Overview

I am not quite convinced by the logic of this exercise. It's the most interesting data we've had to work with (to me), but nobody should give any weight to my results. The Kepler people are experts on the measurements made, which involve detection of sometimes very weak signals against a universe of background noise. Even with their expertise, the assignments are only a best estimate - it could be quite some time before there is a real test of the predictions. Given that, and my near ignorance of both astronomy and ML, I can't say which model is best, only which comes closest to matching the Kepler conclusions in terms of the accuracy parameter. For analyses not including Kepler flags (see below), Random Forest methods have so far yielded the highest scores.

Data cleanup

Models were tested against the koi_pdisposition column, after checking to see that it used only two values (FALSE POSITIVE or CANDIDATE). koi_disposition has additional categories and so was dropped. All error columns were dropped, although obviously in real life they are important. NaN and infinite values were removed. Non-numeric ID columns were dropped; the kepids were kept in case of future interest (didn't materialize).

Also noted were four "flag" columns used by the Kepler group to identify problematic values. I suspected that including these might increase model "accuracy", but only because the data already contained the results of an analysis. Therefore some models were tested both with and without these columns (it might be interesting to use only these, but that wasn't tried).

The grid search method was by far the most time-consuming, so in order to allow multiple runs the data was further sifted. Rows 5001-9563 were dropped, cutting the dataset approximately in half, and seven more columns removed. Not sure my reasons make astronomical sense, but using the random tree rankings as a guide, I guessed that a star's location in the sky (my column names 'sky_location_declination', sky_location_right_asc') and magnitude ('stellar_magnitude') might affect detectability of any planets, but not their actual existence. Several stellar characteristics were dropped because they were low in the Random Tree rankings ('stellar_surf_gravity', 'stellar_photosph_rad', 'stellar_eff_temp'). 'time_first_trans_detected' might be meaningful if details of the data gathering were known (e.g., were there special problems at that time?) but is not really interpretable to outsiders, so it was also dropped.

KNeighborsClassifier (file exoplanet_search_KNN)

This model was run twice, with or without the "flag" columns. As expected, including them increased "accuracy", but this reasoning is a little circular. Optimal "k" values were selected from graphs.

> With flags: for k in range(1, 40, 2): ~best k=15, Test Acc: 0.987
> Without flags: for k in range(1, 50, 2): ~best k=17, Test Acc: 0.761

Sequential Model (file exoplanet_search_deep)

This model was run varying a number of parameters; I would say the only significant one was whether the "flag" columns were included or not. Without the flagged columns, accuracy was a bit higher than for the KNeighborsClassifier.

Run	Flags included?	Intermed.layers	Units	Epochs	Starting state	Norm.	Loss	Accuracy
1	Yes	2	100	60	1	Adam	0.0544	0.9886
2	No	2	100	60	1	Adam	0.4311	0.8121
3	No	3	100	60	1	Adam	0.4283	0.8113
4	No	3	100	60	18	Adam	0.4340	0.8091
5	No	3	200	60	1	Adam	0.4426	0.8117
6	No	3	100	120	1	Adam	0.4818	0.8104
7	No	3	100	60	1	Nadam	0.4285	0.8186
8	No	3	100	60	1	Adamax	0.4372	0.8121

Decision trees (file exoplanet_search_random_forest)

Just two runs; as above, including the flag columns gives a very high accuracy. Without them comparable to other models.

First run: random state=57, "flag" columns included. clf.score(X_test, y_test) 0.9843
Second run: random state=57, "flag" variables removed. clf.score(X_test, y_test) 0.7669

Random forest (file exoplanet_search_random_forest)

Run varying several parameters. Noted that when the flagged columns were included, they were always among the highest ranked (see file for details). "planet_radius" was high on the list in all cases, not surprising since planets are detected as they pass in front of their stars. Changes in stellar flux also rank high - seems predictable, a bigger change should be easier to detect.

Surprising here was that removing the flag columns had little effect when the "gini" criterion was used with all features, but reduced accuracy when either of these was changed. I don't pretend to understand this well enough to offer an explanation.

Run	Flags included?	Random state	Estimators	rf.score(X_test, y_test)	Criterion	max_features
1	Yes	57	200	0.9904	gini	n_features
2	No	57	200	1.0	gini	n_features
3	Yes	57	50	0.9895	gini	n_features
4	Yes	57	10	0.9891	gini	n_features
5	Yes	312	200	0.9882	gini	n_features
6	Yes	57	200	0.9904	entropy	n_features
7	Yes	57	200	0.9904	gini	auto
8	No	57	200	0.8386	entropy	n_features
9	No	57	200	0.8369	gini	auto

Grid search with Support Vector Classification (SVC) (file exoplanet_search_grid)

As mentioned above, used a reduced set of columns and rows (with no flagged values), because even so each run took several hours. Results are therefore not directly comparable with those of other methods. The main result from the first four runs is simply that C and gamma are not yet optimized.

Run	C	Gamma	Best C	Best gamma	Test Acc
1	[1, 5, 10]	[0.0001, 0.001, 0.01]	1	0.0001	0.752
2	[0.1, 0.5, 1]	[1e-05, 5e-05, 0.0001]	0.1	1e-05	0.771
3	[0.001, 0.05, 0.1]	[1e-06, 5e-06, 1e-05]	0.001	1e-06	0.793
4	[0.0001, 0.001, 0.01]	[1e-08, 1e-07, 1e-06]	0.01	1e-08	0.797