Recall	0.945	0.977
F1 Score	0.965	0.945
Accuracy	0.957	

Table 3: Classification performance metrics for Class 0 and Class 1, rounded to 3 decimal places.

We see that the simple logistic regression model already performs quite well on the unseen data. We note, however, that precision for Class 1 is lower than we would like, as a key objective here is to minimise false positives. This is because there is significant cost associated with assigning research resources to follow-up studies of spurious signals.

Given that logistic regression is an easily interpretable model, we can use this to gain insight into the features most important for determining the target value. In Figure 4, we plot the values of the coefficients corresponding to each of the features.

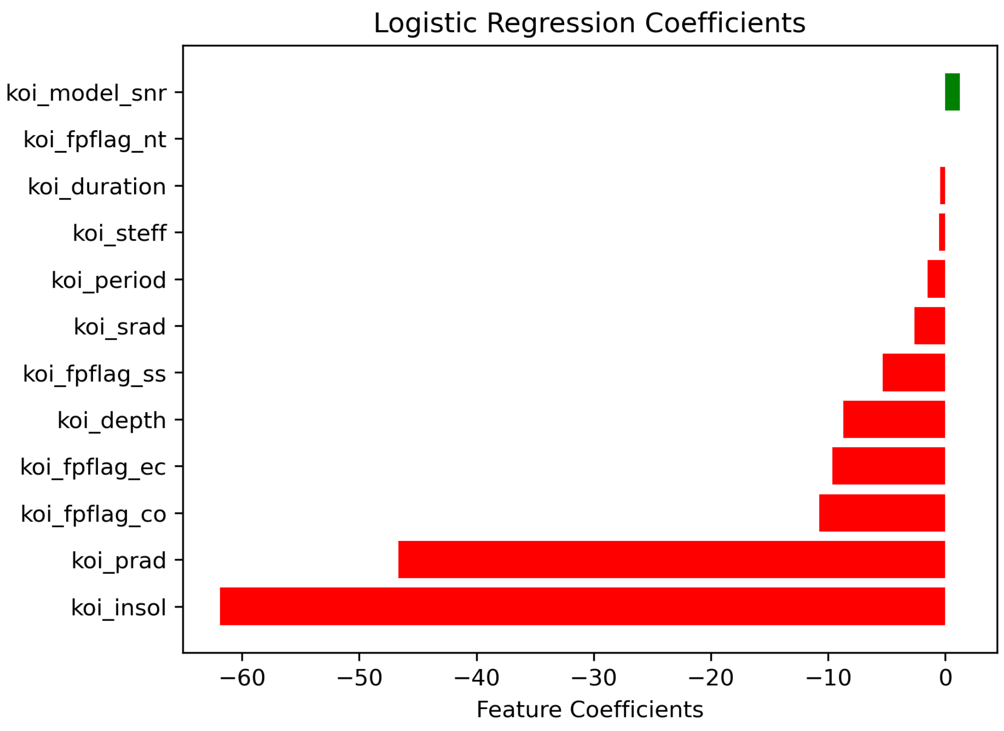

Figure 4: Pair plot of selected float columns, coloured by target label.

We see that `'koi_insol'` and `'koi_prad'` are weighted most heavily in the logistic regression. This is intuitively sensible, since `'koi_prad'` represents an estimate of the size of the candidate exoplanet, and values which are similar to the sizes of known exoplanets are much more likely. By contrast, values which are too high are suggestive of binary star systems, while values which are too low are suggestive of noise or measurement error. Furthermore, `‘koi_insol’` is an estimate of the insolation flux of the candidate, and is important in determining whether an object's orbit and stellar environment are physically plausible.

4.2 K Nearest Neighbours Classifier

As an alternative to the logistic regressor, we train a K neighbours classifier on the same data. We use `GridSearchCV` to test hyperparameters for this model, and find the following to give optimal F1 score performance: `n_neighbors=3, weights=‘uniform’, p=1`. When we evaluate the performance of this model on the same hold-out set as above, we obtain the following metrics:

Metric	**Class 0**	**Class 1**
Precision	0.993	0.988
Recall	0.993	0.988
F1 Score	0.993	0.988
Accuracy	0.991	

Table 4: Classification metrics for KNN using L1 distance measure (rounded to 3 decimal places).

Clearly, KNN is even better suited to this problem than logistic regression, with greater overall accuracy. Importantly, the KNN model achieves very high precision in Class 1, which is the most important metric in this case.

4.3 Support Vector Classifier

We also train a support vector classifier on the same data. We use `GridSearchCV` to test hyperparameters for this model, and find the following to give optimal F1 score performance: `‘C’: 50, ‘gamma’: ‘scale’, ‘kernel’: ‘rbf’`. When we evaluate the performance of this model on the same hold-out set as the previous two models, we obtain the following metrics:

Metric	**Class 0**	**Class 1**
Precision	0.997	0.987
Recall	0.992	0.995
F1 Score	0.995	0.991
Accuracy	0.993	

Table 5: Classification metrics for SVC with L1 distance (rounded to 3 decimal places).

The performance of the SVM model is very similar to that of the KNN model, with almost identical values of Class 1 precision and overall accuracy.

4.4 Random Forest Classifier

Finally, we train a random forest classifier on our data. As before, we use `GridSearchCV` to test hyperparameters, and find the following to give optimal F1 score performance: `criterion=‘gini’, max_depth=15, max_features=5, n_estimators=25`. The performance of this model on the test set is as follows:

Metric	**Class 0**	**Class 1**
Precision	0.996	0.994
Recall	0.996	0.994
F1 Score	0.996	0.994
Accuracy	0.995	

Table 6: Classification metrics for Random Forest with 25 estimators (rounded to 3 decimal places).

As with the KNN and SVM models, the random forest model performs exceptionally well on the test data. This model achieves the highest performance in both Class 1 precision and overall accuracy. Therefore, we choose this model as our preferred classifier for this task.

In order to gain insight into the random forest model, we emulate the model with a much simpler global surrogate. We choose a single decision tree classifier with `max_depth = 5, max_features = 5` as the surrogate. When trained to reproduce the predictions of the random forest model on the test set, we find that the surrogate achieves an accuracy of 0.996, suggesting that this much simpler decision tree model is able to almost perfectly emulate the more complex model. The advantage of the simpler model is the increased interpretability. We illustrate the surrogate decision tree in Figure 5. We see that `'koi_prad'` again appears as a significant predictor of target class, with the decision tree performing the first data split on this feature.

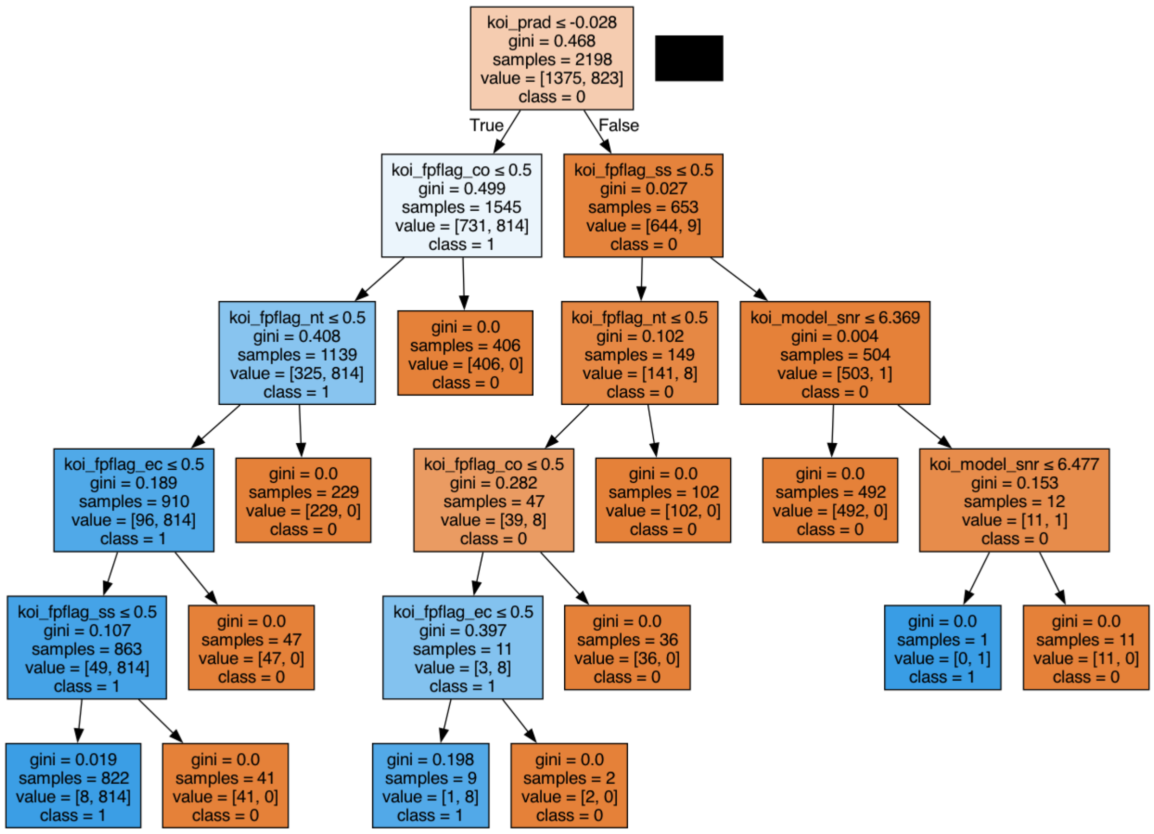

Figure 5: Global surrogate for the random forest model.

5 Key Findings and Insights

Overall, we find that the classification of KOI data into true exoplanets or spurious signals is a task well suited to a variety of machine learning approaches. We built a variety of classifier models to address this task, and all were able to achieve classification accuracy of over 95%. The best of our models (random forest) was able to achieve an accuracy of 99.5% and a precision on the positive class of 99.4%. We chose the random forest classifier as our preferred final model due to this remarkably good precision. Because there is a large cost associated with follow-up studies, we want to minimise false positives as far as possible, making precision the most appropriate metric by which to judge these models. A side-by-side comparison of model performance across various metrics is given in Figure 6.

The objective of building this classifier was to then apply it to the as-yet unclassified 'CANDIDATE' data from the KOI dataset. When we do so, we find the following class split: Class 1: 1513, Class 0: 363. We can use this to generate a list of likely candidates for follow-up studies. For example, the Kepler ID numbers corresponding to the first few promising candidates are [10811496, 11818800, 11918099, 9579641, 3115833]. Given the quality of the predictions made by the random forest model, we anticipate that this will be a very useful tool for researchers, enabling them to assign resources more confidently. A further advantage of the random forest classifier is that we are able to retrieve probabilities for each classification, and may therefore choose a relatively high probability threshold for follow-up studies. We note also that this random forest model can be emulated remarkably well by a simple single decision tree model, which is very useful in increasing interpretability.

As our models already achieve excellent performance, there is little room for improvement. Indeed, we analysed whether combining our three best performing models into a voting classifier would further increase performance, but found that performance in this case did not exceed that of the random forest. However, further studies in which alternative combinations of features from the raw dataset are used may yield even better performance and insight.

In future it may also be desirable to build models which make use of only raw data (i.e. the float-valued columns in Table 1), without recourse to statistically derived flags (i.e. the integer-valued columns in Table 1). This would be beneficial in that minimal processing of the direct observational data would be required before applying the classifier, further streamlining the processing pipeline.

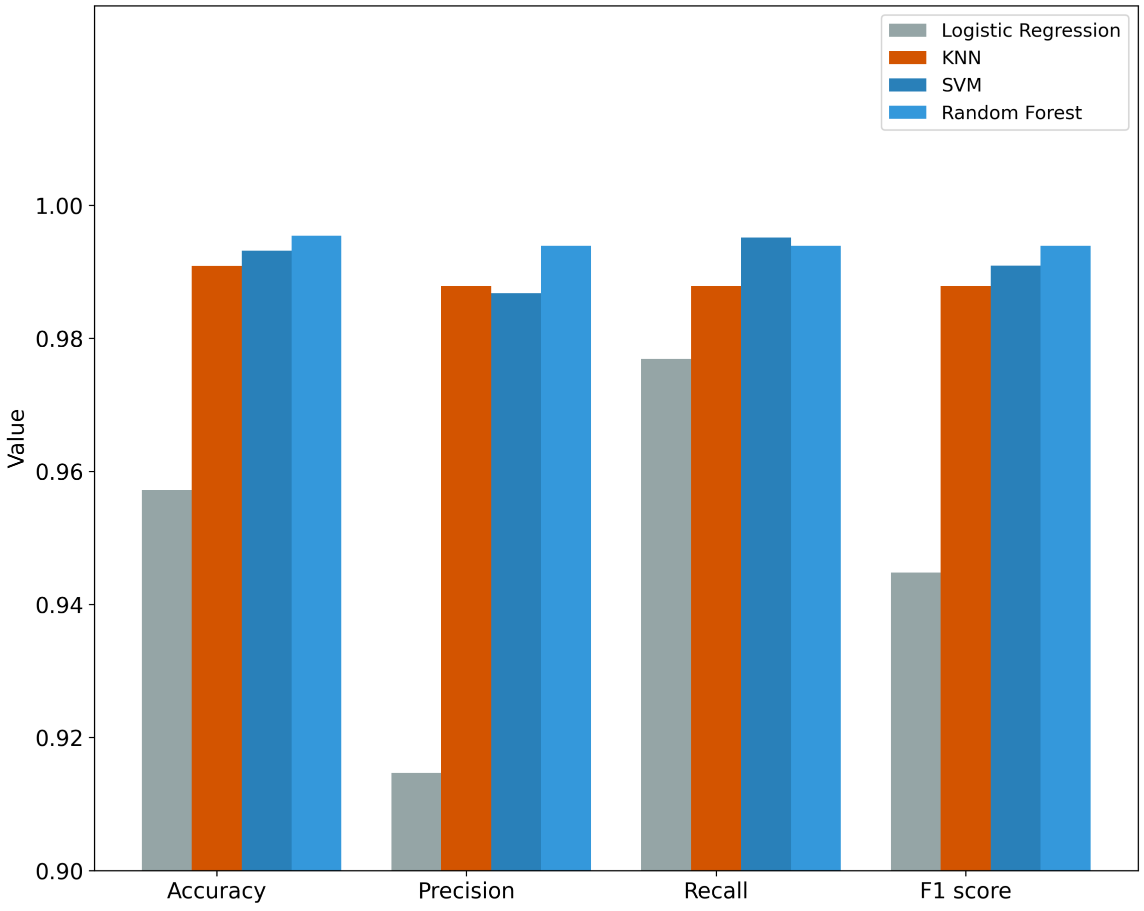

Figure 6: Comparison of the four models tested. All scores are relative to the positive class ('CONFIRMED').

References

Almenara, J. M., Deeg, H. J., Aigrain, S., et al. 2009, , 506, 337, doi: 10.1051/0004-6361/200911926

Christiansen, J. L., McElroy, D. L., Harbut, M., et al. 2025, arXiv e-prints, arXiv:2506.03299, doi: 10.48550/arXiv.2506.03299