Deliverable 1: Final Year Dissertation

Using Machine Learning to classify exoplanets

BSc Computer Systems

Menezes, Ashwin Daniel
SUPERVISOR: DOCTOR HANI RAGAB HASSEN

Declaration.

I, *Ashwin Daniel Menezes* confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of references employed is included.

Signed: **Ashwin Daniel Menezes**

Date: **24/11/2017**

Abstract

With telescopes reaping and sending terabytes of data to Earth, we cannot hope to sift through the data by hand. To make use of our information inflation we turn to artificial intelligence, a way to automate our search for answers. In this final year dissertation, we use machine learning, a form of artificial intelligence, to classify Kepler objects as ‘exoplanet candidate’ and ‘non-exoplanet’.

Table of Contents

Chapter 1: About the Project.

Introduction.

In this day and age, we've reached so far out into the Universe in an attempt to understand it, that we've ended up with more than we can chew. With telescopes beaming down terabytes of data every year, manually gleaning useful information from the data, wasting as little as possible, is nearly impossible. So, we turn to machines and automation to give better, faster, and more accurate results. As a starting point of this epic journey that combines AI and Astronomy, we use Machine Learning to detect Exoplanets. So far, the most popular ways are through Light Curves and Gravitational Lensing. Artificial Intelligence has already worked its magic on self-driving cars and image recognition. It has the potential to be our “looking glass” into the Universe, observing things that we couldn't hope to see with just our senses. [1]

Document Overview

In Chapter 1, we introduce to the topic, and show our aims and objectives. We then look at some background research and related work in Chapter 2. In Chapter 3, we show our analysis performed for our requirements, and the evaluation of them. Finally, Chapter 4 presents our planned system architecture, our methodology of choice, and our risk management.

Research Question.

Use Machine Learning to classify the Kepler Mission data into ‘exoplanet candidate’ and ‘non-exoplanet’.

Aims

The aim of this project is to use Machine Learning to classify Kepler Objects of Interest (KOIs) as exoplanet candidates and non-exoplanets.

Objectives.

- Identify the most relevant features in the dataset.
- Perform normalization on the data as required.
- Classify dataset as exoplanet candidates and non-exoplanets.
- Apply the most appropriate machine learning algorithm to the training dataset.
- Evaluate the training done using the most appropriate evaluation methods and the validating dataset.

Chapter 2: Background Reading.

In this Chapter, we will look at the research done for this project, and the decisions reached based on it. We first briefly explore different topics in *Machine Learning* and *Astronomy* to familiarize ourselves with terms that will be used in the rest of the paper.

Machine Learning.

In this topic, we will look at the different terms that are used in *Machine Learning*. To begin, we start with *Neural Networks.*

Neural Networks [2] are essentially a directed graph that uses a learning algorithm that gives a desired output based on the input. *Figure* 1 shows an example of a neural network.

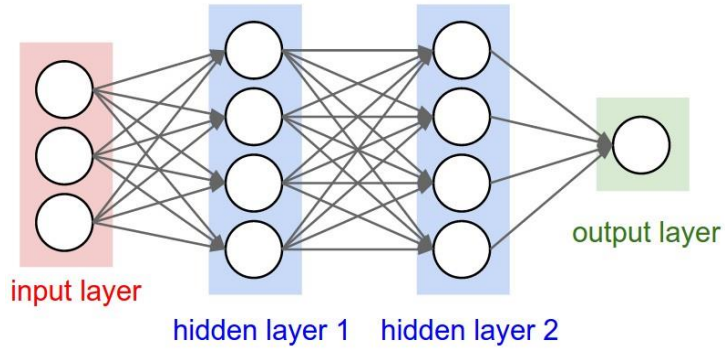

Figure 1: A diagram showing a basic representation of a neural network model. Note that there can be more than one hidden layer. Source: [3]

Neural *Networks* can have many *start* and *end* nodes. Between the input and output layers lies a *hidden layer*. The *Hidden Layer* are a collection of layers that represent all the nodes between the *start* and *end* nodes. The *Hidden Layer* is responsible for all the "learning" in the model. Each node consists of a number of basic elements: an input, a weight, a function to combine the input with the weight, and an *activation function* to produce an output. An *Activation Function* is the function used by the *hidden layer* to calculate the output after the summing function has combined the input and the weight. *RELU* is a kind of activation function that helps speed up training. The weights in the model are then adapted based on the calculations performed. [2]

Before training, [4] the dataset being used is split into 3 subsets, 70% for *training* and 30% for *evaluation*. Before tests are done on how accurate the training is, *validation* needs to be performed. *Validation* is the process of 'fine-tuning' the parameters of the model. If the *validation* set was used in the *training* set, it would cause *overfitting*. *Overfitting* is a what happens when the model has trained so well with the *training* subset, that its accuracy drops with the *evaluating* subset. This can be an issue when you are attempting to use the model to make predictions, as the model will be restricted to what it has learned from the *training set.* Therefore, using data from the *training* set for *validation* should be avoided. Finally, *Evaluation* or *Testing* is the process where the model is tested with data from the *evaluating* subset. The accuracy is then calculated depending on the evaluation method used.

Sometimes, the different features in the dataset have different scales. This can stunt the accuracy of the training. This is where *Normalization* or *Feature Scaling* or *Standardization* comes in. *Normalization* is the process of scaling the features such that all features have a standard deviation of 1 and a mean of 0. *Figure 2* helps to illustrate the effects of *Normalization*. [5]

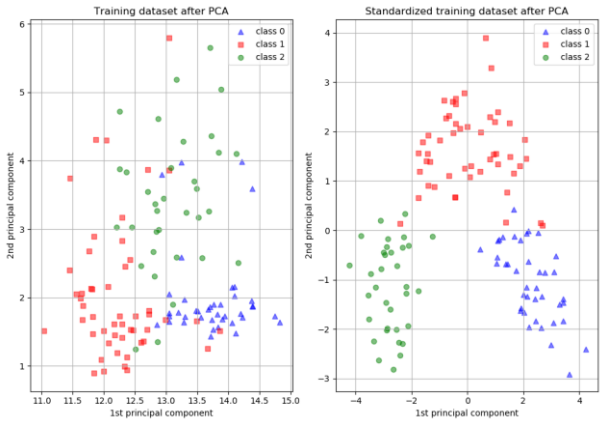

Figure 2: (Left) Before *Normalization,* (Right) After *Normalization*. Notice how (Right) shows recognizable clusters forming. Source: [5]

To conclude, these are some of the popular terms used in Machine Learning. In the next section, we will be discussing some of the popular terms used in Astronomy.

Astronomy.

To begin, let's start with what the project is about, *exoplanets*. *Exoplanets* are planets that are found outside our solar system, and orbit stars other than our sun [6]. When a planet comes in the path between the observer and its orbiting star, the brightness of the star appears to slightly drop for the observer. This phenomenon is known as a *Transit*. If the brightness of a star was plotted over time, you would see a drop, or *curve*, in the graph. This is known as a *light curve*. [7] *Figure 3* shows an illustration with a transiting planet and a graph.

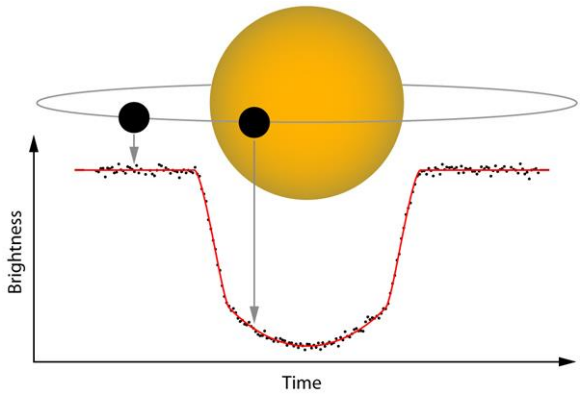

Figure 3: A representation of a transiting planet, and a light curve graph. Notice how the brightness drops as the planet transits. Source: [6]

Another type of event known as *microlensing* is used for exoplanet detection. *Gravitational Microlensing* [7] is an event that occurs when an object's gravity warps the light from a star behind it, causing the brightness to magnify slightly.

Figure 4 shows a visual representation of a microlensing event.

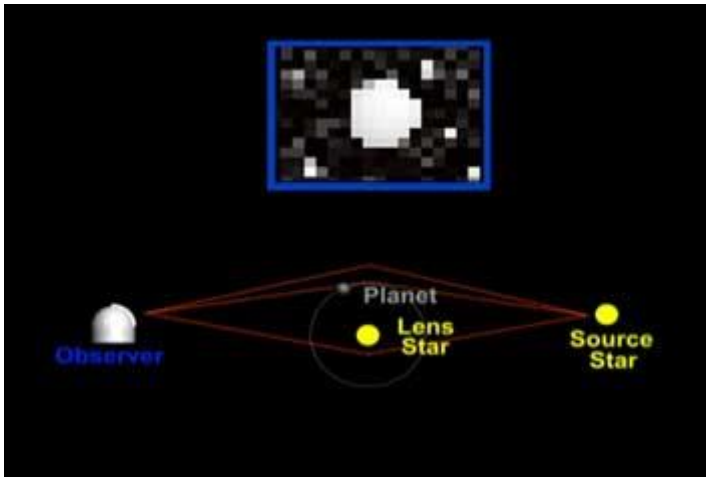

Figure 4: A visual representation of a gravitational lensing event. The gravity from the Lens Star causes the light from the source star to bend around the Lens Star. Source: [7]

These terms will be part of the main focus of our project. In the next section, we look at the different research papers found and our given criticisms for those papers.

Related Work.

Machine Learning with Light Curves.

In this subsection, we look at all the papers that used different Machine Learning techniques, and light curves as input, to detect exoplanets.

Pearson, **Palafox**, and **Griffith** tried to prove the usefulness of Neural Networks over other methods for detecting exoplanets. They did this by training a neural network over artificially generated light curve data, and validating their deep net on light curves from the Kepler mission. They also tested the sensitivity of the algorithms, the effects of using a low-resolution signal onto a high-resolution grid, and the effects of feature loss on the output. The authors started by creating 311,040 light curve entries for training and validating their neural network. They chose to use reLU as the activation function in their hidden layer. The hidden layer size they decided to use was 64, 32, 8, 1 (each is the number of neurons in that layer of the neural network). The program ran through the data 30 times, randomizing the samples in the batches at each epoch. The optimal parameters the authors found was a regularization weight of 0, learning rate of 0.1, momentum of 0.25, and a decay of 0.0001. They tried different regularization terms between $10 – 10^{-6}$ and found that it only decreases the network performance. They tested different algorithms such as 1D Convolutional Neural Networks (CNN) and Wavelets, and found that deep learning algorithms had an accuracy of 99% compared to 73% found from using Least-Squares. To test the sensitivity of the algorithms, they generated another completely new dataset such that the transit depth is less than the noise. The results showed that the size of the transit depth varies little with the accuracy and that the ratio of transit depth to noise dictates the accuracy of the detection algorithm. When they added the low cadence signal onto a higher cadence grid, they found that the CNN had the smallest performance drop because it pools local information together. The accuracy of detection remained the same when evaluating data from a higher resolution grid. They setup the test for Feature Loss by randomly removing chunks from each light curve, and randomized the positions of the chunks to a certain

extent. The CNN had the best performance because it pools local information together. Overall, they found that CNNs had the best performance since they consider local properties in the data before adding it into a fully connected network. CNNs also have properties and technique that help reduce some of the scatter in the data. The authors believed that this algorithm would be able to handle shallower transits at longer periods for fainter stars. They also felt that while the CNN algorithm was robust, additional post processing is needed to help constrain the period, and that detection accuracies could be improved by pre-processing the Kepler data to remove any systematic trends or stellar variability. [8]

Criticism

Even though their data was validated using Kepler data, the use of artificially generated light curve data could have reduced the accuracy of their results, and the fact that they reached an accuracy of 99% could mean that their implementation has "overfit" the data. They performed *Cross-Validation* to evaluate their results, when they could have used *10-fold Cross Validation* repeated 10 times to improve their results even further.

Buisson, **Sivanandam**, **Bassett**, and **Smith** tried to find an optimal solution for transient (A **transient** is a small fluctuation in frequency) classification by testing different machine learning algorithms. They took their data from the 2nd and 3rd years of the SDSS-II SN (Sloan Digital Sky Survey – II Super Nova) survey, with a mean cadence of 4 nights. The transient detection algorithm subtracts the search image from the historical image of the same area in the sky using Sloan color bands *g*, *r*, and *i*. The resulting difference image is then classified into one of 3 classes: real objects, artefacts and saturated. 25% of the data was kept for testing, and 75% for training. 30% of the training data was kept as the 'validation set' for optimization via cross-validation. To get the most useful features from the images, they used PCA (a method of dimensionality reduction) on all the datasets. Linear Discriminant Analysis (LDA) was implemented on the training set, and all the data was normalized to

give all features a standard deviation of 0.5 and mean of 0. The algorithms tested were Naïve Bayes, k-Nearest Neighbors, Support Vector Machines, Artificial Neural Networks, and Random Forest. The winning algorithm was determined using accuracy and recall. They believed that accuracy was a good measure of performance, and recall shows the rate of false negatives (False positives can be weeded out by humans). They found that RF had the best performance overall except for recall. KNN performed the second best except recall. MEC, the simplest algorithm, did the best in recall but severely lagged in other aspects. They realized that the differences between RF, KNN, ANN, and SVM were small and that better optimization could potentially change the final ordering obtained. [9]

Criticism

They used *Cross-Validation* for evaluation, and could instead have used *10-fold Cross Validation* repeated 10 times to further improve their results. They used PCA for Feature Extraction, when **Abdelmajed et al.** proved that LPP is a better Feature Extraction method than PCA, therefore LPP should have been used [10].

Thompson et al. describes a new metric that uses machine learning to determine if a periodic signal found in a photometric timeseries appears to be shaped like the transit of a transiting exoplanet. The data comes from the Kepler Project's Threshold Crossing Event. The system makes use of several layers of feature extraction and dimensionality reduction. This metric uses Locality Preserving Projections (LLP) dimensionality reduction and KNN to determine whether a given signal is sufficiently like known transits in the same data set. Changing the parameters (k, N, n) did not yield significant changes to the outcome, and the chosen values were determined by removing non-transit like signals and preserving transit-like signals. K-Nearest Neighbors was used for training. The authors found that there was a slight bias towards incorrectly classifying the signals of some of the smaller planets. [11]

Criticism

Not performing validation and normalization could have given them inaccurate results. No evaluation methods were mentioned. Using *10-fold cross validation*, repeated 10 times, for evaluation would have given far more accurate evaluation results. The classes had an imbalance in size, and introducing *artificial balancing* could potentially remove any bias found in the results.

Agrawal et al. focused on the applicability and ability to produce a desirable output of various machine learning algorithms such as KNN, Decision Trees (DTs), Random Forests (RFs), SVMs, Naïve Bayes, Linear Discriminant Analysis (LDA) in analysis and inference of decision theoretic problems in Astronomy. The authors created a software called ExoPlanet to reduce the programming overheads in research involving data analytics. The software provides a GUI to select a data set, and then a method (classification, regression, and clustering) of choice can be selected. The results (accuracy, sensitivity, specificity, etc.) and all necessary graphs (ROC, etc.) are displayed in the same window. The data was retrieved from Planetary Habitability Laboratory, University of Puerto Rico's Exoplanet Catalogue (PHL-EC), and PCA was used for feature extraction. Any missing features were accounted for by taking the mean for continuous features and the mode for categorical features in the dataset. Data includes features like Atmospheric type, mass, radius, surface temperature, escape velocity, earth's similarity index, flux, orbital velocity, etc. Initially they carried out a ten-fold cross validation, with one test-bin and 9 bins for training. The datasets were then tested on the 6 classification algorithms mentioned above. For their analysis, they found that the PHL-EC catalogue they were relying on was missing a lot of data, and there a large bias towards the Non-Habitable class, and Psychro-planets (planets with surface temperatures ranging from -50 to 0C) and Meso-planets (planets with diameters ranging from 1000 to 5000km) had an insufficient amount for an effective classification. The sensitivity and specificity using this method were very close to 1 for all classifiers. They found all methods returned unbelievably high accuracies, and they believe it's due to the heavy

bias towards the Non-Habitable class. To counter the problems faced with data bias, they performed another test where smaller datasets were constructed by selecting all planets belonging to Meso-planet and Psycro-planet classes and selecting 10 planets which belonged to the habitable class at random, resulting in 26 planets in a smaller, artificially balanced data set. For each iteration of testing, they selected one entity from Meso-planets and one from the Psychro-planets and two from non-habitable, and the remaining were used as training data for that cycle, using all combinations of training and testing overall. The results of their second analysis showed that Random Forest and decision trees rank best with highest accuracy closely followed by Naïve Bayes. They achieved a higher accuracy with ML by performing artificial balancing on the data, which told them that Deep Learning isn't necessary in every difficult classification scenario. [12]

Criticism

The discovery of bias in their system and using *artificial balancing* to fix it will be strongly considered when we are building our systems. They used PCA as a feature extraction method, when **Abdelmajed et al.** proved that LPP performs better than PCA. No validation technique was mentioned, and using one could have improved results. *10-fold Cross Validation* could have been repeated 10 times (for a total of 100 epochs) to further improve evaluation results. [10]

Karim's aims were to use machine learning and photometric timeseries data to detect exoplanets. The data used processed Target Pixel Files (TPFs), also known as light curve files. The features were normalized to a range of -1 to 1 using a form of Min-Max. They created a labelled dataset of 3 classes: *Candidate* (if the object was originally labelled as *Confirmed* or *Candidate*), *False Positive* (if the object was originally labelled *False Positive*), and *No Signal* (If the object was not a KOI). Before they split the data into training, validating, and testing sets, they combined *Candidate* and *False Positive* into one class called *Candidate*, leaving just two classes. For training the author used 70% of the data, for validation and testing they used 15% of the data each. Classes were divided evenly

between the 3 sets. For training they used a Binary Sequence Classification, which contained many 1D Convolution layers, a Max Pooling Layer, a Global Average Pooling Layer, a Dropout Layer, and finally a Classifier Layer. For calculating the Loss Function, they used Binary Cross-Entropy, and for Optimization they used Stochastic Gradient Descent. The author also had additional DNN implementations to distinguish between *False Positives* and *Candidates*. One of them was a Multi-Class Classification with Multilayer Perceptron (MLP) which used a Dropout between each layer. Another implementation was Multi-Class Classification with 1D Convolution which used RELU for its activation function. Finally, they also implemented a Multi-Class Classification with Long Short-Term Memory (LSTM) which used Recurrent Neural Networks (RNNs) to make predictions based on what happened in the recent past. The results of the Binary Classifier showed the training accuracy at 75% and the validation accuracy peak at 73%, then drop to around 72%, which the author suspected could be due to the trainer beginning to overfit the data, and any further training might decrease the accuracy. After evaluation, the accuracy was 73.3%. *Figure 1* shows the results of the Multi-Class implementations.

Model	**Training Accuracy**	**Validation Accuracy**	**Evaluation Accuracy**
MLP	59.20%	55.80%	58.20%
1D Convolution	62.60%	61.30%	58.20%
LSTM	60%	51%	52.80%

Figure 5: Multi-Class implementation results. Source: [13]

With these results, the author theorized that the algorithms may not have understood the data properly. [13]

Criticism

Full Frame Images (FFIs) could have been used to produce microlensing data along with the TPF light curve files they used, which could potentially improve accuracy results. There was a slight imbalance of classes in the dataset (60% of the data was *Candidate*, 40% of the data was *No Signal*). Performing artificial balancing on the dataset before training could have slightly improved training results. Also for validation they could have used repeated k-fold cross-validation, where k = 10 which could have

given better validation accuracy, for it trains with all combinations instead of training with random combinations. No Normalization or Validation methods were mentioned.

This is the end of the 'Machine Learning with Light curves' section. A section on criticizing all of these papers as a group can be found here.

Oscillations

In this section, we look at papers that attempted to use oscillation frequencies of stars with machine learning techniques.

Davies et al. attempted to estimate mode frequencies and modes of oscillation via Bayesian Markov Chain Monte Carlo (MCMC) framework. To do this, they used a sample of Kepler solar-type stars that host exoplanets that produce transits. They also attempt to improve the quality using Bayesian Unsupervised Machine Learning. They evaluated their results using measured frequencies of the modes of oscillation, common frequency ratios and the covariance of frequencies measured and calculated. These variables were then used to draw constraints on the parameters of the exoplanet stars. They first defined prior probabilities of the solar-types stars parameters, which helps constrain the parameters and removes any outputs that do not adhere with the asymptotic theory. To avoid unwanted correlations arising, they give their own magnitude in their covariance matrix. They tried to improve test results by adding stellar rotation properties to improve detection at low signal-to-noise ratios. They evaluated their test using a Bayesian framework. Their results estimated 68% confidence intervals as the stander deviation of the Markov Chain. [14]

Criticism

The authors did not test their theory using other methods like Naïve Bayes, which could have resulted in better results or provided more information regarding the constraints on the parameters of planet-hosting stars.

Microlensing.

In this section, we look at papers who used microlensing with machine learning techniques.

Wei Zhu et al. show their solution to performing reduction on Kepler 2 Campaign 9's microlensing data, which is known to be difficult because of its crowded field and the unstable pointing of the spacecraft. They applied differential photometry to the Kepler 2 Campaign 9 (K2C9) microlensing data set and develop modelling techniques that can properly extract the microlensing signals, and applied their method to two example microlensing events and discuss their method and its implications. They also derived precise $K_p - I$ vs $V - I$ color-color relations which can predict flow in the Kepler bandpass. The authors also show that the microlensing parameters can be better controlled by implementing the color-color relation in the light-curve model. The authors first retrieved the Target Pixel Files (TPFs) and combined them to form Sparse Full Frame Images (SFFIs), which were then cross-checked with full-frame images from Campaign 9. They took the median of the frames that have approximately the same offset to form the Master Photometric Reference Frame. The authors then extracted raw light-curves from the difference images and the master photometric reference images. They then simultaneously modelled the microlensing and the systematic effects. They derived the relation between $K_p - I$ and $V - I$ that applies to the source stars of K2C9 microlensing events. K_p is the Kepler Magnitude (stellar magnitude of the Kepler bandpass), and V and I are the primary band passes used in ground-based microlensing surveys. Combining V, I, and K_p covers the broad K_p bandpass, making the derived color-color relation less sensitive to the details of the stellar spectrum and the interstellar extinction. Using this color-color relation, they were able to predict the microlensing source flux in the K_p based on the known $V - I$ color. They used the predicted $K_p - I$ color to validate the result of the light curve modelling. They modelled the K2C9 data and the Optical Gravitational Lensing Experiment data simultaneously, followed by the detrending and microlensing modelling. Detrending results in significantly improved photometry while preserving the physical signals. They found a significant trend whose value is close to the expected peak of the microlensing event. Next, they simultaneously modelled the detrending terms

and the microlensing signal. The authors found that the derived best-fit parameters and their uncertainties were significantly reduced. [15]

Criticism

The authors used a photometric technique that uses a large number of bright sources to derive the astrometric solutions. This method cannot be applied to objects that are outside the super stamp region. Therefore, additional techniques might be required to get improved accuracy of light curve extraction.

Criticism of Machine Learning with Light Curves

Along with having a criticism for each paper, we have prepared a section that compares all of the papers from the "Machine Learning with Light Curves" section. *Figure 2* is a table we made to compare the different goals and methods mentioned in their papers.

Reference No.	Goal	Data Source	Feature Extraction	Normalization	Labelled Dataset Classes	Learning Algorithm	Validation	Evaluation Method
[8]	Present Neural Networks as a method of detecting exoplanets	Artificially Generated	None Mentioned	Normalized to unit variance. The mean was subtracted from the prior.	"Transit" and "Non-Transit" Class sizes not mentioned.	Neural Networks, Restricted Boltzmann Machines (RBM), RELU activation layer	Data was validated using Kepler data	Cross-Validation
[9]	To find an optimal algorithm for transient classification	Sloan Digital Sky Survey (SDSS)- II survey	PCA	All features have a Standard deviation of 0.5 and mean of 0.	15,521 "Real Objects", 11,595 "Not Real"	RF, kNN, SkyNet, SVM, MEC, NB	Min-Max	Cross-Validation
[11]	Presents Feature Reduction and kNN as ways to compare signals with transits in a dataset	Kepler Mission's Data Release 24 KOI Catalog Data Validation files	LPP	None Mentioned	5678 "Transit-Like", 1039 "Non Transit-Like"	kNN	None mentioned	None mentioned
[12]	To find the optimal algorithm from kNN, DT, RF, SVM, NB, LDA for classifying types of exoplanets	Planetary Habitability Laboratory, University of Puerto Rico	PCA	None Mentioned	"Mesoplanet", "Psychroplanet", "Non-habitable"	kNN, DT, RF, SVM, NB, LDA	None mentioned	10-fold cross validation with artificial balancing
[13]	Use machine learning and light curves to detect transiting exoplanets.	Kepler Mission's Data Release 25 KOI Catalog Data Validation files	LPP	None Mentioned	7068 "Candidate", 9989 "No Signal"	Binary Sequence Classifier	None mentioned	None mentioned

Figure 6: Table showing the 5 papers under the topic "Machine Learning with Light Curves". The *Reference No.* shows the IEEE reference number.

In all the papers in *Figure 6*, their *Goals* are about solving classification problems with the highest degree of accuracy possible. Papers [9] and [12] compare algorithms' performances against each other to find the one that performs the best. The only reason we have a *Data Source* section in the table is [8] artificially generated its own data. Even though the authors of [8] validated their dataset using the Kepler data, I strongly feel that had they used *real* Kepler data they would have gotten different results. For *Feature Extraction* (or *Dimensionality Reduction*), all papers used either PCA or LPP. [8] didn't mention any methods used. According to **Abdelmajed et al.** in [10], experiments showed them that LPP performed better than PCA, and therefore [9] and [12] should have used LPP instead of PCA. Most papers did not mention any *Normalization* methods used, which is not good as not having *Normalization* can give inaccurate results. The reason why we have a *Labelled Dataset Classes* section is to show all the classes they used, and the number of objects in each class. Papers

[9], [11], and [13] have imbalanced class sizes (which is normal to have), and did not use any form of *artificial balancing* to compensate. To recap, *Artificial Balancing* [12] is where all classes used in training have the same number of objects. After every epoch, the datasets are shuffled so that the excluded objects can be trained on. This was shown in [12] when the authors ran their first test and found a bias towards the *Non-Habitable* class, and using *artificial balancing* in their second test removed most of that bias. Therefore, not using it could results in a bias towards the class that has the most number of objects in it. Many papers used different *Learning Algorithms*, and some papers used many at once to compare their performance. From all the algorithms used, *Random Forests* and *Decisions Trees* were the most successful and popular. Most papers didn't mention any *Validation* technique used, and using one could have improved their performances drastically. Of all the *Evaluation* techniques used, *k-fold Cross Validation* (where k = 10) produces the most accurate evaluation results. However, repeating the *10-fold Cross Validation* process 10 times (for a total of 100 epochs) would have improved the accuracy even more.

Now that we have evaluated these papers as a group, we shall show our conclusion in the next section.

Conclusion.

In conclusion, these papers have shown different algorithms with different precisions and accuracies, and different techniques to optimize results. *Figure 7* shows the evaluation results on the best techniques used.

Feature Extraction	**Learning Algorithms**	**Evaluation**
LPP	RF and DT	10-fold Cross validation, with artificial balancing

Figure 7: Best techniques used in "Machine Learning with Light Curves" section.

Given our research, these methods in *Figure 3* are best ones used in its section, and will be strongly considered when building the system. In the next chapter, we look at our Requirements Analysis

Chapter 3: Requirements Analysis and Evaluation

In this chapter, we discuss our Requirements and how we are going to evaluate each of them.

Requirements

Figure 8 shows our table of requirements.

RID	FR/ NFR	Type	Requirement
FR-SR1	Functional	System	Our tool must be able to extract and process the Kepler data.
FR-SR2	Functional	System	Our tool must be able to perform appropriate transformations, such as Feature Selection and Normalization.
FR-SR3	Functional	System	Our tool must be able to generate a dataset based on the Kepler data, and validate it.
FR-SR4	Functional	System	Our tool must be able to train on the dataset
FR-SR5	Functional	System	Our tool must be able to evaluate the training and produce graphs based on the results
NRF-SR1	Non-functional	System	Previous work on this topic by [16] have yielded an evaluation accuracy of 73.3%. We aim for our tool to reach an accuracy that is greater or equal to it.
NFR-UR1	Non-functional	User	The system must be cohesive. Parts of the system can be re-run without having to re-run the entire program.

Figure 8: Our table of requirements. The 2nd row shows whether it is a functional or non-functional requirement. The 3rd row shows whether it is a user or system requirement.

In the next section, we show how these requirements will be evaluated.

Testing and Evaluation

Figure 9 shows a table with an evaluation strategy for the requirements defined earlier.

RID	Evaluation Strategy
FR-SR4	We will test this by performing evaluation on the model
FR-SR5	We will be showing all graphs produced in the next Deliverable
NRF-SR1	We will test this by using one of (or both) the methods, *K-fold Cross-Validation* and *Confusion Matrix.*
NFR-UR1	We will be using JuPyTer, an IDE capable of having code in multiple 'cells', and each of them can be run independantly.

Figure 9: A table that shows all the testable requirements and their evaluation methods

K-Fold Cross Validation

The system will be tested based on its training, validating, and evaluating accuracy. The accuracy will be tested by performing a k-fold Cross Validation repeated 10 times (where *k* = number of parts and number of epochs). K-fold Cross Validation splits the dataset into *k* equal subsets. In each epoch, each subset gets a turn to be the testing set, while the rest are part of the training set, resulting in all combinations used as training and testing. In the end, the best result is selected from all the epochs.

Confusion Matrix

A confusion matrix can be used to show and compare predicted results versus actual results, and shows the false positive and false negative rate. Through this, we can calculate the *success* rate and the *error* rate. The error rate can be misleading, which is why we will calculate *Sensitivity* and *Specificity*. *Specificity* shows the true negative rate, and *Sensitivity* shows the true positive rate. We can then use them to find out the *Precision*, which shows which the true positive rate out of all the positives, and the *Recall* (aka Sensitivity), which can be used to calculate the *F-Measure*.

In the next chapter, we will be looking at how we plan to implement this system, how we plan to manage our workflow, and our plan to mitigate any possible risks.

Chapter 4: Project Management.

In this chapter, we look at our plan for this project. We look at our architecture, our workflow, and the risks involved. *Figure 10* shows the planned architecture for our system. It shows the different components, how they are integrated together, and the flow of information between them.

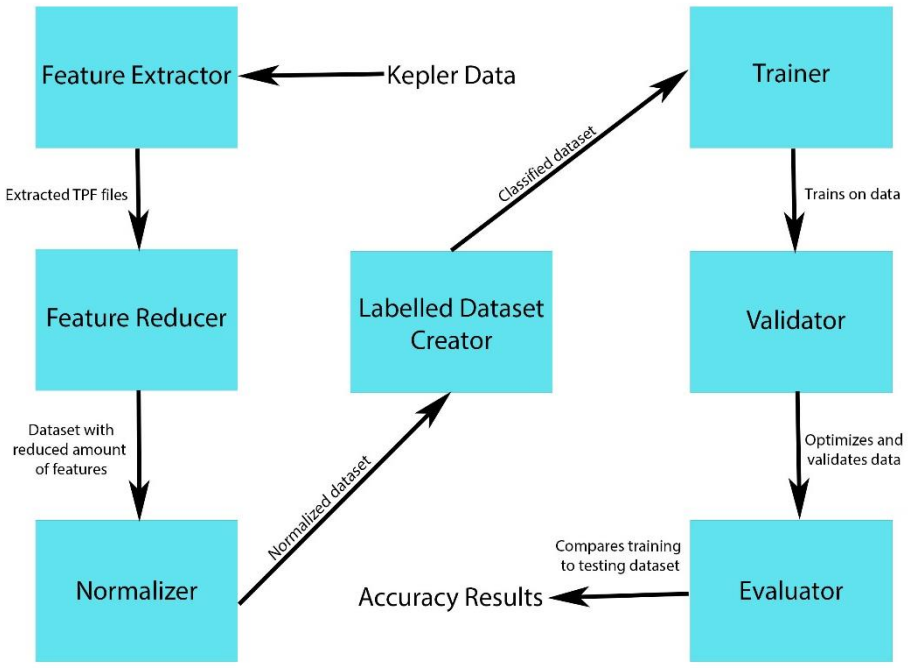

Figure 10: Planned architecture of system.

We will have a *Feature Extractor*, that takes in the Kepler data and extracts the TPF files. The *Feature Reducer* then selects the optimal parameters and discards the rest. The *Normalizer* scales the data down. Then, the *Labelled Dataset Creator* adds labels to our dataset. It will also split the dataset into subsets for training, validating, and testing. The *Trainer* then trains the model using the training subset. The model will be validated by the *Validator* using the validation subset. Finally, the *Evaluator* will evaluate the model using the evaluation subset. We can either have the graph-producing component in the *Evaluator*, or as a separate component.

In the next section, we look at how we plan to organize the workflow for this project.

Methodology.

For this project, I have decided to use an Agile methodology, Scrum, with a few changes. The supervisor will act as the Product Owner. They will also partially act as Scrum Master, as they will advise the student based on the progress of the project, while the student maintains the Burndown Charts and Product Backlogs. The student acts as the development team. Different requirements of the project are implemented in "Sprints", which are shorts bursts where the development team complete the assigned task such that it's in "shippable" condition. A Product Backlog is a list of tasks that need to be completed by the development team. Tasks are taken from the Product Backlog and are assigned to Sprints. Weekly "Scrum meetings" must be held with the student and the supervisor. In this meeting, the student updates the professor on the project's progress and any issues that the student has been facing. They will also discuss if any tasks need to be reprioritized. [16]

Figure 11 shows our 1st half of the Gantt Chart, ranging from September to December. *Figure 12* shows our 2nd half of the Gantt Chart, ranging from December to April.

ID	Task Mode	Task Name	Duration	Start	Finish
1		Gather Research Papers	3 days?	Wed 9/20/17	Fri 9/22/17
2		Prepare Literature Review	38 days	Mon 9/25/17	Wed 11/15/17
3		Prepare Requirements	1 day?	Wed 11/1/17	Wed 11/1/17
4		Prepare Evaluation and Testing Methods	1 day?	Thu 11/9/17	Thu 11/9/17
5		Prepare Project Management	1 day?	Mon 11/13/17	Mon 11/13/17
6		Prepare Risk Management	1 day?	Wed 11/15/17	Wed 11/15/17
7		Compile Deliverable 1 Report	1 day	Wed 11/22/17	Wed 11/22/17
8		Draft Deliverable 1	1 day	Thu 11/23/17	Thu 11/23/17
9		Submit Draft Deliverable 1 to Supervisor, making needed improvements	1 day	Sat 11/25/17	Sat 11/25/17
10		Deliverable 1 Submission	0 days	Mon 11/27/17	Mon 11/27/17
11		Period for other coursework and exams	12 days	Tue 11/28/17	Wed 12/13/17
12		Meet with Supervisor and 2nd Reader	5 days?	Thu 12/14/17	Wed 12/20/17
13		Make improvements based on feedback	3 days?	Thu 12/21/17	Mon 12/25/17

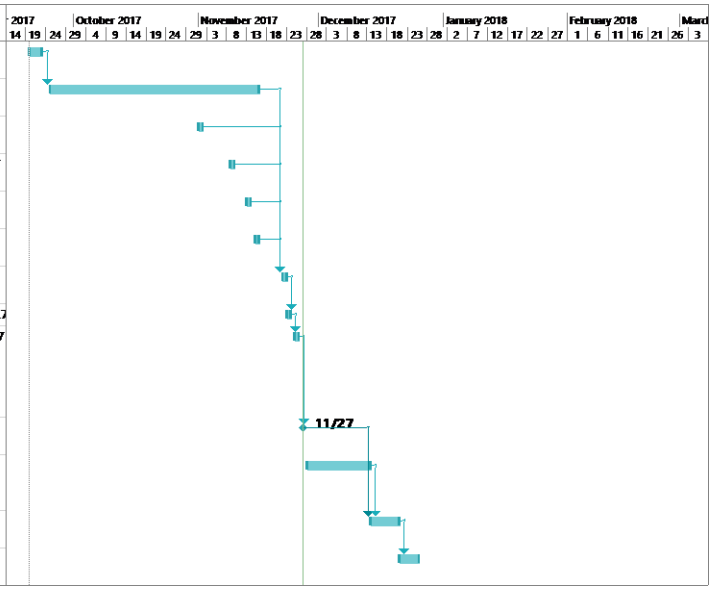

Figure 11: Gantt Chart (1/2) September - December

ID	Task Mode	Task Name	Duration	Start	Finish
14		Identify useful features in Kepler Data	7 days?	Thu 12/14/17	Fri 12/22/17
15		Research on Kepler Feature Extraction Methods	7 days?	Mon 12/25/17	Tue 1/2/18
16		Develop Feature Extraction Component	5 days?	Wed 1/3/18	Tue 1/9/18
17		Research on Feature Reduction Methods	7 days?	Wed 1/10/18	Thu 1/18/18
18		Develop Feature Reduction Component	5 days?	Fri 1/19/18	Thu 1/25/18
19		Research Normalization Methods	7 days?	Fri 1/26/18	Mon 2/5/18
20		Develop Normalization Component	3 days?	Tue 2/6/18	Thu 2/8/18
21		Create Labelled Dataset	4 days?	Fri 2/9/18	Wed 2/14/18
22		Develop component to split the dataset into training, validating and testing	4 days?	Thu 2/15/18	Tue 2/20/18
23		Develop Training Component	5 days?	Wed 2/21/18	Tue 2/27/18
24		Research Validation Methods	7 days?	Wed 2/28/18	Thu 3/8/18
25		Develop Validating Component	3 days?	Fri 3/9/18	Tue 3/13/18
26		Develop Testing Component	3 days?	Wed 3/14/18	Fri 3/16/18
27		Perform tests on the labelled dataset	3 days?	Mon 3/19/18	Wed 3/21/18
28		Create Graphs, Tables, etc.	2 days?	Thu 3/22/18	Fri 3/23/18
29		Put together notes of work carried out so far	7 days?	Mon 3/26/18	Tue 4/3/18
30		Put together notes from Evaluation and Results	3 days?	Wed 4/4/18	Fri 4/6/18
31		Start writing Deliverable 2	7 days?	Mon 4/9/18	Tue 4/17/18
32		Compile Deliverable 2 Report	1 day?	Wed 4/18/18	Wed 4/18/18
33		Submit Draft Deliverable 2 to Supervisor	1 day?	Thu 4/19/18	Thu 4/19/18
34		Improve Draft with suggestions from Supervisor	1 day?	Fri 4/20/18	Fri 4/20/18
35		Deliverable 2 Submission	0 days	Mon 4/23/18	Mon 4/23/18
36		Create Poster	2 days?	Mon 4/23/18	Tue 4/24/18
37		Submit Poster	0 days	Wed 4/25/18	Wed 4/25/18
38		Present Poster	0 days	Wed 4/25/18	Wed 4/25/18

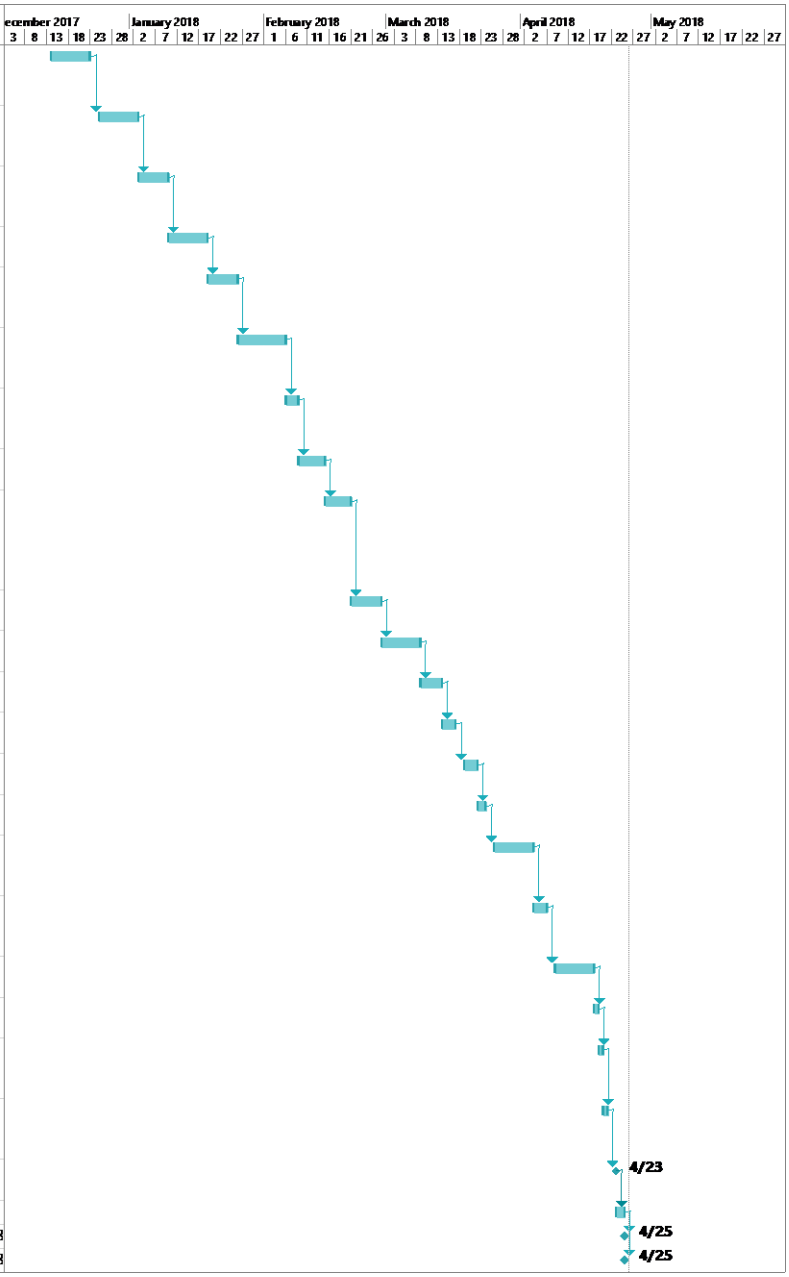

Figure 12: Gantt Chart (2/2) December – April

In the next section, we look at the professional, legal, ethical, and social issues.

Professional, Legal, Ethical, and Social Issues

I confirm that my project does not involve any human subjects, personal data of people alive, sensitive personal data, or confidential data. The Kepler and K2 mission data that I will be using for this project are 'open access data'. My project does not need approval from another body. The project does not need anything more than standard IT equipment to perform machine learning and

it poses no threat to anyone involved. We filled the ethics form online with the details mentioned above and it was approved.

In the next section, we will look at the different potential risks on our project and our plan to combat them.

Risk Management.

In Risk Management, we consider the possible risks and assess how each of their impacts can be minimized. Listed below are the 4 steps that were performed for each of the threats:

Risk Identification: This stage identifies potential threats to the project and the type of threat they pose.

Risk Analysis: This stage draws up the chance of occurrence and level of impact for each of the risks.

Risk Planning: This stage develops different levels of strategies for each threat.

Risk Monitoring: This stage lists possible signs or indications of a risk happening in the future.

Risk Identification.

This section lists the possible threats to the project. Risks fall into 5 categories:

Technology: These are issues related to devices, implementing them, the programs they run, the data they store, and their overall performance.

People: These are issues related to the human aspect, such as their health condition, their abilities and skills, and their availability.

Tools: These are issues related to tools implemented in the project such as their relevance, compatibility, and efficiency.

Organizational: These are issues related to the organization the project is meant for. These issues could be systematic issues, or related to the people who are part of the system.

Requirements: These are issues related to the foundation of the project and what it's meant to do. Issues could be with the requirements defined in the beginning of the project, or with the customer having an issue with them.

Estimation: These are issues with estimating the project in general, such as the time taken to complete certain milestones and the size of certain parts of the project.

Figure 13 shows the color code and their associated priority.

Color Code			
Priority	Low	Medium	High

Figure 13: Color code and their priority. Note that this Legend should be used with the *R#* found in the following tables.

Figure 14 shows each of the derived risks and their associated risk type.

R#	Risk	Risk Type
1	Lack of processing power	Technology
2	Hard Drive Failure/Device crashes	Technology
3	Data Corruption	Technology
4	Unsure how to implement the system	Technology
5	Unable to meet deadlines	People
6	Communication issues between supervisor and student	People
7	The supercomputer is unavailable for use	Tools
8	The student is new to using certain to certain tools	Tools
9	Deadlines from other courses at the same time	Organizational
10	Supervising professor is away/unavailable	Organizational
11	Sudden change of requirements	Requirements
12	Too many extracted features	Estimation

Figure 14: Risk Identification, shows risks and their types

Risk Analysis.

In this section, we rank the possibility of the risk. *Figure 15* shows the terms and color codes used for the levels of possibility and impact.

Possibility				**Impact**		
Low	Moderate	High		Tolerable	Serious	Catastrophic

Figure 15: Possibility and Impact Scale.

R#	Risk	Possibility	Impact
1	Lack of processing power	High	Medium
2	Hard Drive Failure/Device crashes	Medium	Low
3	Data Corruption	Low	High
4	Unsure how to implement the system	Low	High
5	Unable to meet deadlines	Medium	Low
6	Communication issues between supervisor and student	Medium	Low
7	The supercomputer is unavailable for use	Medium	Medium
8	The student is new to using certain to certain tools	High	High
9	Deadlines from other courses at the same time	Medium	Low
10	Supervising professor is away/unavailable	Medium	Medium
11	Sudden change of requirements	High	High
12	Too many extracted features	Medium	High

Figure 16: Risk Analysis, shows the risks and their possibility and impact levels.

Risk Planning.

Here, we consider different types of strategies for each situation.

Avoidance: Reduces the chance the threat will occur.

Minimization: Reduces the impact of the risk on the project.

Contingency: Deals with the threat if it occurs.

R#	Risk	Avoidance	Minimization	Contingency
1	Lack of processing power	Either, use the university's supercomputer in Edinburgh, or follow the contingency for risk 7.	Make sure to properly perform Reduction, or use better data reduction methods	Use batch training/processing, utilize more cores, or request for the supercomputer
2	Hard Drive Failure/Device crashes	Don't have other applications running during execution as it could overload the RAM.	Carefully investigate the required specifications and make frequent saves and backups.	Use available backups, request supercomputer for usage
3	Data Corruption	Make backups of data on an external device, and perform operations on a copy of the data.	Use a copy of the original data, leaving the original as a backup.	Make use of any back ups to reduce the impact.
4	Unsure how to implement the system	The student should do their own research about available tools.	The supervisor can recommend tools to the student.	The supervisor and the student should have a meeting where they discuss strategies on which tools to use.
5	Unable to meet deadlines	Break down bigger tasks into smaller, simpler tasks that can be completed on time.	Contact the supervisor at the first sign that the task isn't doable.	The supervisor could suggest an alternative, or decide which tasks could be done later or cut from the project.
6	Communication issues between supervisor and student	Keep in touch with the supervisor regularly.	Create weekly reports, which can be discussed about during meetings.	Hold a meeting between the student and the professor to clear any doubts regarding the status of the project.
7	The supercomputer is unavailable for use	Check with the university as early as possible about using the supercomputer	Reduce the amount of data being processed by reducing the amount of features selected.	Run the program locally, using gpu and batch processing/training
8	The student is new to using certain to certain tools	Supervisor and student must decide on tools that the student is familiar with, or easier to learn.	Student should attempt to learn to use the tools.	Supervisor can help the student with difficult tasks
9	Deadlines from other courses at the same time	Break down larger tasks into smaller tasks that are easier to accomplish.	Supervisor should be made aware of clashing deadlines, who could help reprioritize tasks.	Supervisor could look into altering deadlines, or advise the student on their next course of action.
10	Supervising professor is away/unavailable	Meetings should be planned when the supervisor and student are available.	Tasks should be assigned in advance so that the student isn't lost when the supervisor is away.	Meetings should be rescheduled.
11	Sudden change of requirements	Keep frequent meetings so that both parties are updated on any developments.	Hold frequent meetings to help clear any misinterpretations of the requirements.	A meeting could be held to determine whether the requirements are feasible as they are.
12	Too many extracted features	Select only the best features that are known give positive results.	Use information Gain, Gini Impurity, or other methods of Feature Ranking to help understand which features are better than others.	Use alternative methods of feature selection.

Figure 17: Risk Planning, shows all the risks and their associated Avoidance, Minimization, and Contingency Strategies.

Risk Monitoring.

In this section, *Figure 18* shows each of the risks, and their associated strategy for monitoring the.

This allows us to foresee threats before they happen and take action before the threat hits.

R#	Risk	Monitoring
1	Lack of processing power	Keep a track of the device's performance and how much data it is expected to process.
2	Hard Drive Failure/Device	Make notes on the device's performance, if it starts to slow down or hang.
3	Data Corruption	Keep watch of the device's performance and if it begins to slightly falter.
4	Unsure how to implement the	The project's progress is very slow, or the student is spending too much time researching for tools.
5	Unable to meet deadlines	Use the project plan to keep track of the project and observe for any signs of falling behind. Keep in touch with the professor, who will give suggestions on if the project development needs to speed up.
6	Communication issues between	Hold regular meetings and make sure there is no confusion regarding the project.
7	The supercomputer is	Keep contact with the university's IT department in UK.
8	The student is new to using	Project progress can be monitored during weekly meetings
9	Deadlines from other courses at	Supervisor could look into altering deadlines, or advise the student on their next course of action.
10	Supervising professor is	Tasks should be assigned in advance so that the student isn't lost when the supervisor is away.
11	Sudden change of requirements	Keep frequent meetings so that both parties are updated on any developments.
12	Too many extracted	Both parties need to be clear on what their expectations are.

Figure 18: Risk Monitoring, shows each of the risks and their monitoring strategies

References.

[1] "AI could be the perfect tool for exploring the Universe", *The Verge*, 2017. [Online]. Available: https://www.theverge.com/2017/11/15/16654352/ai-astronomy-space-exploration-data.

[2] Guresen, E. and Kayakutlu, G. (2011). Definition of artificial neural networks with comparison to other networks. *Procedia Computer Science*, [online] 3, pp.426-433. Available at: https://ac.els-cdn.com/S1877050910004461/1-s2.0-S1877050910004461-main.pdf?_tid=3ef5e3b4-cba6-11e7-86bc-00000aacb361&acdnat=1510930234_d13fd21ef4041866f8ba9e8aefea70cf

[3] E. Rieuf, *A basic representation of a neural network*. 2017. Available at: https://www.datasciencecentral.com/profiles/blogs/a-simple-neural-network-with-python-and-keras

[4] C. Elkan, "Evaluating classifiers", *University of San Diego, California, retrieved [01-11-2012] from http://cseweb. ucsd. edu/elkan B*, vol. 250, 2012.

[5] "Importance of Feature Scaling — scikit-learn 0.19.1 documentation", Scikit-learn.org, 2017. [Online]. Available: http://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html.

[6] "Institute for Astronomy Press Release", *Ifa.hawaii.edu*, 2008. [Online]. Available: http://www.ifa.hawaii.edu/info/press-releases/JohnsonDec08/.

[7] "NASA - Cosmos Provides Astronomers with Planet-Hunting Tool", *Nasa.gov*, 2004. [Online]. Available: https://www.nasa.gov/vision/universe/newworlds/microlensing.html.

[8] K. Pearson, L. Palafox and C. Griffith, "Searching for Exoplanets using Artificial Intelligence", *Monthly Notices of the Royal Astronomical Society*, 2017.

[9] L. du Buisson, N. Sivanandam, B. Bassett and M. Smith, "Machine learning classification of SDSS transient survey images", *Monthly Notices of the Royal Astronomical Society*, vol. 454, no. 2, pp. 2026-2038, 2015.

[10] A. Abdelmajed, "A Comparative Study of Locality Preserving Projection and Principle Component Analysis on Classification Performance Using Logistic Regression", *Journal of Data Analysis and Information Processing*, vol. 04, no. 02, pp. 55-63, 2016.

[11] S. Thompson, F. Mullally, J. Coughlin, J. Christiansen, C. Henze, M. Haas and C. Burke, "A MACHINE LEARNING TECHNIQUE TO IDENTIFY TRANSIT SHAPED SIGNALS", *The Astrophysical Journal*, vol. 812, no. 1, p. 46, 2015.

[12] S. Agarwal, S. Basak, S. Saha, M. Rosario-Franco, S. Routh, K. Bora and A. Jeremiel Theophilus, "Machine Learning Exploration via Mining and Automatic Labeling of the Habitability Catalog", A Comparative Study in Classification Methods of Exoplanets, 2016.

[13] K. Karim, "Detecting Exoplanets Using Machine Learning Techniques", Masters Thesis, Heriot-Watt University, 2017.

[14] G. Davies, V. Aguirre, T. Bedding, R. Handberg, M. Lund, W. Chaplin, D. Huber, T. White, O. Benomar, S. Hekker, S. Basu, T. Campante, J. Christensen-Dalsgaard, Y. Elsworth, C. Karoff, H. Kjeldsen, M. Lundkvist, T. Metcalfe and D. Stello, "Oscillation frequencies for 35Keplersolar-type planet-hosting stars using Bayesian techniques and machine learning", *Monthly Notices of the Royal Astronomical Society*, vol. 456, no. 2, pp. 2183-2195, 2015.

[15] W. Zhu, C. Huang, A. Udalski, M. Soares-Furtado, R. Poleski, J. Skowron, P. Mróz, M. Szymański, I. Soszyński, P. Pietrukowicz, S. Kozłowski, K. Ulaczyk and M. Pawlak, "Extracting Microlensing Signals from K2 Campaign 9", *Publications of the Astronomical Society of the Pacific*, vol. 129, no. 980, p. 104501, 2017.

[16] M. James, "An Empirical Framework For Learning (Not a Methodology)", Scrummethodology.com, 2017. [Online]. Available: http://scrummethodology.com/.