FYS-STK4155
Applied data analysis and machine learning

EXOPLANET CLASSIFICATION: A SEARCH FOR HABITABLE EXOPLANETS DISCOVERED BY THE KEPLER MISSION

Anna Eliassen, Aron Jansson Nordberg and Kristina Othelia Lunde Olsen
Department of Physics, University of Oslo, Norway
{annaeli, aronjn, koolsen}@astro.uio.no

All materials can be found on:
https://github.com/kristinaothelia/FYS-STK4155/tree/master/Project3

December 22, 2019

Abstract

In this project we have studied the data gathered from the NASA Kepler mission by applying machine learning classification methods. This mission was the first space mission to search for Earth-sized and smaller planets in the habitable zone of other stars in our neighborhood of the galaxy.

From the four machine learning methods we used, Logistic regression, Neural Network, Random Forest and XGBoost, we determined that the Random Forest method provided the best model. XGBoost may produced the best model evaluation metrics, but this was probably a result of overfitting. The Logistic regression method did not handle the data set as well as the other methods, and Neural Network classification gave good, but unlikely, predictions. On average, all methods predicted more object candidates to be exoplanets than not exoplanets, which were surprising considering that the confirmed training data consists of about twice as many objects not being exoplanets. If this is due to overfitting or other machine learning difficulties we have encountered, or simply because it is many exoplanets among the object candidates, are impossible to say with certainty. At the end of the project, we used our models to predict habitable object candidates with at least a 90% probability of the object actually being an exoplanet. To be categorized as an habitable exoplanet, that can support advanced life and liquid water, the exoplanet need a surface temperature $T \in [0, 50]^{\circ}$C. The Random Forest model predicted three exoplanets within this range.

Contents

1 Introduction

A hot topic in astrophysics is exoplanets. Aside from the question of extraterrestrial life, we wonder where this life may exist, and the answer is (most likely) exoplanets. What is out there, how many and most important, can we find a habitable second earth? Another hot topic in the scientific community is machine learning, which can be used for classification problems and data handling. Machine learning algorithms have the unique capability of learning and improving from experience, making exoplanet hunting easier.

One can train a machine learning to process large amounts of data, and train models that map our information about the measured data (predictors) to a probability estimate of there being an exoplanet among the candidates or not. This requires already existing datasets made by humans/satellites, and the results will have to be reviewed by scientists, but the computing power is far greater in a computer and it can perform way more heavy lifting than a team of scientists. Given the great success of machine learning in science and industry involving large amount of data, it is an invaluable tool in discovering exoplanets. Models will also improve with time, given more data and more advanced models.

In newer time, we gather most exoplanet data from satellites, like the Kepler space telescope. These satellites are advanced and can use astronomical techniques to produce data features specified for exoplanet identification by image processing. In this project we want to combine the NASA Kepler Objects of Interest table, exoplanet data, with machine learning methods to create a classification model for predicting true exoplanets. The methods we are going to use is Logistic regression, Neural Network, Random Forest and XGBoost, as well as GridSearchCV.

2 Background

An exoplanet, by definition any planet outside our solar system, is the target of interest for the Kepler mission. The data used in this project is gathered from a NASA mission, and we will therefor focus on exoplanet exploration done by NASA. Exoplanet Exploration Program (ExEP) is NASA's goals of exploring planetary systems around nearby stars and gain a deeper understanding of astrophysics. The main purposes of this program is to discover, learn and find habitable exoplanets, which includes searching for bio-signatures in the atmosphere of exoplanets [10]. Some information on the future program of ExEP can be found in the Conclusion section about The future of exoplanet exploration.

2.1 The Kepler space telescope

Before the Kepler mission, the most common discovered exoplanets were Jupiter-sizes exoplanets, or even larger hot-Jupiter's. These planets are unlikely to support life as we know it, so we are more interested in discovering Earth-like planets. That is where the Kepler mission is important. The objective of the mission were to discover Earth-sized exoplanets, that lies in or near the habitable zone of the host star.

The Kepler space telescope were operational until October 30, 2018, and the nine-year mission were considered a huge success. About half a million stars were observed, and 2662 exoplanets are confirmed, with more candidates still being evaluated [11]. The telescope were in the majority of the time pointed in a direction of the Milky Way that covers a small region of the Cygnus constellation [8]. The surprising result of the mission were that Kepler found more planets than host stars, in a huge diversity. See Appendix A for more Kepler telescope facts.

The Kepler mission has several scientific goals to extraterrestrial planetary systems. This is achieved by surveying a large sample of stars to:*

- Determine the percentage of terrestrial and larger planets that are in or near the habitable zone of a wide variety of stars
- Determine the distribution of sizes and shapes of the orbits of these planets
- Estimate how many planets there are in multiple-star systems
- Determine the variety of orbit sizes and planet reflectivities, sizes, masses and densities of short-period giant planets
- Identify additional members of each discovered planetary system using other techniques
- Determine the properties of those stars that harbor planetary systems.

**This bullet list is from https://www.nasa.gov/mission_pages/kepler/overview/index.html*

2.2 Exoplanet identification techniques

Discovering new exoplanets is a tricky business. There are several methods to do so, although it turns out that some methods, with today's technology, works best in theory. From figure 1, we can see that from the total 4093 confirmed exoplanets, we have 11 discovery methods. From these, *Transit* is clearly the most effective, followed by *Radial velocity*.

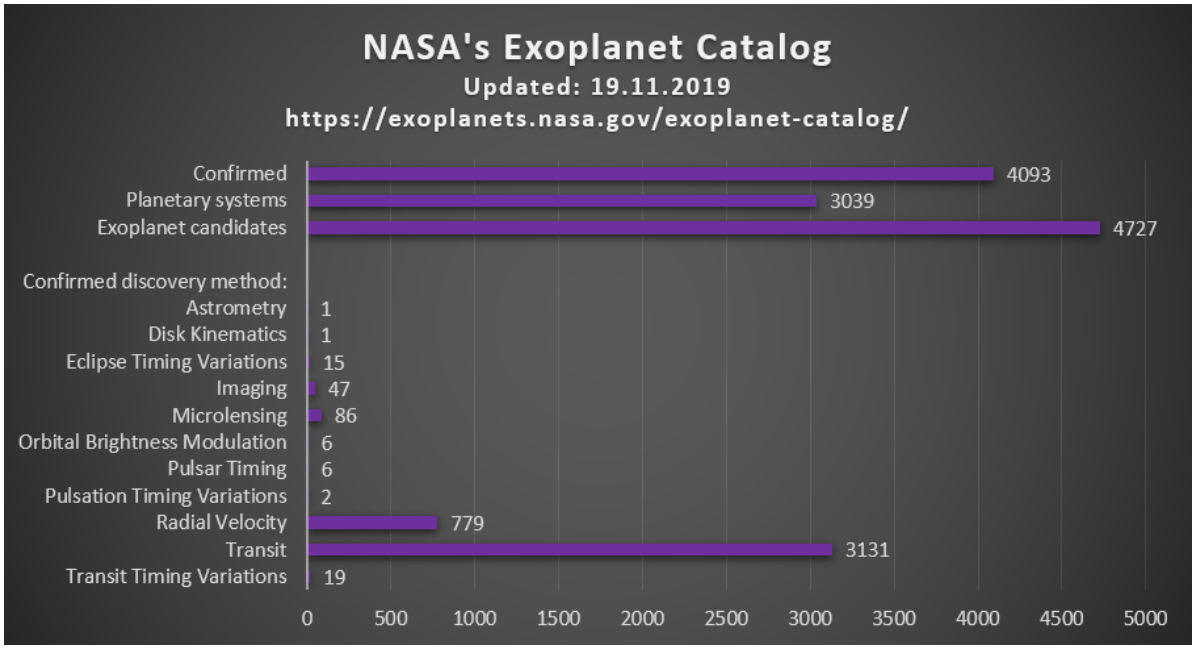

Figure 1: NASA's exoplanet catalog

Transit method

This information is based on the yr. 2017 project in the UiO course AST2000, and the NASA site *Ways to find a planet* [6].

When finding exoplanets using the transit method, we look at the light curve generated by the host star, as illustrated in figure 2. The figure shows how the intensity in brightness changes with time when there is an orbiting planet around the star. In other words, when a planet passes between us and the star, the star light is dimmed. If a star has multiple planets in orbit, the light curve will be more complex, and harder to decipher.

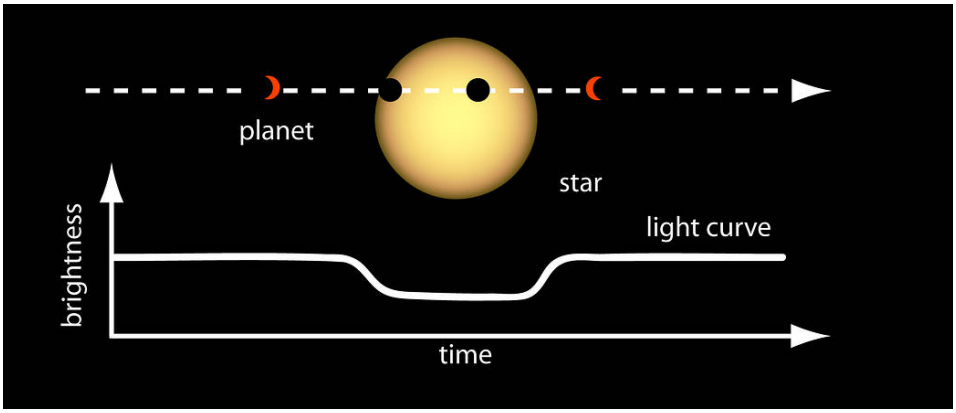

Figure 2: Transit method. The light curve from a star with an orbiting planet. Illustration from NASA Ames; https://www.nasa.gov/mission_pages/kepler/multimedia/images/transit-light-curve.html

We can determine the orbital period of the planet by measuring the passed time between transits. Kepler's third law of Planetary motion can then be used to determine the average distance between the star and the planet. Another way to establish the average distance is by using the transit period time. The further away a planet is from the host star, the longer time it will use passing in front of the star, thus giving a longer orbital time and a longer star-planet distance. We can also calculate the radius of the planet by knowing the size of the star and measuring the dip in brightness. A larger planet will block out more light and cause a deeper light curve than a smaller planet.

Useful information about the temperature and atmospheric composition of a planet can also be provided by the transit method. The light from the star passes through the planets atmosphere, if any, before it reaches our instruments. By using spectroscopy, or looking at the absorption line spectrum, we can determine which molecules exists in the atmosphere of the planet.

2.3 Habitable zone

In the Solar system, we have three planets within the habitable zone, but only Earth has life. Venus, on the inner edge of the zone, suffers from a runaway greenhouse effect, and are therefor unable to support life as we know it. Mars on the opposite edge of the zone, lost its atmosphere due to loosing its magnetic field. A consequence wast that the oceans disappeared, the temperature dropped and Mars transformed into the red dust planet we know it today. Since we only know about life on Earth, the search for extraterrestrial life is narrowed down by criteria based on the fundamental requirements for life found on Earth. We look for exoplanets that are Earth-size, has liquid water on the surface and has an atmosphere filled with oxygen.

The habitable zone, or the Goldilocks zone, is the area around a star where the conditions for liquid water on the surface is just right. The planetary objects, or moons, in this zone is neither too hot nor too cold. The exact temperature range for this zone is hard to determine, but one used range is 260-390 Kelvin [7]. It is important to know that the habitable zone is just an estimate of the possibility of liquid water. Factors like having a magnetic field and an atmosphere is important.

With eq. 1, we can calculate the theoretically inner and outer boundary of the stars habitable zone by inserting a minimum and maximum planet surface temperature, 260 and 390 K.

$$r = \frac{R_* T_*^2}{2T_p^2} \tag{1}$$

where R_* and T_* are the star radius and surface temperature, and T_p is the planet surface temperature. The formula is derived from Stefan-Boltzmann law, where the planet is a black body and the atmosphere is neglected.

The habitable zone of a planet will vary greatly with the size of the star. A giant star, which are burning hot, will have a larger habitable zone further away from the star. A smaller, cooler, star will have a habitable zone relatively close to the star. Red dwarf stars are an example. We also need a stable system for advanced life to evolve. Hot-burning stars burn through their fuel fast and have a relative short lifespan, not ideal for life. Cooler stars live much longer, which gives a stable system and enough time for advanced life to evolve.

3 Machine Learning

Machine learning deals with two main problems: Supervised learning and unsupervised learning. In supervised learning, we have a set of data that contain both the input X and the output/response Y, and we train a model to map from X to Y. In unsupervised learning one does not have a reponse variable. This project deals with supervised learning.

Further, within supervised learning we have two main problems: Regression and classification. In regression, we attempt to make a model that will predict outcome of a continuous function, while in classification we do so for discrete outcomes that are labelled classes. In this project we have a dataset where the binary response variable is labelled 0 or 1, so this is a binary classification problem in supervised learning.

We considered four methods for classification: Logistic Regression, Neural Networks, Random Forests, and eXtreme Gradient Boosting(XGBoosting).

3.1 Logistic regression

The theory for Logistic regression can be found in chapter 3.1, in FYS-STK4155 project 2, found on the linked git page [2]. In the project Python code, we used Scikit-Learn LogisticRegression [16].

3.2 Neural Network Classifier

The theory for Neural Network can be found in chapter 3.2, in FYS-STK4155 project 2, found on the linked git page [2]. In the project Python code, we used Scikit-Learn MLPClassifier [17].

3.3 XGBoost Classifier

XGBoost is an optimized gradient boosting method, used for classification and regression problems. This section is in large part based on *How to explain gradient boosting* by Terence Parr and Jeremy Howard, both of University of San Francisco's *MS in Data Science Program* [23], and also John M. Aiken's slides *xgboost* [1]. In the project Python code, we used XGBClassified provided by XGBoost [24]

Boosting is a method, where a model is derived by making it a sum of simple models. Suppose we have a data set of N observations. Contained in the row vector $\mathbf{y}$ are the outputs. Let X be an $n \times p$ matrix, which contains the input data for p predictors for each of the n observations. Let $\hat{\mathbf{y}}$ be the model made to approximate $\mathbf{y}$. Then a boosting strategy would be to let

$$\hat{\mathbf{y}} = F_M(X) = \sum_{m=1}^{M} f_m(X) \tag{2}$$

where f_m are the weak models, which in our case are going to be decision trees, so the resulting model F_M will be a generated forest. The model is built with adding one weak model as a time, and that can be summed up by the difference equation:

$$F_m = F_{m-1} + \eta f_m$$
$$F_0 = f_0$$

That is, the model at any given stage is the same as the last one with one additional weak model added to it. Given an initial condition $f_0(X)$, which is the first model as well, we evaluate it by computing the residual using the loss function L. Since we are considering classification, our loss function is the cross entropy.

$$L = -\left[\mathbf{y}^T \ln(\hat{\mathbf{y}}) + (1-\mathbf{y})^T \ln(1-\hat{\mathbf{y}})\right] \tag{3}$$

What separates gradient boosting from, say, Random Forests, is that it only generates one tree at a time, evaluates it and then generates a new one. It only stops when a threshold is met in regards to acceptable error or when the maximum number of steps is reached. In that sense it is a gradient descent method, since it takes steps in solution space to approximate the solution.

The parameter η is called the shrinkage, and is analogous to the learning rate in stochastic gradient descent. It gives a weight to each model, which allows us to make them less important, and the model is instead improved by having many small trees approximate the solution rather than a few one that are just right. This prevents overfitting.

3.4 Random Forest Classifier

This theory is inspired by Tony Yiu's explaination of Random Forests, [25].

To better understand the Random Forest Classifier, it's useful to look into the structure of a basic decision tree, since decision trees basically are the building blocks of a Random Forest. To better visualize how a decision tree works, lets use an example. In figure 3, we see that we start out with 7 ones and zeros, which is either red or blue and may be underlined or not. The decision tree now needs to classify the data based on the feature, which brings us to the first node in the tree. A node, which is the point of the decision tree where the path splits into two observations. The tree wants to split the data as reasonably as possible, so at each node the tree will ask:

What feature will allow me to split the observations at hand in a way that the resulting groups are as different from each other as possible (and the members of each resulting subgroup are as similar to each other as possible)? ([25]).

Since all the zeros are blue, except one, this seems to be the feature that will separate our data best. After the first node which splits the data by color, we see that all the blue zeros are not underlined. This means that our data separation is done. However, in the red branch, some numbers are underlined, and some are not, which creates a new node which separates the data again by using this feature. Now we can see that our separation of data based on the two features is complete. In other words, we have finished building our decision tree.

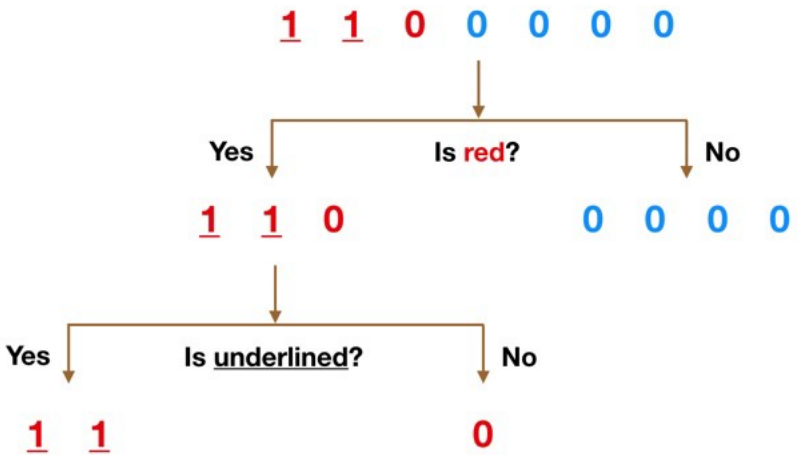

Figure 3: A simple example illustrating decision trees ([25]).

The Random Forest Classifier consists of individual decision tress (described above), that works together as an ensemble. Each individual tree will result in a class prediction (see figure 4), and the class prediction returned the greatest number of times, will end up as our model's final prediction. So even though some of our trees may predict wrong, the majority of the trees will hopefully make the right prediction. In other words, the trees protect each other from their individual errors. This is of course under the precondition that the Random Forest Classifier will work well on our set of data. There are essentially two main requirements that must be fulfilled when considering using a Random Forest:

- The features must be in such a way, that a model based on those features, will do better than a random guess.
- The predictions (thus also the error) must have low correlation with each other.

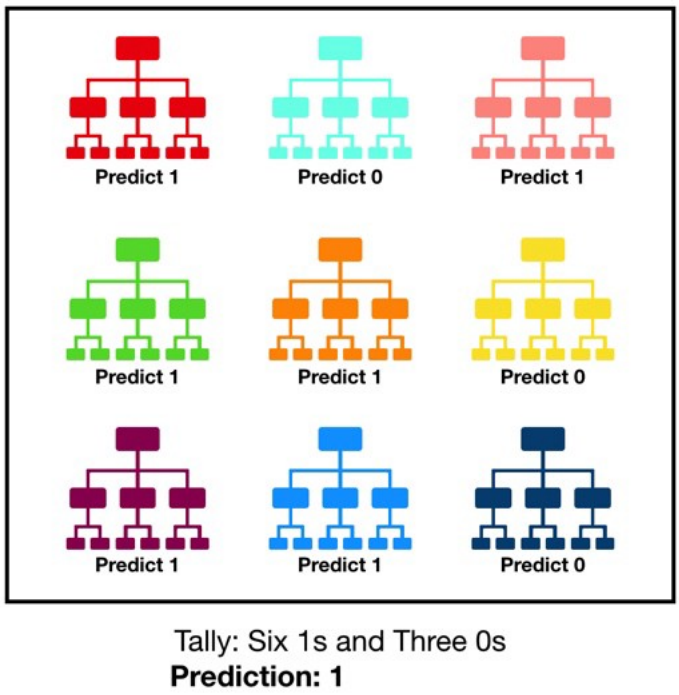

Figure 4: A forest of individual decision trees ([25]).

To ensure that the predictions have low correlation, the Random Forest uses the methods bagging (Bootstrap Aggregation) and feature randomness. Since decision trees are sensitive to the training data, the Random Forest takes advantage of this by allowing each tree to sample from the dataset with replacement. This results in different trees. So, in bagging, we do not split the training data into sub train sets. This means that if we have a sample of size N, we still give each individual tree a training set of size N. However, instead of the original training data, we take a random sample with replacement as the training set.

As described above, we choose the feature that creates the largest separation between observations in the left node versus the right node, when using a single decision tree. In contrast, when using a Random Forest, each tree is only given a random subset of the features to evaluate. This results in even less correlation between trees.

So in our random forest, we end up with trees that are not only trained on different sets of data (thanks to bagging) but also use different features to make decisions ([25]).

In this project we have used Scikit-learns built-in Random Forest Classifier, [18].

3.5 Model evaluation

To assess the performance of our machine learning algorithms, we used the evaluation metrics accuracy, precision, recall, F_1 score and the absolute error. These metrics, except from the absolute error, are thoroughly described in Project 2 [2]. The absolute error is calculated by $\varepsilon = |\hat{\mathbf{y}} - \mathbf{y}|$, where $\hat{\mathbf{y}}$ are the predicted values and $\mathbf{y}$ are the true values.

We also made a confusion matrix for each model, illustrating the true and false predictions.

	Predicted: Not exoplanet	Predicted: Exoplanet
Actual: Not exoplanet	True Negatives (TN)	False Positives (FP)
Actual: Exoplanet	False Negatives (FN)	True Positives (TP)

Figure 5: A Confusion Matrix

The confusion matrix tells us that the classifier predicts the class correctly if we have True, and incorrectly if we have False. Positive and Negative tells us if the classifier predicted the desired class.

When using the KOI data, we define Positive to be the prediction of an Exoplanet, since these are the KOIs we want to predict.

Thus, the confusion matrix can be explained as the following [14]:

- **TN:** The classifier predicts Not exoplanet, and the KOI is not an exoplanet
- **TP:** The classifier predicts Exoplanet, and the KOI actually is an exoplanet
- **FN:** The classifier predicts Not exoplanet, but the KOI is an exoplanet
- **FP:** The classifier predicts Exoplanet, but the KOI is not an exoplanet

4 Data: Kepler Object of Interest (KOI)

The KOI table consists of Kepler's objects of interest, where the goal is to discover exoplanets. The data is from NASA Exoplanet Archive - KOI (Cumulative List) [4]. Data information is described in detail under NASA's *Data Columns in Kepler Objects of Interest Table* page [3].

We will therefor only give a brief explanation about the koi_pdisposition column. This is the column that contains the information about witch class an object is classified as. The classification is divided into four classes; CANDIDATE, CONFIRMED, FALSE POSITIVE and NOT DISPOSITIONED. We only look at the first three in this analysis.

CANDIDATE
The candidates, are as expected, the class consisting of unidentified objects that has to be determined as either an exoplanet (CONFIRMED) or not an exoplanet (FALSE POSITIVE). Testing, data analysis and often gathering additional data, has do be conducted to reclassify a KOI.

CONFIRMED
A CONFIRMED object is an object that is believed to be an exoplanet. There are several tests a KOI has to pass to be classified as an CONFIRMED object. These are described in Batalha et al. (2013) [5], and will not be discussed in this project. Further testing, can still reclassify a CONFIRMED object to a FALSE POSITIVE. It is believed that when NASA's JWST is operational, it will be easier to classify objects of interest more correctly, as well as determining if an exoplanet has an atmosphere and can be classified as an predicted habitable exoplanet.

FALSE POSITIVE
A KOI object classified as an FALSE POSITIVE is an object that failed one or more tests as mentioned above. There are several possibilities for a KOI to be classified as something else than an exoplanet. The KOI could be an eclipsing binary star or the transit lightcurve could be contaminated. Stellar variability and instrumental artifacts could be confused for coherent planetary transits.

Figure 6, shows the distribution of CANDIDATE, CONFIRMED and FALSE POSITIVE after we processed the data as described in section 4.2. We have 1641 CANDIDATES, 2103 CONFIRMED and 2938 FALSE POSITIVE KOIs.

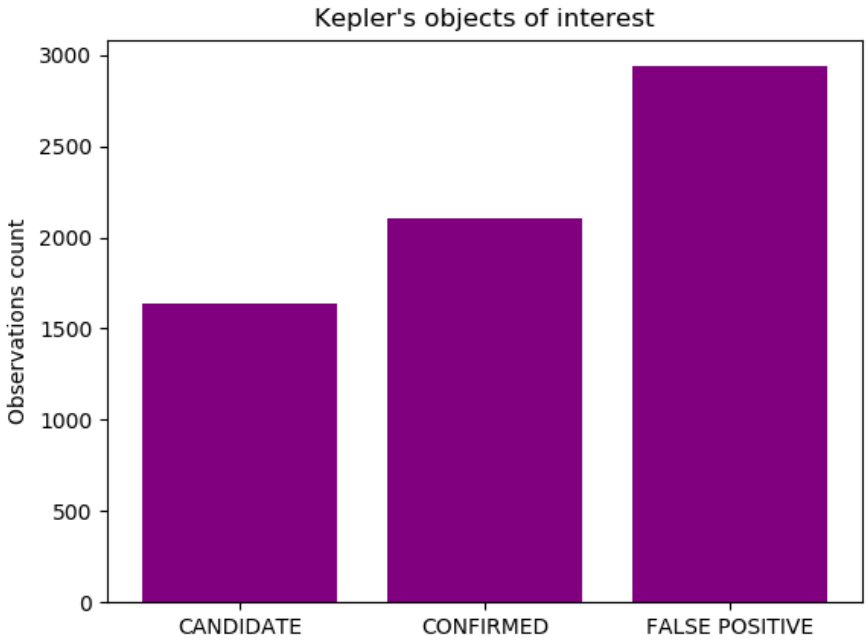

Figure 6: KOI distribution between Candidates, Confirmed and False Positive status

4.1 Feature Elimination

Taking a closer look at the different features, we can observe that some of the features are natural to remove from the dataset. A few of the features consists of only zero values, while other features consists of just raw text comments. In addition, the features koi_time0bk and koi_time0 is basically the same, except from a constant offset. This would result in the two columns to have a 100 % correlation, so we choose to remove one of them. All the features we removed from the dataset, can be viewed in table 1, and a more thoroughly description of the dropping of these features may be found in the scientific article [22]. As mentioned earlier in the section, NASA provides a full overview of all columns/features in the original data [3].

Table 1: Features removed from the dataset

Reason	**Feature**
All zeroes	koi_longp, koi_ingress, koi_model_dof, koi_model_chisq, koi_sage
Free form text	koi_comment, koi_limbdark_mod, koi_parm_prov, koi_trans_mod, koi_datalink_dvr, koi_datalink_dvs, koi_tce_delivname, koi_sparprov
Leakage	kepoi_name, koi_pdisposition, kepler_name, koi_score
Unique Database Identifier	rowid, kepid
Duplicate	koi_time0bk
Zero variance	koi_vet_stat, koi_vet_date, koi_disp_prov, koi_ldm_coeff3, koi_ldm_coeff4

4.2 Further pre-processing of the KOI data

After removing several of the features as described above, we continued to pre-process the remaining data. We noticed that there were quite many NaN values in the dataset, which we chose to eliminate by just simply dropping the corresponding KOI from the data.

We then extracted the candidates from the data, to obtain the case of binary classification. Now we had a dataset just containing the false positives (not exoplanets) and confirmed (exoplanets). The columns koi_disposition was renamed as our target values, while the remaining columns was set as the features. The target names CONFIRMED and FALSE POSITIVE was then renamed to 1 and 0 respectively.
Our target columns were now complete, but the features needed to be scaled. We chose to use the **StandardScaler()** provided by scikit-learn, which standardize the features independently by removing the mean and scaling to unit variance [19].

4.3 Habitable exoplanets

When considering which exoplanets that could be habitable, we filtered out candidates by looking at two parameters. The exoplanet radius and surface temperature. Using the surface temperature directly, means the exoplanet can lie outside the habitable zone of the system, but it can still be inhabitable and liquid water may exist on the surface. We confined the planet radius to be 0.5 to 2.5 Earth radii.

According to the scientists that wrote the article *From climate models to planetary habitability: temperature constraints for complex life*, "The thermal limits for the active metabolism and reproduction of multicellular poikilotherms on earth are approximately bracketed by the temperature interval $T \in [0, 50]^{\circ}$C" [21]. We therefor used this temperature interval, instead of $T \in [260, 390]^{\circ}$K as mentioned earlier, when filtering out candidates. This temperature range corresponds to $T \in [275, 323]^{\circ}$K, and gives the highest probability of complex life evolving on a planet. The candidates were saved in **GoldiLock_PandasDataFrame.xlsx**.

5 Analysis

5.1 Training and testing the models

For the analysis in this project, we split the features and targets into training data and test data. Using scikit-learn's built in function for splitting data, we trained the model on the training data, before we tested the models performance by calculating different metrics based on the test data. To optimize the models performance, we also used GridSearch Cross-Validation on the hyper-parameters, which is further described in the next sub-section. A workflow chart describing the building of the model, is presented by figure 7.

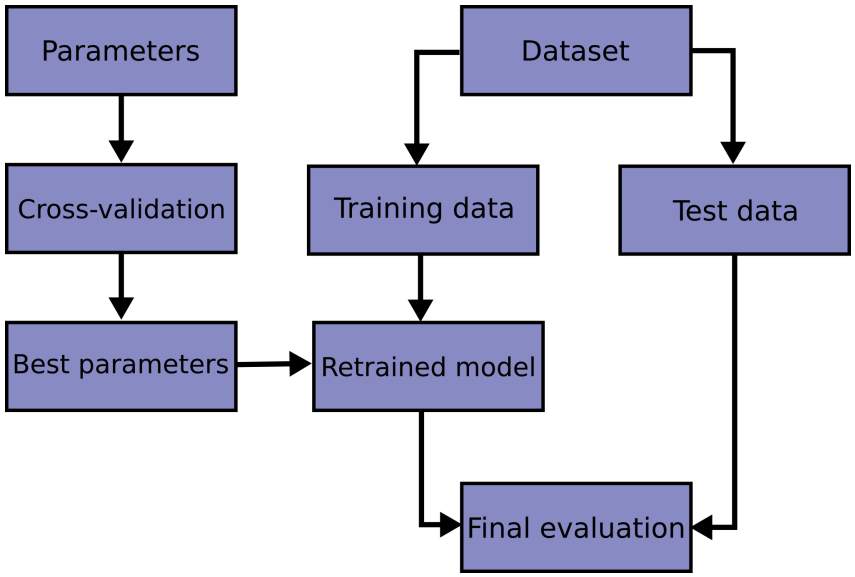

Figure 7: Cross-Validation Workflow in model training [15].

5.2 Tuning hyper-parameters of an estimator: GridSearch Cross-Validation

Hyperparameters of a given model are the parameters that are not determined by the model itself (like the weights and biases of a neural network), they have to be given values beforehand. This was especially important in XGBoost, since its reliability is heavily dependent on tuning the parameters.

To find the optimal hyper-parameters for our machine learning algorithms, we chose to use scikit-learn's built-in method **GridSearch Cross-Validation**. The method search the hyper-parameter space, and finds the best cross validation score that is possible. Any parameter needed when constructing an estimator, can be optimized using this function [20].

A certain set of hyper-parameters may have a large impact on the models computation performance and the predicted outcome of the model. Therefore, it is highly recommended to use methods such as GridSearch on some of the parameters. Others may be less important, and it is sufficient enough to use the default values. The parameters we provided to GridSearchCV(), together with the chosen best fit parameter, is provided below in table 2 and 3 for all four methods.

Parameter tuning proved to be an important step to get reasonable results from XGBoosting, so the process will be described here. Because there are many parameters that goes into XGBClassifier [24], we used a proceeding described by Aarshay Jain ([9]).

Initially, the learning rate was set to 0.1. All other parameters were set to default values, while the parameters `min_child_weight` and `max_depth` (which sets the maximum depth of the given decision tree). Both of these are important to get right to avoid over-fitting. After these were determined, we moved on to gamma, which is a parameter that sets a lower limit on the loss per decision tree. Then lastly, `subsample` and `colsample_bytree` were determined to the first decimal. Only after this point did the model start to yield results that were not obviously false (for example, that all the candidates were exoplanets). After this the learning rate could be lowered and the number of trees increased to set the final parameters, as given in Table 2.

Table 2: GridSearchCV for XGBoost. With $cv = 5$

Parameter	**Tested values**	**Step length**	**Best fit**
min_child_weight	(0, 10)	2	0
max_depth	1,2,3,4,5	-	2
gamma	(0, 1)	0.1	0
subsample	(0.1, 1)	0.01	0.5
colsample_bytree	(0.1, 1)	0.01	0.6

In table 3 we have listed all parameters used in GridSearchCV for Neural Network, Random Forest and Logistic regression. In all methods, including XGBoost, we also used a fixed seed for random_state.

Table 3: GridSearchCV for Neural Network, Random Forest and Logistic regression. With $cv = 5$

Parameter	**Tested values**	**Best fit**
Neural Network		
hidden_layer_sizes	100, 120	100
learning_rate_init	0.001, 0.01	0.001
max_iter	1000, 3000	3000
alpha, α	0.0001, 0.001	0.001
Random Forest		
n_estimators	200, 300, 400, 500	300
max_features	'auto', 'log2'	'auto'
max_depth	7, 8, 9, None	8
min_samples_leaf	1, 2, 3, 4	1
Logistic Regression		
solver	'lbfgs', 'liblinear', 'saga'	'liblinear'
max_iter	50, 100, 200, 500	50

5.3 Threshold

The classification models needs an threshold parameter for predicting the number of confirmed exoplanets and false positives. The default value is 0.5, which means an KOI need more than 50% probability to be an exoplanet to be classified as an exoplanet. In this project we have used threshold 0.5 and 0.9. We use 0.9 when predicting habitable exoplanets, that have a high probability of actually being an habitable exoplanet. This is important for further research, when NASA scientist are labeling exoplanet candidates, and for the day we want to further explore new worlds.

5.4 Correlation matrix

After pre-processing the data we made an correlation matrix (figure 8) that shows areas of correlation between features. We can see from the correlation matrix that koi_dicco_msky and koi_dikco_msky, as well as koi_prad has some correlation with other parameters. But we do see that most correlation is between closely linked parameters, like "mag" features that gives the white square. So in general we can say that this correlation matrix gives little useful information, at least without knowing the data set better, and how the features impact the probability of a KOI being an exoplanet.

To make the correlation matrix, we used Panda's **DataFrame.corr()**, with default values [13].

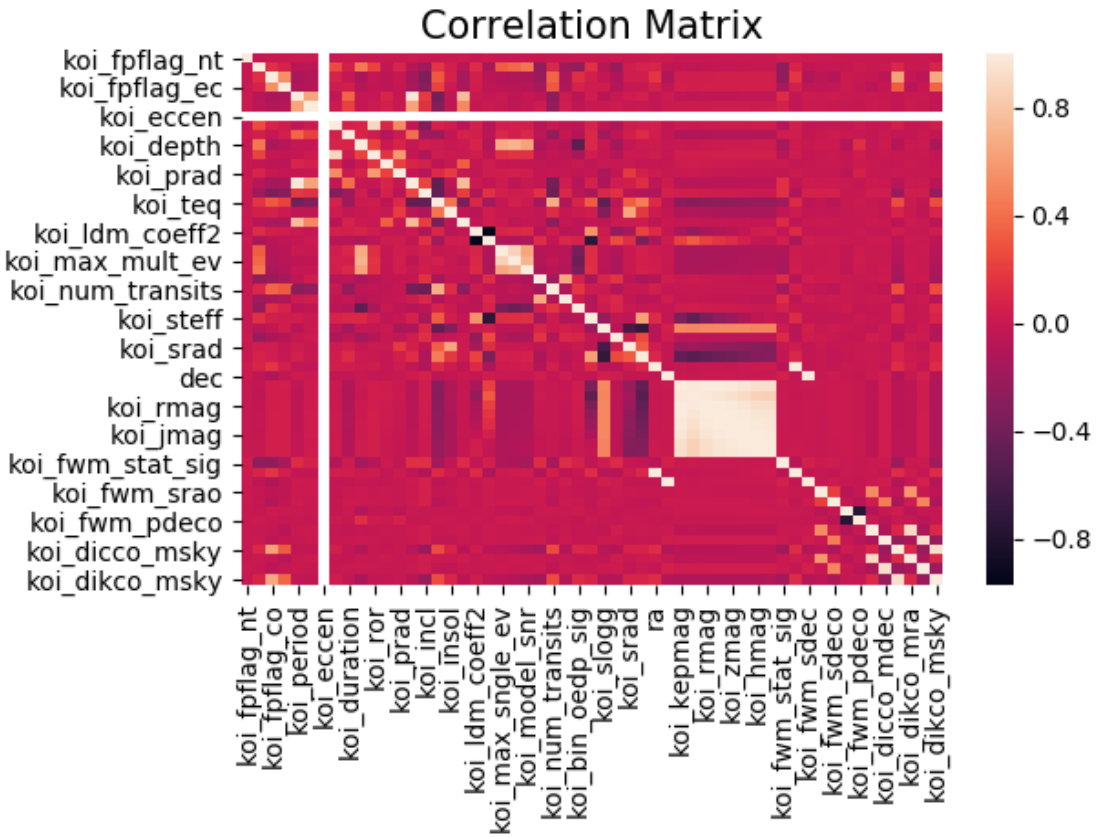

Figure 8: Correlation matrix for the raw data, after unimportant features were removed

5.5 Feature importance

Table 4 gives a short description ([3]) of the most important features illustrated by figure 9. It seems reasonable that the size of the planet, angular offset, flux measurements and presence in a solar system would be important features to determine if an object is indeed an exoplanet.

We can see that individually these features do not have very high importance. So, since some of the features provides information similar to each other, it could be a good idea to merge some of the features into just one feature with a higher importance.

Since the universe is quite a big place to search through, it is normal to direct satellites and other instruments to fields where interesting objects have been found earlier. This is also the case for Kepler, which can cause bias in the data. More information about this bias, can be read in the article *Machine Learning Pipelinefor Exoplanet Classification* ([22]). In this article, the feature koi_count turned out as the most important feature,

Therefore, an object of interest with similar features as a confirmed planet could be classified as a "false positive" simply due to there not being a candidate planet being detected in the system.

Thus, the authors behind the article chose to remove this feature. Since it did not turn out as important for us, we chose to keep it, but the impact of the feature should be kept in mind.

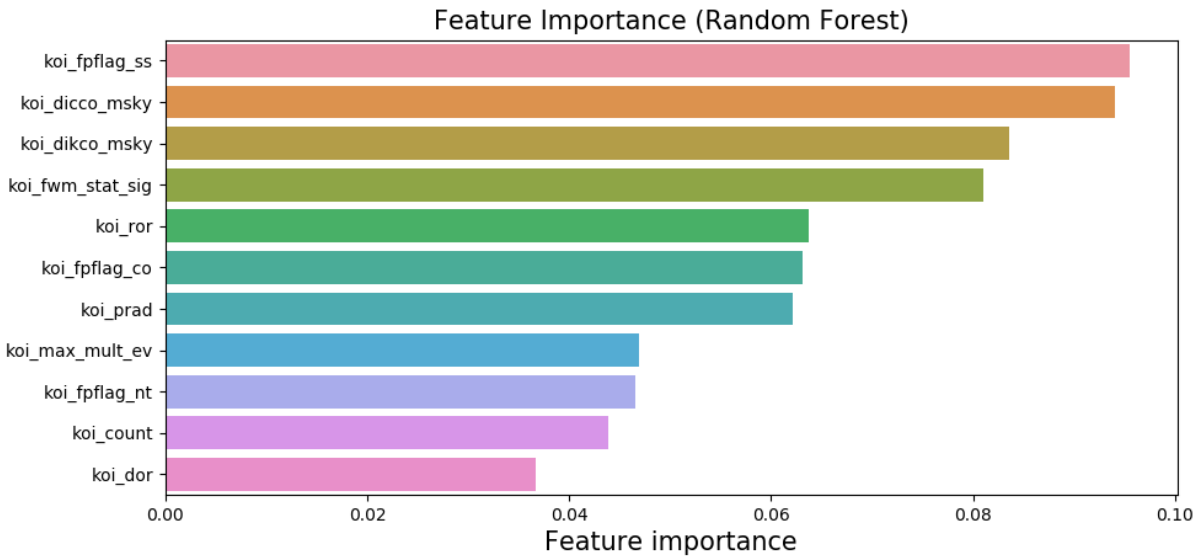

Figure 9: Feature importance for Random Forest

There is a similar feature importance figure for XGboost on the linked GitHub page.

Table 4: Description of the most important features

Feature	Description
koi_prad	planet radius
koi_fpflag_ss	Transit
koi_fpflag_nt	Transit
koi_dicco_msky	Angular Offset
koi_dikco_msky	Angular Offset
koi_fpflag_co	Transit and presence in a solar system
koi_fwm_stat_sig	Flux-Weighted Offset Significance
koi_ror	Planet-Star Radius Ratio
koi_dor	Planet-Star Distance over Star Radius
koi_max_mult_ev	Maximum Multiple Event Statistic
koi_count	Number of Planets (identified in a system)

5.6 ML Classification method selection

The calculated values on the test data for accuracy, precision, recall, F_1 score and the absolute error for the four methods can be viewed in table 5.

We can see that Logistic Regression did the worst predictions, while the three other methods did pretty well. XGBoost seems to be the best model based on these results, however, observations on the candidate predictions in the next section, may indicate that Random Forest would be our best choice.

Table 5: Evaluation metrics for the different methods

Classification	XGBoost	Random Forest	Neural Network	Logistic regression
Accuracy	0.999	0.989	0.930	0.799
Precision	0.999	0.995	0.924	0.791
Recall	0.998	0.977	0.934	0.800
F_1	0.999	0.986	0.928	0.794
Absolute error	0.001	0.011	0.070	0.201

6 Results and discussion

Instead of dropping the NaN values, we could have categorized them as one category and used as a part of the training data. In the article *Machine Learning Pipelinefor Exoplanet Classification* [22], they investigated both zero filling (which should be used with caution) and KNN imputation. Using KNN imputation, they were able to obtain 930 observations.

We chose to drop the NaN values both from the training/test data and from the CANDIDATE data. However, using a method to handle the NaN values, could have resulted in even better model predictions.

6.1 Predictions of the Candidates

In table 7 the number of predicted exoplanets and false positives are listed for all four methods. The Neural Network method predicts the same distribution for $TH = 0.5$ and $TH = 0.9$. Printing the prediction probabilities for for instance Neural Network, we observed that the probability equals to exactly 1, which seems a bit odd. Since logistic regression performs poorer on the test data, and also do not predict any exoplanets with $TH = 0.9$, and XGBoost (which we expected to produce the best results) was difficult to handle in regards to hyperparameters, we concluded that Random Forest seems to be the best method to go on with. We will therefore mainly discuss the further results of Random Forest, while more results from the other methods may be viewed in the GitHub repository found on the front page.

Table 6: Number of predicted exoplanets, out of 1641 possibilities. Exoplanet threshold = TH

Exoplanet predictions	TH = 0.5		TH = 0.9	
	Exoplanets	**False Positives**	**Exoplanets**	**False Positives**
Logistic regression	1166	457	0	1641
Neural Network	1343	298	1343	298
XGBoost	1590	51	1448	193
Random Forest	1213	428	447	1194

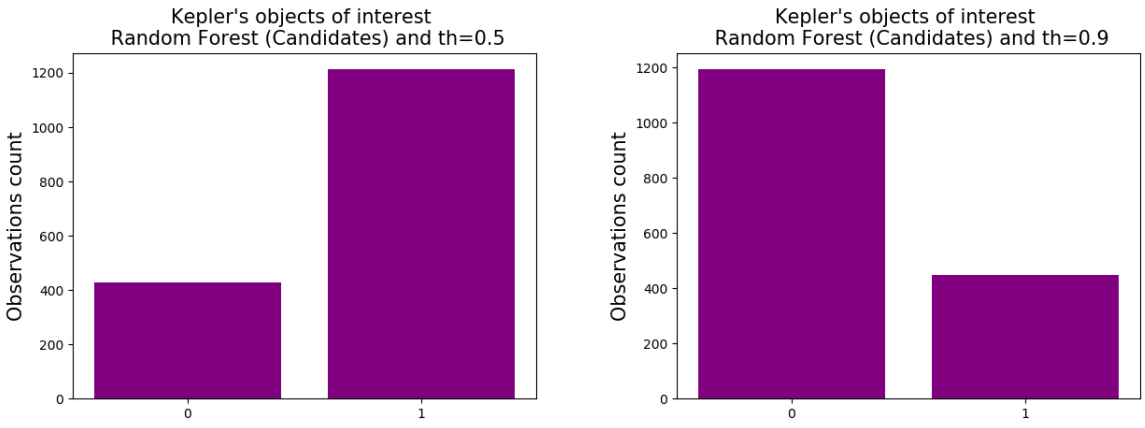

Figure 10: The distribution of exoplanets for RandomForest classifier. 1=Exoplanet and 0=False Positive

The confusion matrix created by Random Forest on the test data can be viewed in figure 11. The model performed quite well, with only 13 wrong predictions of FALSE POSITIVES (not exoplanet), and 3 wrong predictions of CONFIRMED (exoplanet). Similar confusion matrices for all methods can be found on the GitHub page.

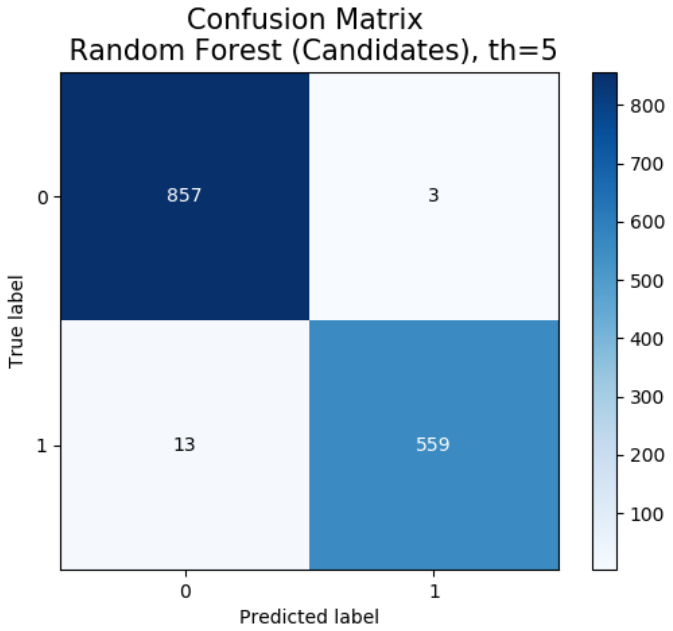

Figure 11: Confusion matrix for the RandomForest classifier. 1=Exoplanet and 0=False Positive

6.2 Predicted Habitable exoplanets

In table 7 we can see that Neural Network predicts the same number of exoplanets regardless of the threshold. We have earlier concluded that XGboost was difficult to tune, and we see that Logistic Regression do not predict any exoplanets when increasing the threshold to 0.9. So, Random Forest is the method that provided the best model for discovering habitable exoplanets, most likely because it avoided overfitting.

Table 7: Number of predicted habitable exoplanets, out of 24 possibilities. Exoplanet threshold = TH

Habitable exoplanet predictions	TH = 0.5		TH = 0.9	
	Exoplanets	**False Positives**	**Exoplanets**	**False Positives**
Logistic regression	13	11	0	24
Neural Network	16	8	16	8
XGBoost	21	3	17	7
Random Forest	16	8	3	21

By using RandomForest classification on the KOI data set, filter out candidates for the habitable zone and increasing the threshold to 0.9, we found 3 possible habitable exoplanets. These exoplanets are listed in table 8, with corresponding important parameters. Figure 12 and 13 shows the temperature-radius placement of the predicted habitable exoplanets. Similar plots as figure 12 and 13 for all classification methods can be found on the linked GitHub page.

By calculating the theoretically habitable zone (eq 1), and comparing the planets orbit semi-major axis, we get a verification that the planet could support liquid water and life.
These 3 exoplanets were also classified as CONFIRMED by the Neural Network- and XGBoost classifier.

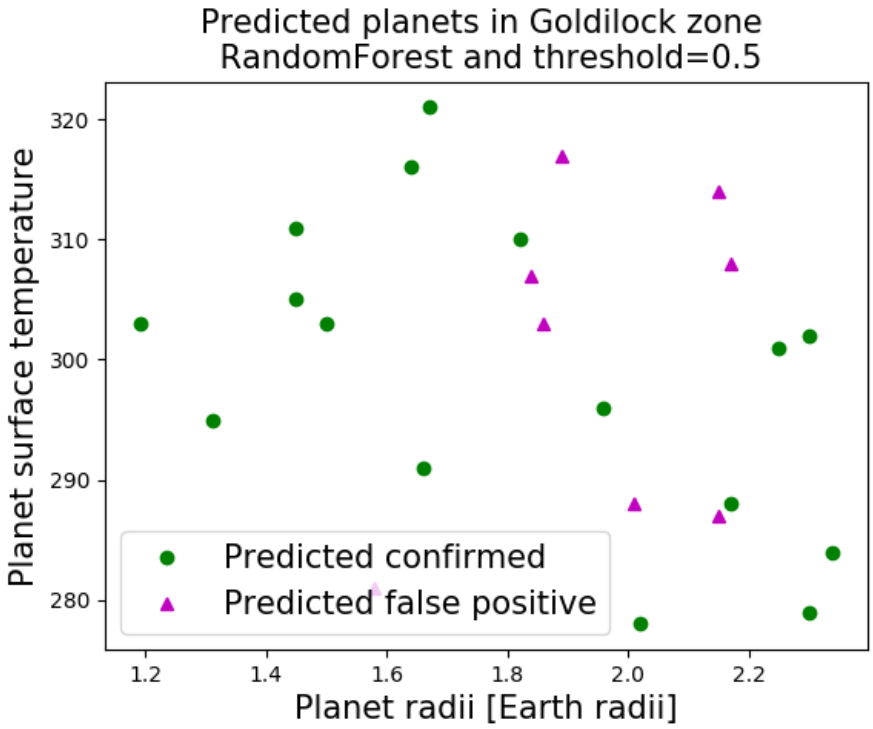

Figure 12: Predicted habitable exoplanets by RandomForest classifier. Where exoplanets with at least 50% chance of being an exoplanet is classified as CONFIRMED.

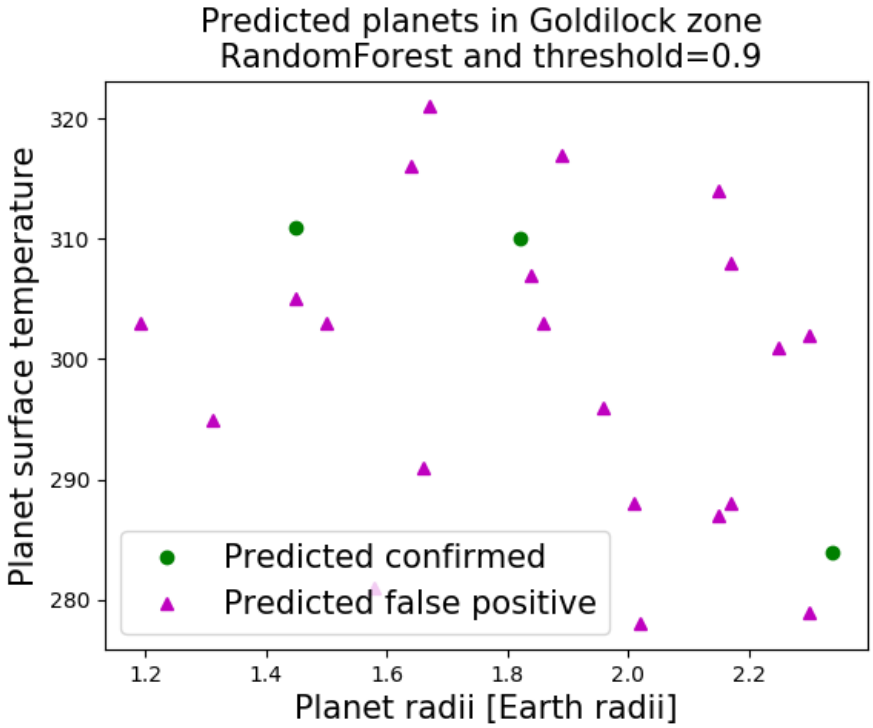

Figure 13: Predicted habitable exoplanets by RandomForest classifier. Where exoplanets with at least 90% chance of being an exoplanet is classified as CONFIRMED.

We can see from table 8 that all three exoplanets orbits a star within the calculated habitable zone. In addition, we can see that all three host stars are smaller than the Sun, which likely gives stable solar systems, with enough time for advanced life to evolve. All exoplanets are however larger than the Earth, but Earth-sized planets in general have been found a bit larger than the Earth. This could be caused by the transit method, where larger objects, compared to the host star, are easier to discover.

Table 8: Parameters for KOI object K02124.01, K02290.01 and K02469.01

Parameter		**K02124.01**	**K02290.01**	**K02469.01**
kepid		11462341	12121570	6149910
Nr. of planets in system		1	1	1
Nr. of transits		33	14	10
Habitable zone, eq 1	[AU]	0.151 to 0.341	0.275 to 0.619	0.294 to 0.662
Star-planet dist., eq 1	[AU]	0.238	0.435	0.555
Exoplanet parameter				
Orbit Semi-major axis	[AU]	0.1997	0.3648	0.4644
Orbital period	[days]	42.33	91.5	131.18
Transit duration	[hrs]	4.924	4.842	6.9
Eccentricity		0	0	0
Inclination	[deg]	87.86	89.99	89.97
Planetary radius	[Earth radii]	1.45	1.82	2.34
Equilibrium temperature, T_p	[K]	311	310	284
	[C]	37.85	36.85	10.85
Stellar parameter				
Effective temperature, T_*	[K]	4132	5051	4898
	[C]	3859	4777	4625
Mass, M_*	[Solar mass]	0.593	0.786	0.774
Radius, R_*	[Solar radii]	0.581	0.706	0.803

Parameters from NASA KOI table, without errors.
Note that "Nr. of planets in system" is discovered nr. of planets in the system. We know that multiplanetary systems are common, as well as binary-star systems.
Read about the parameters at the NASA page; Data Columns in Kepler Objects of Interest Table [3]

7 Conclusion

The hunt for habitable exoplanets is well underway, and the motivations behind the driving force are many. There are everyone from scientists to private companies that invest a huge amount of time and resources to search for other Earth-like planets.

During this project, we have tried to employ our knowledge about Machine Learning, to contribute to the discovery of exoplanets that may be suited for life. The datasets associated with exoplanets are quite fit for supervised learning, since there are a large number of predictors for labelled outcomes. As the size of the data grows, these trained models will only be better and able to handle more and more general cases. To use machine learning to discover new exoplanets it is crucial to learn about the features in the data set, and have enough knowledge about astronomical aspects around exoplanet hunting.

Due to its popularity in modern supervised learning, we expected to get the best results with XGBoosting. However, it turned out to be difficult to correctly tune the hyperparameteres, and it was prone to overfitting.

As the technology to investigate these planets is constantly developing, there are few things to consider. Maybe it will bring us to these exotic planets outside our solar system in the future. If so, this gives rise to several important questions. For instance, there are ethical aspects to consider when claiming new territories, and exoplanets are no exception. If the planets contain any form of life, the visit from other species may bring deceases and can cause destruction of the entire ecosystem of the planet. Also, if it turns out that it is intelligent life out there, the hunt for habitable planets may cause other beings to be aware of us. This can be a potential threat to our planet.

Despite the ethical aspects, it is no doubt that the search for other habitable planets is a very interesting research area, that we hope will bring many new and exciting discoveries in the near future.

The future of exoplanet exploration

ExEP has a detailed plan* until late 2020s and a pending plan for 2030 and beyond, which includes the use of 7 spacecraft. These spacecrafts, mainly telescopes, will have different main areas of interest, as seen in table 9.

TESS	2018	A telescope that finds nearest transit planets.
JWST	2021	One task of the telescope will be to investigate atmospheric chemistry on exoplanets.
WFIRST	2025	Main tasks includes direct imaging, exozodiacal dust and exoplanet diversity.
Starshade Rendezvous:	Pending	Main focus on habitable exo-Earths discoveries.
Origins	Pending	Will focus on M-dwarf rocky planets bio-signatures and cool gass giants.
HabEx	Pending	Exp-Earth bio-signature and habitable exo-Earth abundance.
Exo-Earth Interferometer	Pending	Life verification!

Table 9: NASA's future plans for exoplanet exploration [12]

Acknowledgement

This research has made use of the NASA Exoplanet Archive, which is operated by the California Institute of Technology, under contract with the National Aeronautics and Space Administration under the Exoplanet Exploration Program. We have used the KOI (Cumulative List), as well as NASA's own column descriptions.

References

[1] John M. Aiken. *xgboost*. URL: https://docs.google.com/presentation/d/1A5tMrZSa7XBwSZMEFDkkfnH_DEhu5eWJDU/edit#slide=id.p.

[2] Aron J. Nordberg Anna Eliassen and Kristina Othelia L. Olsen. *FYS-STK4155 - Project 2: Classification and Regression, from linear and logistic regression to neural networks*. URL: https://github.com/kristinaothelia/FYS-STK4155/tree/master/Project2. Published: 13-11-2019.

[3] NASA Exoplanet Archive. *Data Columns in Kepler Objects of Interest Table*. URL: https://exoplanetarchive.ipac.caltech.edu/docs/API_kepcandidate_columns.html?fbclid=IwAR0IaSuH8qqfclj_KJ2KE84YPhC7sjT5Ew-gLRCylTdD8Cvzh4wzhaXJsjY. Last updated: 17 September 2019.

[4] NASA Exoplanet Archive. *KOI (Cumultive List)*. URL: https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=cumulative. Data downloaded: Saturday Nov. 16, 2019.

[5] Natalie M. Batalha et al. "Planetary Candidates Observed by Kepler. III. Analysis of the First 16 Months of Data". In: 204.2, 24 (Feb. 2013), p. 24. DOI: 10.1088/0067-0049/204/2/24. arXiv: 1202.5852 [astro-ph.EP].

[6] NASA Exoplanet exploration. *A ways to find a planet*. https://exoplanets.nasa.gov/alien-worlds/ways-to-find-a-planet/#/2.

[7] UiO Frode Hansen. *AST2000 project: Part 3 (Calculate Habitable Zone)*. https://www.uio.no/studier/emner/matnat/astro/AST2000/h19/undervisningsmateriell_h2019/prosjektlop/prosjektdeler/part3.pdf. 2019 AST2000 project run.

[8] Elizabeth Howell. *Kepler Space Telescope: The Original Exoplanet Hunter*. https://www.space.com/24903-kepler-space-telescope.html. Last updated: 7.12.2018.

[9] Aarshai Jain. *Complete Guide to Parameter Tuning in XGBoost with codes in Python*. URL: https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/.

[10] NASA. *NASA's Exoplanet Exploration Program (ExEP): Overview*. https://exoplanets.nasa.gov/exep/about/overview/. Retrieved: 11.11.2019.

[11] NASA. *NASA's First Planet Hunter, the Kepler Space Telescope: 2009-2018*. https://www.nasa.gov/kepler/missiontimeline. Last updated: 30.10.2018.

[12] Pet Brennan NASA Exoplanet program. *Technology Overview*. https://exoplanets.nasa.gov/exep/technology/technology-overview/.

[13] Pandas. *pandas.DataFrame.corr*. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html. Retrieved: 11.11.2019.

[14] Dr. Saed Sayad. *Model Evaluation - Classification: Confusion Matrix*. https://www.saedsayad.com/model_evaluation_c.htm. Retrieved: 11.11.2019.

[15] Scikit-Learn. *Cross-Validation workflow in model training*. URL: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation.

[16] Scikit-Learn. *LogisticRegression*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[17] Scikit-Learn. *MLPClassifier*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

[18] Scikit-Learn. *Random Forest Classifier*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
[19] Scikit-Learn. *Standard Scaler*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.
[20] Scikit-Learn. *Tuning hyper-parameters of an estimator*. URL: https://scikit-learn.org/stable/modules/grid_search.html#grid-search.
[21] Laura Silva et al. *From climate models to planetary habitability: temperature constraints for complex life*. URL: https://arxiv.org/ftp/arxiv/papers/1604/1604.08864.pdf.
[22] Brychan; Sturrock George Clayton; Manry and Sohail Rafiqi. "Machine Learning Pipeline for Exoplanet Classification". In: *SMU Data Science Review: Vol. 2 : No. 1 , Article 9* (2019).
[23] Jeremy Howard Terence Parr. *How to Explain Gradient Boosting*. URL: https://explained.ai/gradient-boosting/index.html.
[24] XGBoost. *XGBClassifier*. URL: https://xgboost.readthedocs.io/en/latest/python/python_api.html.
[25] Tony Yiu. *Understanding Random Forest*. URL: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

Appendix A - Kepler information

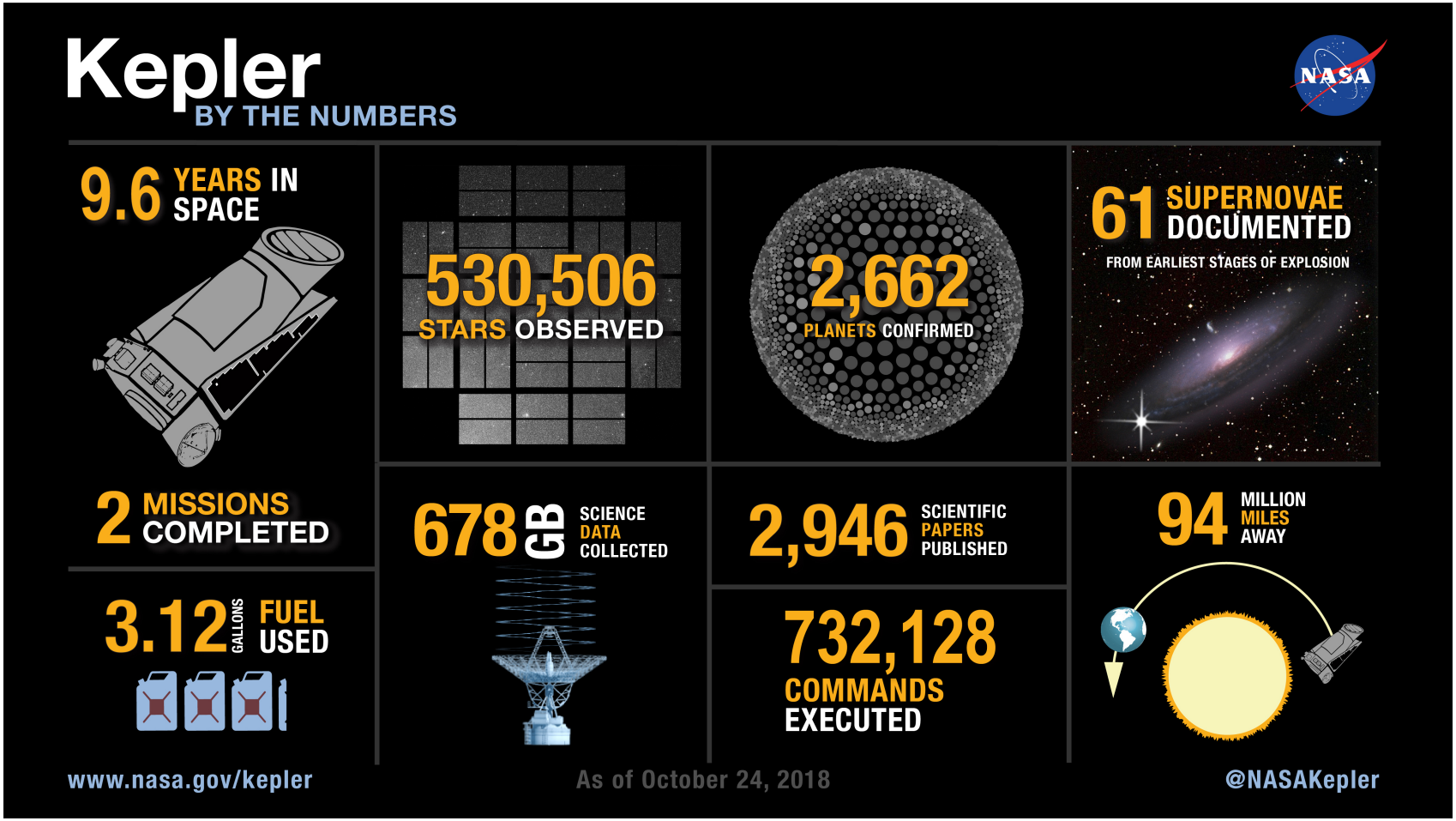

Figure 14: The Kepler mission represented by numbers. Source: NASA https://exoplanets.nasa.gov/resources/2192/nasas-kepler-mission-by-the-numbers/