CLASSIFYING EXOPLANETS AND HABITABILITY WITH KEPLER DATA

By Karl Eknes

CONTENTS

1. Introduction

Have you ever looked up into a cloudless night sky and got captivated by all the twinkling stars? Wondered how vast space really is, and how small we are in comparison to the universe? Have you ever asked yourself what lies behind those distant lights? Could one of them be home to another Earth, a planet like ours, with oceans, and clouds? Maybe even life? Or is it just endless balls of gas and barren rocks, drifting in the dark?

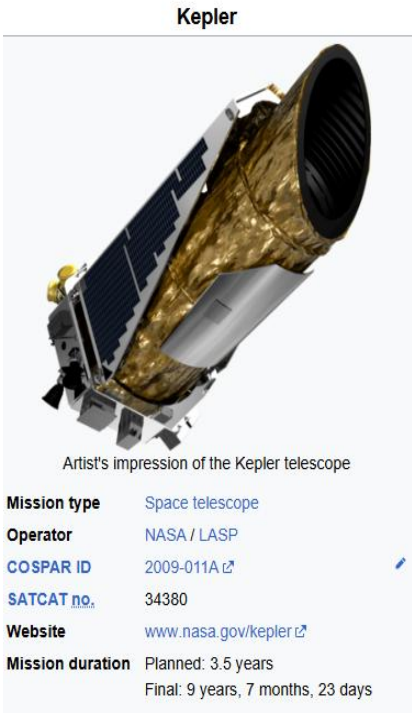
For centuries, humanity has asked these questions. However, it wasn't until 2009 that NASA decided to take a leap in the search for answers by launching the Kepler Space Telescope into orbit (NASA E. S., 2022). Kepler's mission was to find planets beyond our solar system, so-called exoplanets, and to see if any of them might be capable of supporting life.

The Kepler Space Telescope used a method called the transit method (NASA S. , 2023). It works by observing a dip in a star's brightness when a planet passes in front of it. Kepler stared at over 150,000 stars (Jet Propulsion Laboratory, California Institute of Technology, 2015). By the time its mission ended, Kepler had confirmed more than 2,600 exoplanets.

In this project, we dive into the vast Kepler dataset that can be found in the NASA Exoplanet Archive (Institute, 2024), and in Kaggler.com. We will look at the dataset from NASA and explore its information using exploratory data analysis (EDA) and machine learning. By analysing planetary characteristics such as size, orbit, and the type of star they orbit, and maybe, just maybe we can find potential planets that is like our own.

2. Background

2.1 EXOPLANET DISCOVERY AND KEPLER'S MISSION



Kepler

Artist's impression of the Kepler telescope

Mission type	Space telescope
Operator	NASA / LASP
COSPAR ID	2009-011A
SATCAT no.	34380
Website	www.nasa.gov/kepler
Mission duration	Planned: 3.5 years Final: 9 years, 7 months, 23 days

Figure 1: https://en.wikipedia.org/wiki/Kepler_space_telescope

An exoplanet, or extrasolar planet, is a planet located outside of the solar system and orbits a star. The first discovery and confirmation of an exoplanet was during the 1990s, but the field of exoplanets made a huge leap compared to prior discoveries with the launch of Kepler by NASA. Kepler confirmed that there are planets everywhere, and that they are not rare (NASA, 2023). The Kepler Space Telescope was launched in 2009.

The primary method to detect exoplanets was the transit method, where the slight dimming of a star's light indicates a planet passing between the star and the telescope (NASA Science, 2024). The Kepler mission revolutionized the understanding of planetary systems, demonstrating that planets are common around stars and that small, potentially rocky planets are plentiful.

Figure 3: Johannes Kepler - https://www.historyoftelescope.com/telescope-invention/the-keplerian-telescope/

The Kepler Space Telescope is named after Johannes Kepler. An astronomer and a mathematician who lived between 1571 to 1630. He is also known for inventions, like the Keplerian Telescope.

2.2 THE CONCEPT OF HABITABILITY

Habitability is the reference to a planet being able to harbor life as we understand it. One fundamental aspect of habitability is the planet's location orbiting in the habitable zone of its star, which scientists often nickname as a “Goldilocks zone” where temperatures are not too cold, or not too hot, but instead somewhere in-between where liquid water can exist(NASA Science, 2023). However, habitability is complex and also depends on factors such as the planet’s size, mass, atmosphere, magnetic field, and the type of star it orbits (Kane, 2016).

Kepler-186f represents a major milestone because it was the first Earth-sized planet found in a star’s habitable zone. This generated significant levels of excitement in the science community by showing that life may exist somewhere else in the universe. Kepler-186f wasn’t just another distant rock in space, it was cosmic news. It became the first Earth-sized planet to ever be found orbiting in the "just right" zone around its star, where temperatures might allow for oceans to exist (Quintana, 2014). Suddenly, the age-old question “Are we alone?” didn’t seem so far-fetched anymore. Scientists and stargazers lit up with excitement. It was as if the universe had called us back with a wink.

2.3 MACHINE LEARNING IN EXOPLANET RESEARCH

When Kepler started sending back data, it wasn’t just a few planets, it was thousands of candidates, each with dozens of measured values like planet size, star temperature, and orbit time. Sorting through all of it manually is nearly impossible. That’s where machine learning (ML) enters the picture.

ML helps astronomers make sense of huge datasets, just like the one used in this project, which includes over 9,000 Kepler planet candidates. Each row contains features such as **orbital period, planet radius, equilibrium temperature, and stellar brightness**. The dataset looks like there is enough data to teach a computer to spot patterns and make predictions.

In this project, I wish to use ML to classify whether a planet is likely to be **confirmed** or not and whether it could fall into a **potentially habitable zone**. First, we explore the data using **EDA**, then **train models** to learn from it.

This isn't new. Researchers like Shallue and Vanderburg (2018) used deep learning to detect an eighth planet in the Kepler-90 system. Others, like Kane et al. (2016), cataloged which Kepler planets fall into habitable zones using parameters like solar insolation and temperature. In short, ML seems to help turn data into discovery. In a field where thousands of candidates need checking, it's not just a helpful tool, it's essential (Shallue, 2017).

3. Problem statement

As the Kepler Space Telescope brought back data from its long watch over the stars, astronomers were flooded with an overwhelming stream of data information. Thousands of planet-like signals, some real, some false. Behind every faint dimming of starlight could be a distant world or just cosmic noise. The challenge, I believe however, is not just discovering exoplanets, but maybe also distinguishing them from false positives, and even further, asking the question of which of these could potentially be habitable, like our own planet.

This task is made more difficult by the nature of the data itself, which is indirect, complicated, and filled with uncertainty. Many indications that first appeared as planetary are eventually considered stellar noise or an instrument issue(Institute, 2024). Even for confirmed exoplanets, determining whether they lie in a potentially habitable zone is far from straightforward (Kane S. H.-G., 2016). This complexity is reflected in the Kepler KOI dataset used in this project, which contains over 9,500 detections documented with dozens of planetary and stellar attributes across the KOI dataset. Many of these features are incomplete, and some measurements, such as temperature error margins, are missing entirely. The dataset also includes a third label "CANDIDATE," that represents unconfirmed cases that may eventually be reclassified, making the data harder to interpret with confidence. With so many features, missing values, and ambiguous labels, the real question becomes: how can we tell which ones are relevant?

This brings us to the question driving this project:

What features distinguish confirmed exoplanets from false positives in the Kepler dataset, and how can these features be used to predict potential habitability?

4. Project objectives/hypotheses

To help me answer the main question, this project sets out to explore the Kepler Object of Interests (KOI) dataset using data analysis and machine learning. I want first to build a predictive model that can classify whether a KOI is likely to be a confirmed planet or a false positive. If the classification model is accurate, I want to go further and investigate which of the confirmed exoplanets might potentially be habitable.

But before we can do any modeling, the dataset must undergo proper preprocessing. This could include checking for and handling missing values, cleaning the columns and their formats, and preparing the data for analysis. Once the dataset has been cleaned and stored in a structured MySQL schema, each column (feature) will be individually explored to better understand its meaning, data type, and distribution.

After exploring columns individually, we will continue with our EDA more in-depth. We will combine columns that may be most related to finding real planets vs. false planets. This will help uncover relationships between variables, assess the balance of class labels (e.g. confirmed vs. false positive), and identify any patterns or anomalies. This process can maybe help ensure a solid foundation for the machine learning phase, allowing the models to learn from meaningful data.

The objective of this project is therefore:

To develop predictive models to classify Kepler Objects of Interest as either confirmed exoplanets or false positives and to predict the potential habitability of confirmed exoplanets.

To guide this investigation, the following secondary research questions will be addressed:

1. **Which physical and orbital characteristics most strongly influence whether a KOI is classified as a confirmed exoplanet or a false positive?**

2. **How accurately can machine learning models predict whether a KOI is a confirmed exoplanet based on its observed properties?**

3. **What set of criteria based on planetary and stellar properties can be used to define potential habitability among confirmed exoplanets?**

4. **How effectively can machine learning models identify potentially habitable exoplanets among the confirmed exoplanet population using Kepler data?**

As the project explores these goals and questions, it doesn't just aim to use ML to sort planets, it also hopes to better understand the conditions in space that might allow habitable planets. By carefully cleaning the data, looking at each column, and training the models, this study uses the Kepler dataset as a way to search for other worlds that might be a little like Earth.

5. Data warehouse details

The dataset used in this project comes from NASA's Exoplanet Archive, specifically the Kepler Objects of Interest (KOI) cumulative table. It was first obtained in CSV format in Kaggle but noticed a lot of errors in the dataset. I then downloaded the dataset directly from NASA's Exoplanet Archive (as CSV) and got a slightly better result. Each row represents a potential exoplanet, with columns detailing planetary, orbital, and stellar properties. The data contains 50 columns and 9564 unique rows.

After downloading the dataset, it was cleaned and pre-processed by handling missing values, standardizing formats, and removing irrelevant features. The cleaned data was then imported into a MySQL database, where it was stored in a structured schema for easier access and analysis. This process is described in more detail in content 7: Data Pre-processing. Since all data originated from a single source, no merging of datasets was required.

6. Interesting insights

Before we dive into the dataset of the Kelper data, it's important to take a closer look at the columns and what they represent. This step is what we call Exploratory Data Analysis (EDA). Basically, EDA can help us understand what data we are working with and if there is any pattern or weird stuff going on, and which variables might be useful for predicting things like confirmed exoplanets or maybe even potential habitability.

I must address that I have already done a good amount of cleaning the data. Something that I will explain more in the data pre-processing section. To explore the dataset, I have used tools such as Excel (Power Query), PyCharm, and Jupyter Notebook for more advanced charts and deeper analysis.

We will address each column one by one so that we can figure out which ones are important for the goal of this project and which ones may not bring that much value. The key is to highlight the ones that potentially are the key to answering our questions. The information I got from each column mainly comes from:

https://exoplanetarchive.ipac.caltech.edu/docs/API_kepcandidate_columns.html

https://sites.astro.caltech.edu/~howard/cks/column-definitions.txt

https://exoplanetarchive.ipac.caltech.edu/docs/PurposeOfKOITable.html

https://archive.stsci.edu/kepler/koi/help/columns.html

https://exoplanets.nasa.gov/exoplanet-watch/about-exoplanet-watch/glossary/

https://exoplanetarchive.ipac.caltech.edu/docs/API_PS_columns.html?utm_source

The dataset from NASA Exoplanet Archive – NASA Exoplanet Science Institute: https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=cumulative, has 50 columns with 9564 rows. For this EDA, the data has been pre-processed and has now 47 columns and still got 9564 rows.

6.1 IDENTIFICATION AND STATUS

Column: kepid: this is the Kepler ID for the host star of the object we're observing. It's more like a tag to help us track each system.

Column: kepoi_name: This is the Kepler Object of Interest (KOI) name. So, if Kepler saw something interesting, like a possible planet transit, it gets this name. It's a temporary label until it's confirmed.

Column: kepler_name: If a planet *has* been confirmed officially, then it gets a real name here. If this column is blank, then the object probably hasn't been confirmed yet.

These columns will help us keep things organized and figure out *what* we're even looking at. ID-type columns are useful for filtering and referencing, but we won't be using them in modeling since they don't explain any behavior or physical properties.

6.2DISPOSITION (CLASSIFICATION OF PLANET CANDIDATES)

Column: koi_disposition:

This is a pretty important column in the dataset. koi_disposition basically tells us the current status of the object Kepler spotted. It tells us what scientists think it is after checking it out more closely.

There are three values here:

- **CONFIRMED**: This one's a real exoplanet. It's been double-checked and passes the tests.
- **FALSE POSITIVE** - This one looked like a planet at first but turned out not to be. Could be a binary star or some other weird space thing that tricked the system.
- **CANDIDATE** - Still under review. Could be a planet, might not be. Needs more data or checking.

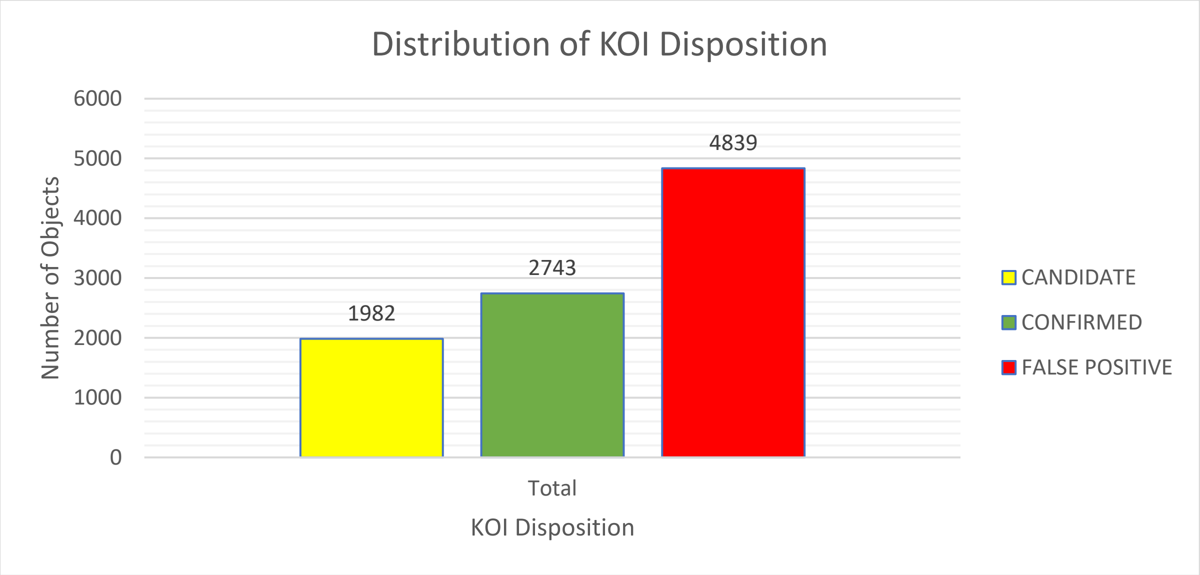


Figure 4: Column koi_disposition

- False positive (red) is the dominant value in the column with 4839 out of 9564 in total.

- Confirmed planets (green) have a total of 2743.

- Candidate has a total of 1982.

This column is super useful for our project because it's basically our label when we try to train a machine-learning model for prediction. Koi_disposition could be the explained variable/dependent variable. So, we will be using this later in classification models and visualization to see patterns between confirmed planets and the rest.

Column: koi_pdisposition:

This column is kind of a "first impression" version of the previous column. koi_pdisposition stands for planetary disposition, and it's what the automated Kepler pipeline initially thought about the object **before** any deeper review by humans.

You'll mostly see two values here:

- **CANDIDATE –** The pipeline flagged this as a potential planet based on the light curve.

- **FALSE POSITIVE –** Something looked off right from the start, so it's probably not a planet.

It's a bit like the machine's first guess, while koi_disposition is more of the final verdict after scientists take a closer look.

This column can be useful for seeing how good the initial pipeline was at spotting planets, or even for checking how much it agrees with the final decision. But we will probably focus more on koi_disposition when doing actual prediction tasks since that's the confirmed information.

Column: koi_score:

This one's actually pretty interesting. koi_score gives us a **confidence score** from 0 to 1 for how likely it is that the object is a real planet. So, the closer the number is to 1, the more confident the system is that this KOI (Kepler Object of Interest) is legit.

- A score near **1.0** means "we're pretty sure this is a planet."
- A score near **0** means "probably not a planet."

I chose to visualize the column koi_score as a histogram with a kernel density estimate (KDE) together with mean and median lines. The purpose of this is to best show how frequently the different confidence levels of the koi_score. This can help detect skewness or the concentration of the values. The KDE is supposed to highlight the overall shape and peaks of the distribution without relying on the bin size.

The mean and median lines may help us summarize the central tendency of the data.

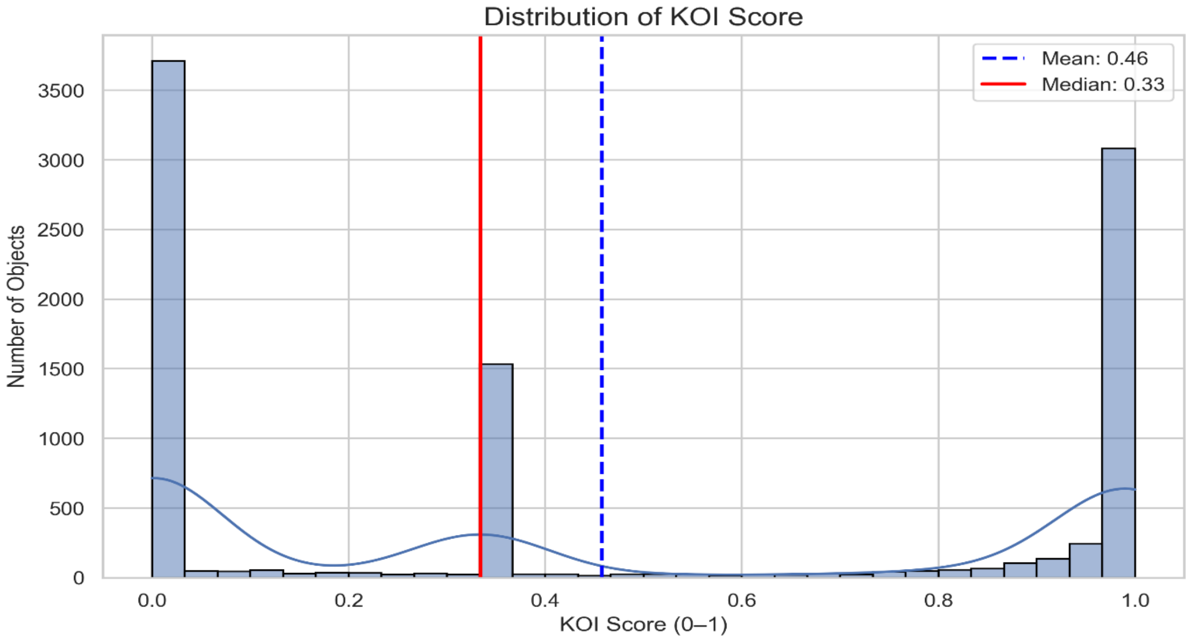
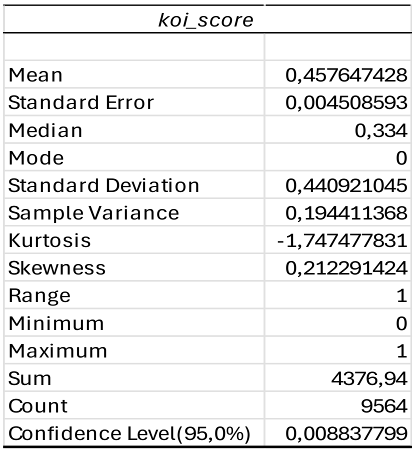


Figure 5: Column koi_score

Chart shows:

- The light blue bars show the actual counts in bins.
- The mean value shows the average score across all koi scores and gives us a general idea of the “center of mass”.
- The median of 0.33 is the midpoint of the score meaning that 50 % have a score below and 50 % above this value. If the median is less than the mean, then the distribution is right-skewed.



koi_score	
Mean	0,457647428
Standard Error	0,004508593
Median	0,334
Mode	0
Standard Deviation	0,440921045
Sample Variance	0,194411368
Kurtosis	-1,747477831
Skewness	0,212291424
Range	1
Minimum	0
Maximum	1
Sum	4376,94
Count	9564
Confidence Level(95,0%)	0,008837799

Figure 6: Descriptive Statistics of koi_score

Descriptive statistics (figure 5):

- Mode of 0 dominates with 3682 (and 1.0 with 2866).
- The standard error of 0.0045 shows how far the sample mean is from the true mean of 0.46.
- Standard deviation of 0.44 shows how spread out the scores are and is likely moderate to high due to the spread between extreme scores such as 0 and 1.

This column can give us a nice extra angle to look at. For example, we can compare this score with the koi_disposition to see if high scores match up with confirmed planets.

This one is super helpful for both EDA and maybe even as a feature in machine learning models since it's a kind of prediction summary from earlier processing steps.

6.3FALSE POSITIVE FLAGS (REASONS OBJECT WAS MARKED FALSE POSITIVE)

Columns: koi_fpflag_nt, koi_fpflag_ss, koi_fpflag_co, koi_fpflag_ec

These four columns correspond to a set of false positive flags (Archive, 2023). Each of these columns is binary (0 or 1) and shows us why the system thinks a KOI may not be a real planet. If the value is 1, that flag has been triggered, like a red light saying, "This could be an issue." Below is what each one means:

- **koi_fpflag_nt** (Not Transit-like) – If this value is a 1, then the light curve doesn't really resemble a planet transit. Maybe it is too noisy or just not the right shape. If so, then it's probably not a planet.

- **koi_fpflag_ss** (Stellar Eclipse) – This one is a signal that the system thinks the signal may resemble a star-to-star eclipse, not from a planet.

- **koi_fpflag_co** (Centroid Offset) – The dip in brightness from the light curve is not centered on the target star; it may be coming from a different stellar source from someone nearby. Therefore, this flag checks to see if the signal is from off-target.
- **koi_fpflag_ec** (Ephemeris Match Indicates Contamination) – This one is a bit more advanced. If the timing (ephemeris) matches that of known false positives or noisy signals elsewhere, this one will be flagged. Basically saying "Hey, this could be a contaminated signal or a copied signal."

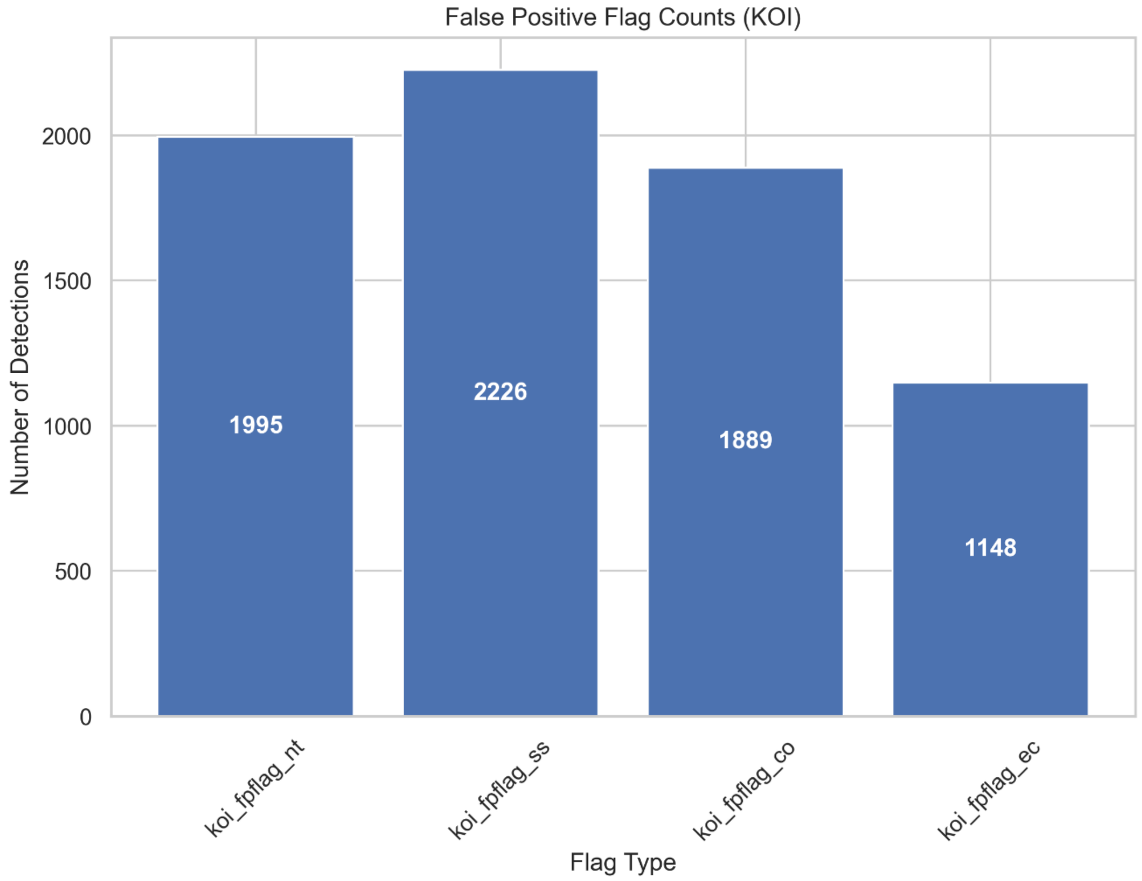


Figure 7: Colums of koi_fpflags

The most common flags in the chart:

- koi_fpflag_ss (Stellar Eclipse) has over 2200 that were marked because of a secondary eclipse. Following the description of columns by NASA's archives, this implies a binary star and not a planet.

- Column koi_fpflag_nt (Not Transit-Like) got nearly 2000 flags. This shows how often nearby transit confuses the detection process.
- Koi_fpflag_co (Centroid Offset) of 1889 errors hinting that the transit source was not perfectly aligned with the target star.
- The least common flag was the Ephemeris Match Indicates Contamination flag with 1148 cases.

These columns may be useful for filtering out suspicious candidates and understanding why some objects got flagged as false positives. We might not use them directly in prediction, but they give a lot of insight into this EDA process.

6.4 ORBITAL PARAMETERS (CHARACTERISTICS OF THE EXOPLANET'S ORBIT)

Column: koi_period:

This one is a key column when we talk about how these planets (or candidates) move. koi_period tells us the **orbital period** of the object, that is how long it takes to make one full trip around its star. The value is in **days**.

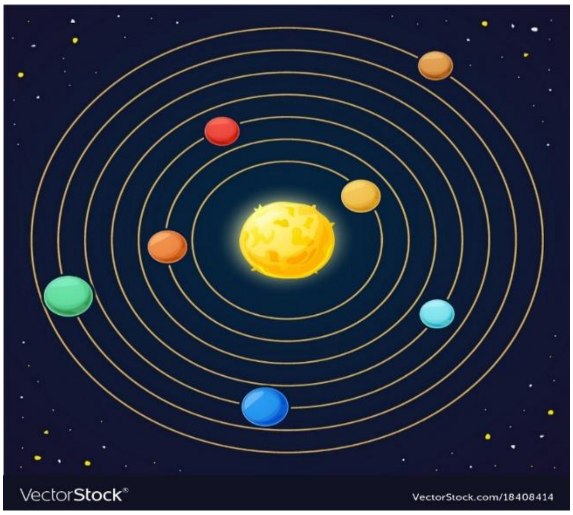


Figure 8: https://www.vectorstock.com/royalty-free-vector/exoplanets-orbiting-stars-vector-18408414

For example, Earth takes about **365 days** to orbit the Sun once, that's our year. So, if you see a koi_period of **3.5**, that means the planet whips around its star in just 3.5 Earth days. That's insanely fast, probably a really close orbit, and the planet is likely way too hot to be habitable. On the other side, a period of **300+ days** means it's cruising at a slower pace and farther out, maybe in a zone more similar to Earth's.

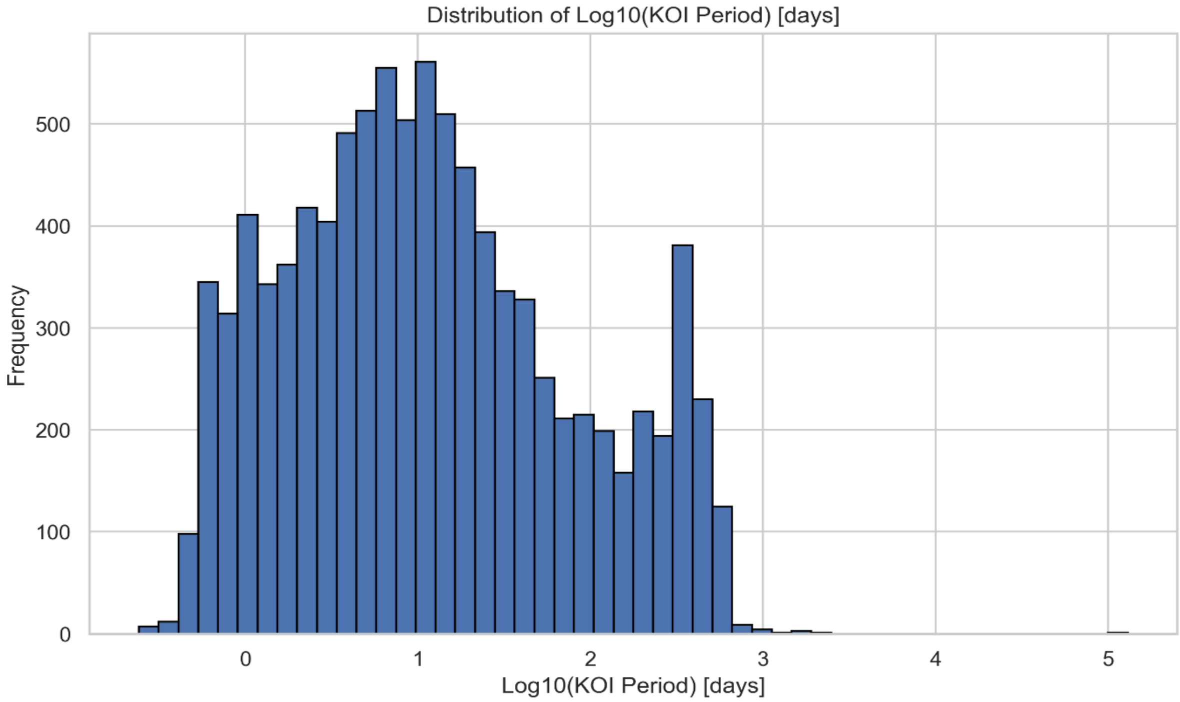


Figure 9: Column koi_period

This histogram shows the distribution with a log-transformed orbital period. The reason for the log transformation is that the original values span from less than a day to thousands of days. Without log 10 the chart is useless. Log 10 compresses the scale and turns multiplicative gaps into additive steps. This makes it easier to visualize the full spread of values.

What does the x-axis log10 show:

- A value of 0 corresponds to 10^0 = 1 day
- A value of 1 = 10^1 = 10 days
- A value of 2 = 10^2 = 100 days
- A value of 3 = 10^3 = 1000 days, and so on.

The Y-axis shows us the number of KOIs that fall into each bin.

Most of the KOIs have short periods. The tallest bars are clustered between log10 values of around 0 to 2, suggesting that many candidate exoplanets orbit their stars with periods between 1 and 100 days. Peak is likely around 10-20 days.

We can also show the column in groups to maybe simplify the ranges:

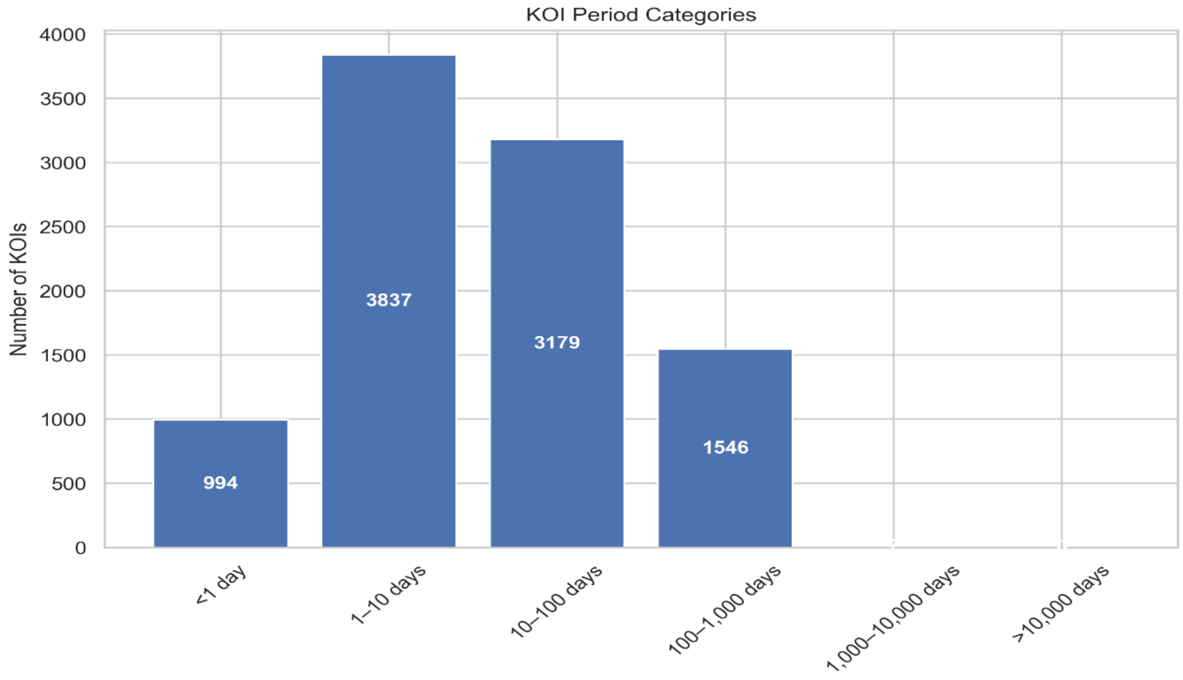


Figure 10: Column koi_period Grouping

Here we can see more clearly the different ranges. <1 day represents very fast orbits. 100-1000 days are Earth-like planets and planets with longer orbits. 1000 - 10,000 days are planets with very long periods.

This column could be super useful for understanding the planet's distance from the star, which ties into things like temperature and possible habitability. It could also be great for visualizing trends and comparing confirmed vs. false positives. Probably also during the modeling phase.

Columns: koi_period_err1 and koi_period_err2

These two columns go hand-in-hand with koi_period. They tell us about the **uncertainty** in the orbital period measurement—basically how confident we are in the value.

- **koi_period_err1** – This is the **positive error margin**. It tells us how much the period could be *higher* than the value in koi_period.

- **koi_period_err2 –** This one's the **negative error margin**, showing how much the period could be *lower* than the value.

So, let's say a KOI has a period of 50 days, an err1 of 0.2 and an err2 of -0.3. That means the real orbital period might be anywhere between **49.7 and 50.2 days**.

These error margins are important in science because measurements are never perfect. I am unsure if we will use columns that include err1 and err2 in this project because these are not primary predictors. I believe errors are more about measurement precisions and not physical conditions. Maybe we will find out when further exploring. I will not go further explaining other err1 and err2 columns.

Column: koi_time0bk

This one might look a bit technical at first, but it's actually pretty important if you're trying to understand when the planet crosses in front of its star. Koi_time0bk stands for the **time of the first observed transit**, measured in **Barycentric Julian Date (BJD)**.

So basically, it tells us **when** the planet was first seen passing in front of its star, from the perspective of Kepler. It's like the "starting point" of the planet's orbit.

What is BJD?:

BJD is a very precise time standard used in astronomy. It corrects the regular Julian Date (JD) for the motion of Earth around the Sun by converting the time measurement to the **solar system's center of mass** called the **barycenter**. This way, observations from different places and times can be compared without being affected by Earth's own movement in space (Boyce-Astro, n.d.; Eastman, Siverd & Gaudi, 2010).

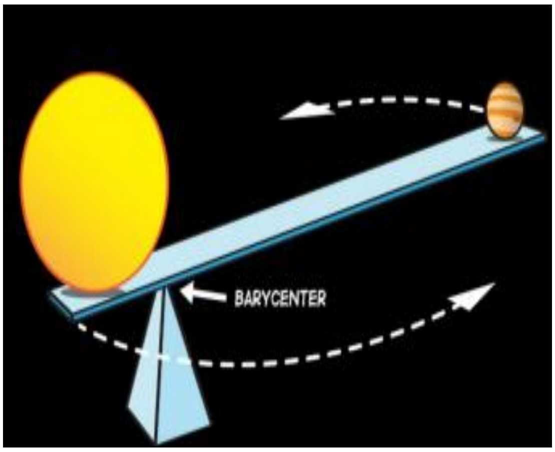


Figure 11: https://boyce-astro.org/wp-content/uploads/BRIEF-Video-Lesson-TIME-What-is-BJD.pdf

In other words, it gives a "universal clock" so astronomers can say *exactly* when something happened, like a planet transit, no matter where Earth was when the telescope observed it.

We probably won't use this column much in charts or ML models, but it's key for astronomical scheduling and understanding the timing of events.

Column: koi_impact

This column tells us about the **impact parameter** of the transit, which is kind of a fancy way of saying how central the planet's path is across the face of the star during transit.

- A value of **0** means the planet passed **right across the middle** of the star, like a perfect bullseye.
- A value near **1** means the planet just skimmed the edge of the star during transit.
- If the value is **greater than 1**, it technically isn't supposed to transit at all (unless some weird geometry going on).

Why does this matter? The impact parameter affects the depth and shape of the transit light curve. A central transit is more likely to exhibit a deeper and clearer dip in brightness, whereas a grazing transit is shallower and a less clear dip. Importantly, this may affect our confidence of detecting the planet and measuring its size (AstroPhil, 2022).

Check out AstroPhil's YouTube channel for more information:

https://www.youtube.com/watch?v=JMjEFgnifXE

This column can also be helpful in spotting false positives or tricky transit signals and could be used as a feature in machine learning models if we try predicting confirmation status.

6.5 TRANSIT & DEPTH INFORMATION (HOW PLANET'S TRANSIT IS OBSERVED)

Column: koi_duration

The koi_duration column tells us **how long** the planet takes to pass in front of its star, basically the **duration of the transit**. The transit duration value is given in **hours**. Exoplanetarchive writes in the column description: "*Contact times are typically computed from a best-fit model produced by a model fit to a multi-quarter Kepler light curve, assuming a linear orbital ephemeris*" *Mandel-Agol (2002)*. What they are telling us here, is that they calculate when a planet starts and ends its transit using the model Mandel-Agol. This is a model for simulating how light behaves when a planet transits a star. It takes into account the size of the planet and the star, the edges of the star that appear dimmer, and the exact path the planet takes across the star (Mandel, 2002). The Keplar collected data over a multi-quarter light curve meaning that they are using data over a long period time span (3-month period). The linear orbital ephemeris means that they assume the planet keeps orbiting at a constant period with no major changes.

So, if a KOI has a duration of **5 hours**, it means it blocks some of the star's light for about 5 hours during its orbit. Some transits are super quick (like under an hour), and others can last many hours depending on things like:

- How big the star is.
- How fast the planet is moving.
- The **impact parameter** (remember, a more skewed or edge-on path might result in a shorter visible transit).

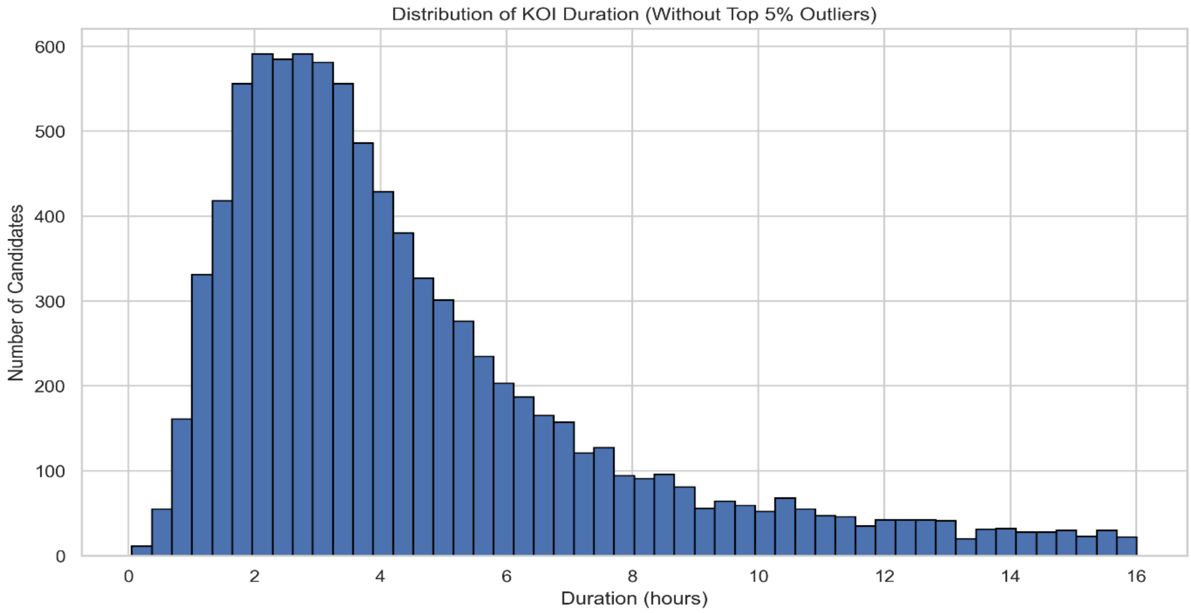


Figure 12: Column koi_duration histogram

Filtering out the top 5% of koi_duration values was the best way to show the values and to reduce the impact of outliers. Without doing this, it was hard to interpret the chart.

By removing the top 5 % of the longest durations we can see that the peak is around 3 to 4 hours. The distribution tapers off towards the right.

We can see the unfiltered data below:

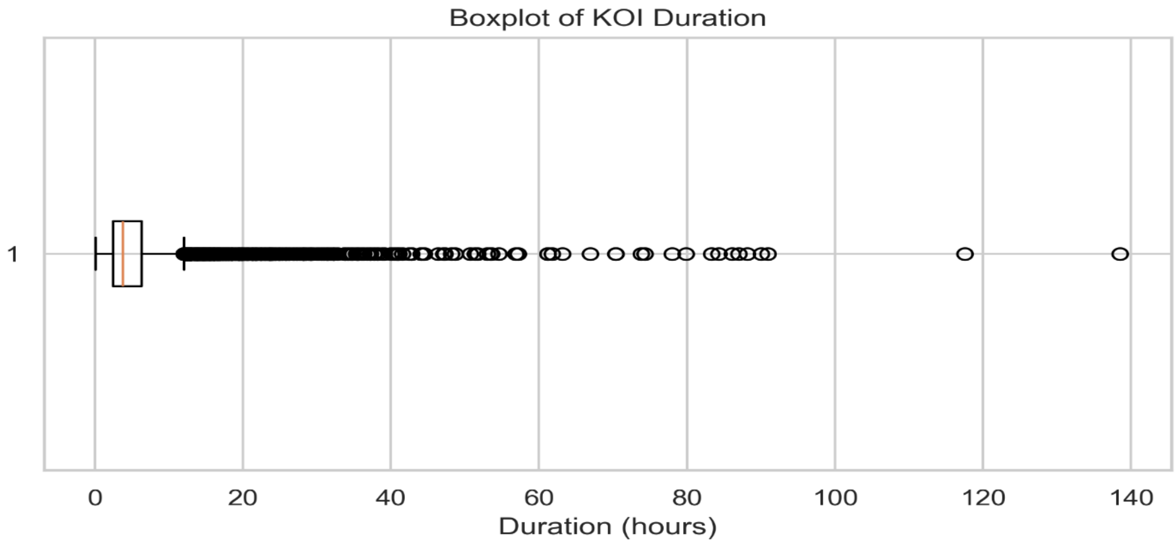
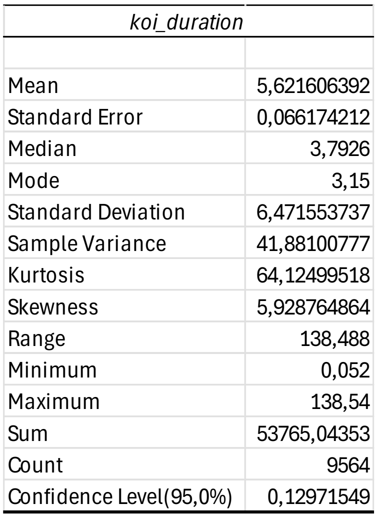


Figure 13: Column koi_duration boxplot unfiltered

Unfiltered boxplot:

- The box inside the chart shows the interquartile range (IQR) and this is where the middle 50 % of the data lies.
- The median line (orange line) is to the left inside the box meaning that most durations are on the shorter side.
- The long line of small black circles are outliers stretching far to the right. It reaches beyond 100 hours. These are extreme values.

The Descriptive Statistics on koi_duration:



koi_duration	
Mean	5,621606392
Standard Error	0,066174212
Median	3,7926
Mode	3,15
Standard Deviation	6,471553737
Sample Variance	41,88100777
Kurtosis	64,12499518
Skewness	5,928764864
Range	138,488
Minimum	0,052
Maximum	138,54
Sum	53765,04353
Count	9564
Confidence Level(95,0%)	0,12971549

Figure 14: Descriptive Statistics on koi_duration

- The mean value of 5.62 hours is the average duration of all KOI transits.
- The low value of 0.066 in the standard error suggests that the mean is reliable.
- Median of 3.79 hours is the middle value.
- Mode of 3.15 hours is the most frequent duration in the data.
- A Standard deviation of 6.47 hours shows how spread out the durations are. A high value confirms a wide range, but this is probably because of the outliers.
- We can also see the minimum (0.052 hours) and the maximum (138.54 hours). The maximum value could be a large object or maybe an incorrect detection.

This column could be useful in detecting confirmed planets vs. false positives because false positives may have longer or irregular durations. We'll probably use this together with other columns in the EDA to see how transit duration compares across candidates. Maybe even as a feature in a machine-learning model. Let's see as we go further.

Column: koi_depth:

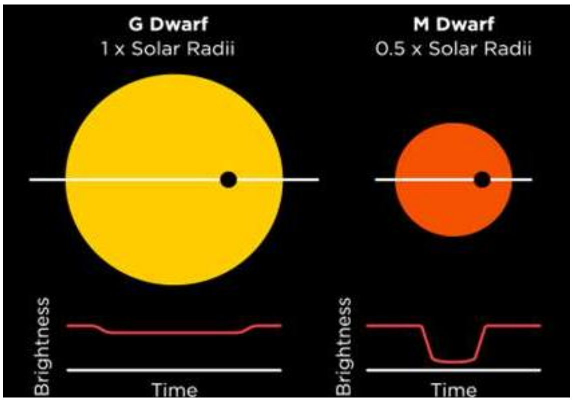


Figure 15: https://astro.unl.edu/newRTs/Transits/background/Transit1.html

The koi_depth column tells us how much the brightness of the star drops when the planet passes in front of it. This drop in brightness is measured in parts per million (PPM). The drop is directly related to the size of the planet relative to the star (University of Nebraska-Lincoln, 2017). The transit depth (D) is equal to the ratio (R_p) of the planet's radius and the star's radius (R_s) squared:

$$D = \left(\frac{R_p}{R_s}\right)^2$$

Let's make a simplistic calculation without considering any other factors, just if a planet has a radius that is 10 % of its star's radius:

$$D = (0.1 / 1)^2 = 0.01 = 1\,\% = 10{,}000\,ppm$$

So, for example: A depth of 1000 ppm means that the star got 0.1 % dimmer during the transit. Or a depth of 5000 ppm would be a 5 % dip, which is huge and could probably be caused by something big, like a gas giant or maybe a binary star.

Let's calculate: 1000 ppm means 1000 parts out of 1,000,00. $\frac{1000}{1{,}000{,}000} = 0.001 = 0.1\,\%$

The depth depends mostly on the size of the planet compared to the star. A bigger planet blocks more light, so the depth is deeper. Smaller planets, like Earth-size, only cause a tiny dip, making them harder to detect. (Wikipedia, 2024).

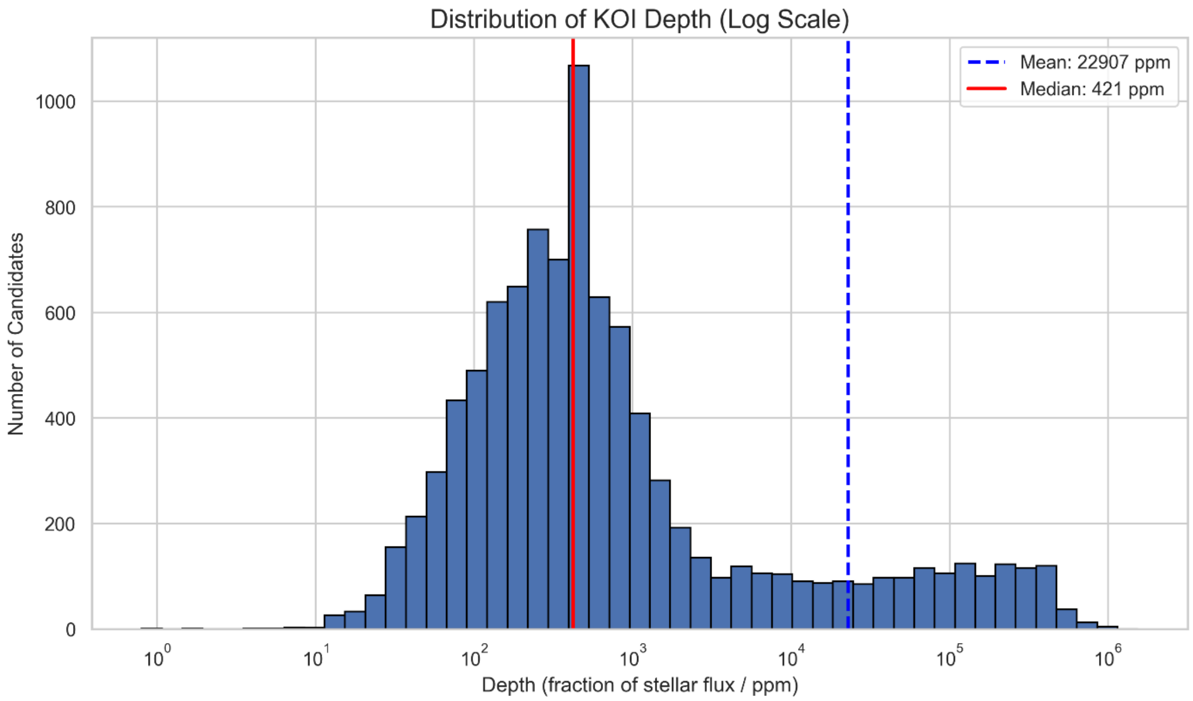


Figure 16: Column koi_depth histogr log10

The log10 histogram uses all valid data, including extreme values.

- The mean depth (blue dashed line) of 22907 ppm is probably pulled to the right because of some extreme values (outliers). We will show this in the boxplot under.
- The median is at 421 ppm which represents the center of the distribution for most of the values.

The log scale is like in the column chart koi_period. It increases like this: 1, 10, 100, 1000, 10,000, 100,000 etc. Most KOIs cluster below 10,000 ppm (10^4). In my opinion, the values that are above are probably large planets or false positives.

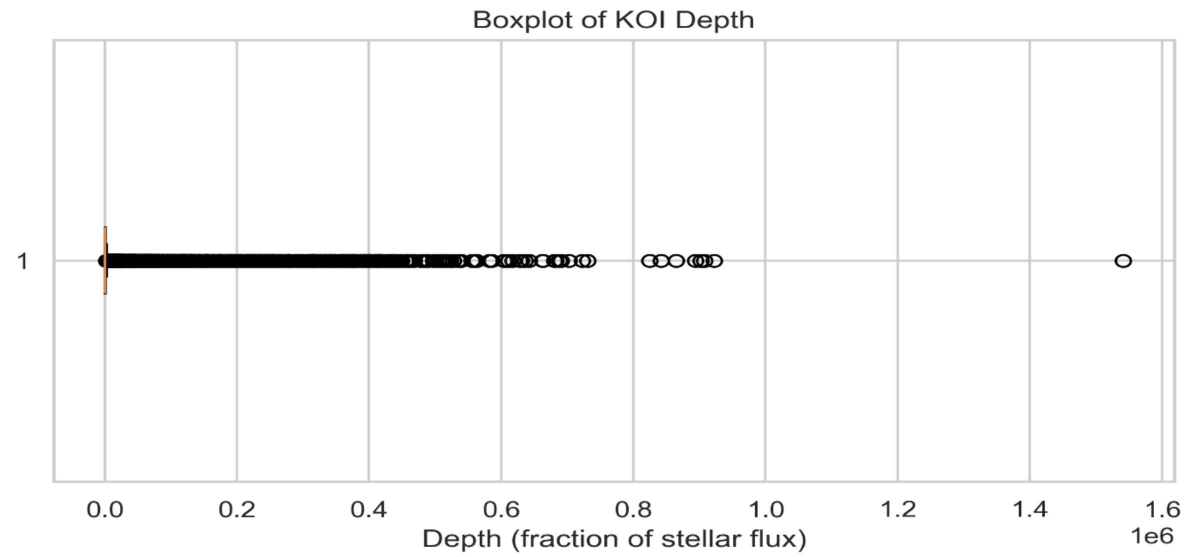
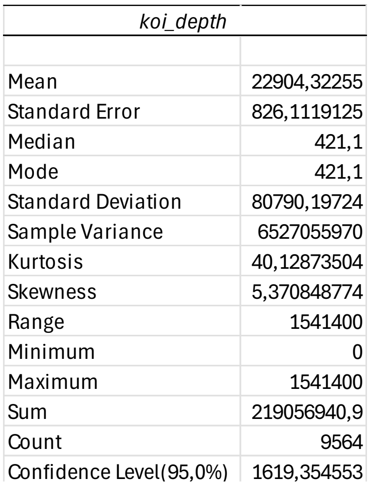


Figure 17: Column koi_depth boxpl.

The boxplot (fig. 16) is just to show the full scale of the column and to see the outliers:

- The x-axis is measured in ppm, and the scale here goes from 0 to about 1.6 million.
- As we can see, the orange rectangle box (which looks more like an orange line), contains the middle 50 % of the data, from the 25th to 75th percentile (IQR). Most values are very small, and they are crammed against the left side near zero. One extreme outlier is between 1.4 to 1.6 million ppm (probably a false positive?).

Descriptive Statistics (fig. 17):



koi_depth	
Mean	22904,32255
Standard Error	826,1119125
Median	421,1
Mode	421,1
Standard Deviation	80790,19724
Sample Variance	6527055970
Kurtosis	40,12873504
Skewness	5,370848774
Range	1541400
Minimum	0
Maximum	1541400
Sum	219056940,9
Count	9564
Confidence Level(95,0%)	1619,354553

Figure 18: Descriptive Statistics of koi_depth

- The mode of 421 ppm has the same value as the median. Many objects are likely clustered here.
- A standard deviation of 80790 is a huge value that is widespread and is driven by outliers.
- The same goes for Kurtosis with a value of 40.13. This is very high indicating extreme outliers, or even sharp peaks with flat tails. Normal kurtosis is 3 (ScienceDirect, n.d.).

- The most extreme value is the maximum with 1,541,400 ppm. As said before, probably a false positive.

This column could be important because it's how Kepler finds planets by measuring the brightness of stars over time and looking for these small dips.

6.6 PHYSICAL PROPERTIES OF THE EXOPLANET

Column: Koi_prad:

This column tries to understand the size of the exoplanet found by the Kepler telescope. What it tells us is the planet's radius that is measured in Earth's radii. Earth's radius is 6378 km (Wikipedia, 2024), but in this column, a value of 1.0 means the same as the Earth's radius. Basically, it shows values on how big the planet is compared to Earth. So, if a planet has a value of 0.5, then it is about half the size of Earth. If the value is 10, then it is a massive planet maybe the size of a Jupiter planet or even larger.

This column also uses the same modeling of Mandel & Agol from modeling the transit light curve. When a planet passes in front of a star, it blocks some of the light. The deeper the dip, the bigger the planet, assuming the star size is known (Mandel, 2002). So, this value is estimated based on how much light is blocked during the transit.

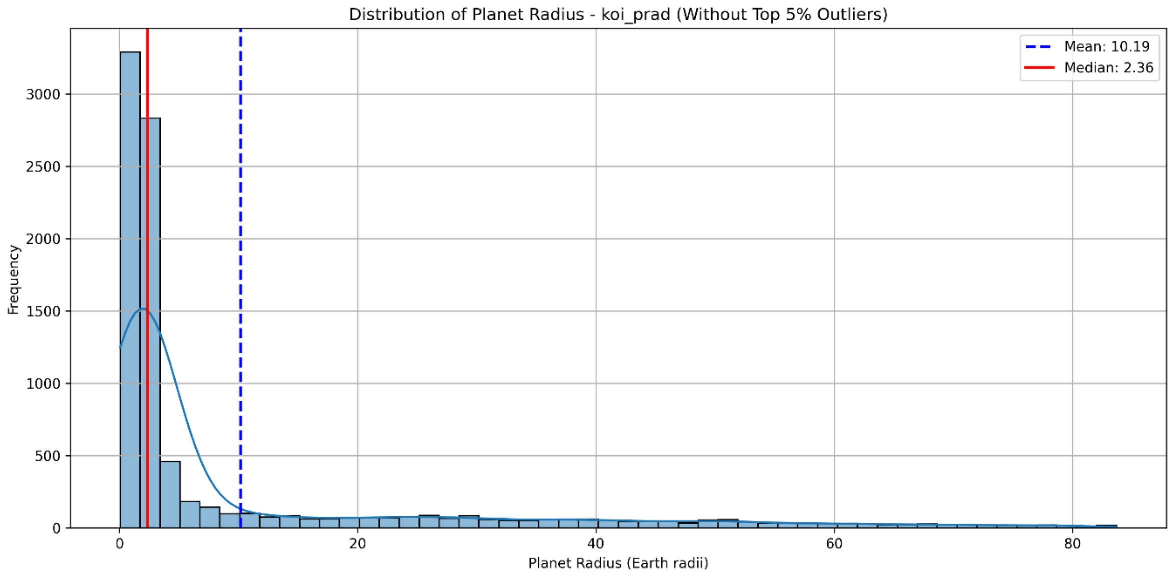


Figure 19: Column koi_prad histogr.

The chart shows:

- A filtered koi_prad (without top 5 % outliers) to help better visualize the typical planet size.
- That the mean 10.19 is being pulled to the right because of a few larger values.
- The median 2.36 lies to the left of the mean, meaning that it is positive skewed, and most of the exoplanets have a radii between ruffly 0.5 to 3 Earth radii.

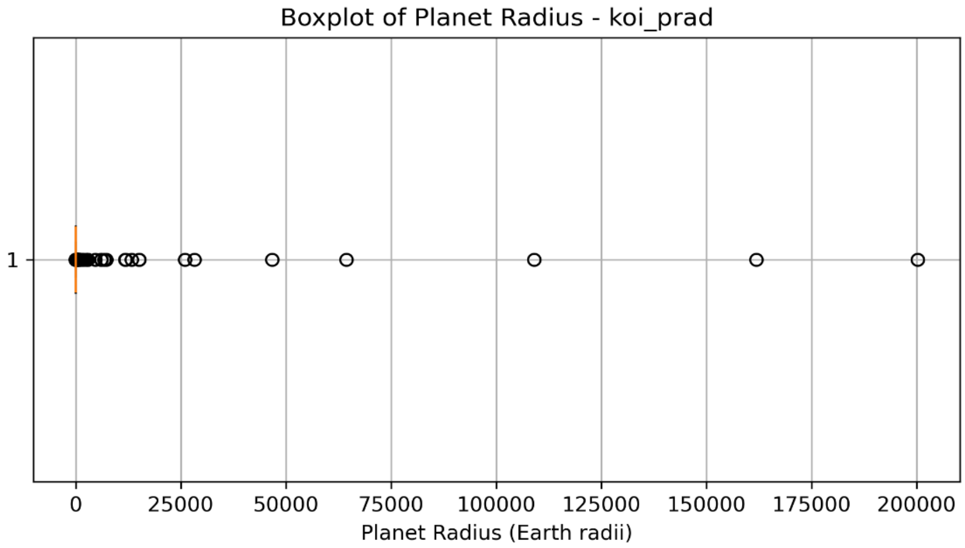


Figure 20: Column koi_prad boxplot unfiltered.

The boxplot shows:

- The unfiltered data values are in the column.
- The majority of planet radii are clustered near value 0. This could indicate that there are numerous Earth-sized or smaller planets.
- The long tail to the right indicates that there might be large planets (and could include gas planets).

The koi_prad could possibly be combined with column koi_period to distinguish the confirmed vs. false positives. It could also be a key feature when we talk about possible habitability (or just understanding the nature of the exoplanet). For example:

- Small rocky planets (like Earth or Mars) usually have koi_prad values below 2.
- Bigger gas giants (like Jupiter) go way higher, over 10 Earth radii.
- A lot of the confirmed planets fall somewhere in between, and those are often called "sub-Neptunes" or "mini-Neptunes" (Wikipedia, Earth radius, 2024).

Column: koi_teq:

The column koi_teq stands for the equilibrium temperature for the planet, meaning it is an estimate of how hot the planet would be if it did not have an atmosphere and was just soaking up energy from its star and re-radiating it back out (Wikipedia, Planetary equilibrium temperature, 2024). Think of it like a rock planet getting baked by starlight.

The value is given in Kelvin (K), the base temperature in the international system of units (Wikipedia, Kelvin, 2024). This temperature is calculated using the star's properties like size and brightness and the planet's distance from it (which we often estimate from the orbital period). So, the formula assumes that the planet is a perfect blackbody (meaning it absorbs all radiation), it has no atmosphere and no internal heating and it is in thermal equilibrium (meaning energy inn = energy out) (Wikipedia, Planetary equilibrium temperature, 2024).

To understand this, we have to reference some equilibrium temperatures in our solar system:

Earth is about 255 K (without our atmosphere's greenhouse effect). Mars is around 210 K, Mercury is around 431 K, and Saturn is around 63 K (Wikipedia, Planetary equilibrium temperature, 2024; Gary, 2010).

So, a koi_teq value of:

- 250-300 K is kind of "Earth-like" (with no greenhouse effect).
- >1000 K is probably a planet very close to the star, or even a "hot Jupiter" (Astrophysics, 2021).
- <150 K is probably a cold, icy world on the outer edge of the system.

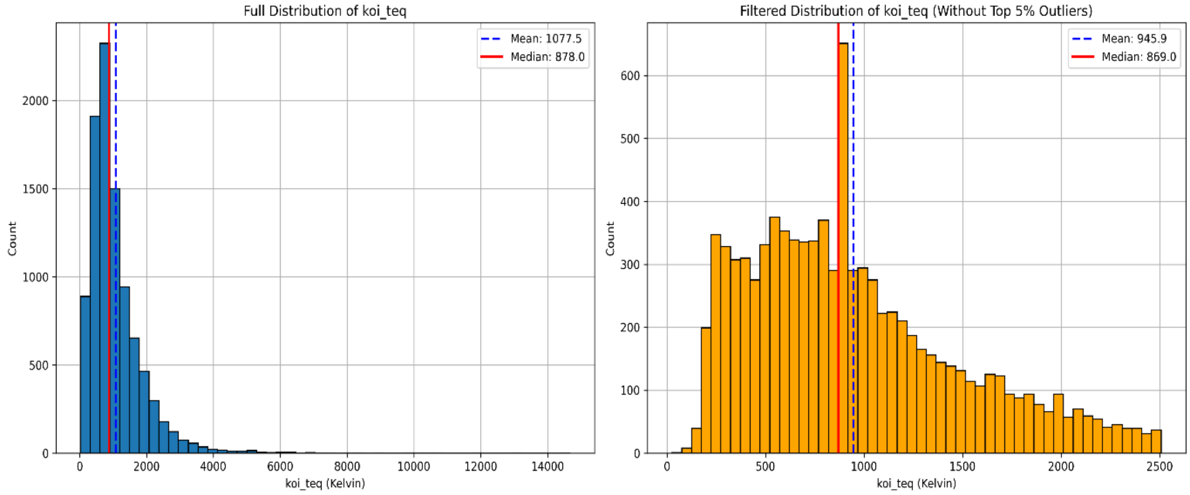


Figure 21: Column koi_teq. Full and filtered

What the full distribution chart (left) shows:

- The chart shows a strong positive skew. Most of the planets have equilibrium temperatures below 2000 K.
- The mean (blue dash) is at 1077.5 K and the median (red line) is at 878 K. As we can see, the mean is higher than the median. This could be caused by a long tail of high-temperature outliers.

This could tell us that most KOIs are cool, but a few are extremely hot planets that pull the average up. The values above 4000 are probably planets that are very close around hot stars or maybe just false positives.

What the filtered distribution chart (right) shows:

- This chart focuses on KOIs with temperatures = or < 2500 K. Taking off the 5 % extreme values.
- The mean is still above the median but much closer to it than before.
- There are large numbers of KOIs in the 200-1000 K range.

This column could potentially be a key factor when estimating potential habitability.

Column: koi_insol:

This column stands for stellar isolation flux, which is another way of saying "how much starlight (or sunlight) the planet gets from its star. It is like measuring the planet's solar exposure, how intensely it is being blasted with energy from its sun.

The values are in Earth units, meaning that:

- A value of 1.0 means the planet receives the same amount of energy as Earth does from the sun (Archive, 2023).
- A value of 0.5 means that it gets half as much, so that means it is cooler.
- A value of 2.0 means that it gets twice as much, so that means its likely hotter.

This column could be important for whether a planet could possibly support life. Scientists like Petigura, E.A., Howard A.W., and Marcy, G.W., use it when estimating whether a planet falls into the habitable zone (not too hot, not too cold region or so-called "Goldilocks Zone") where liquid water could exist. Earth-size planets within 0.25 to 4 (units) times Earth's stellar flux (Petigura, 2013).

6.7 DETECTION AND SIGNAL STRENGTH

Column: koi_model_snr:

Column koi_model_snr stands for Signal-to-Noise Ratio (SNR). This column means how strong the planet's transit signal was compared to the background noise in the data. For example: A high SNR signal means that the transit signal of the planet is really strong. A low SNR signal is uncertain if it was a planet or just some random wiggle in the data.

The calculation of this column is done in several steps like transit modeling, whitening the light curve, signal extraction, and noise estimation to name a few (Twicken, 2019). But we will not go into detail in this matter. We are more interested in the values that the columns represent.

The Kepler mission established a detection threshold of 7.1 SNR, meaning that any transit above 7.1 was considered significant. This was to minimize the false positives due to many numbers of statistical tests. (Jenkins, 2019).

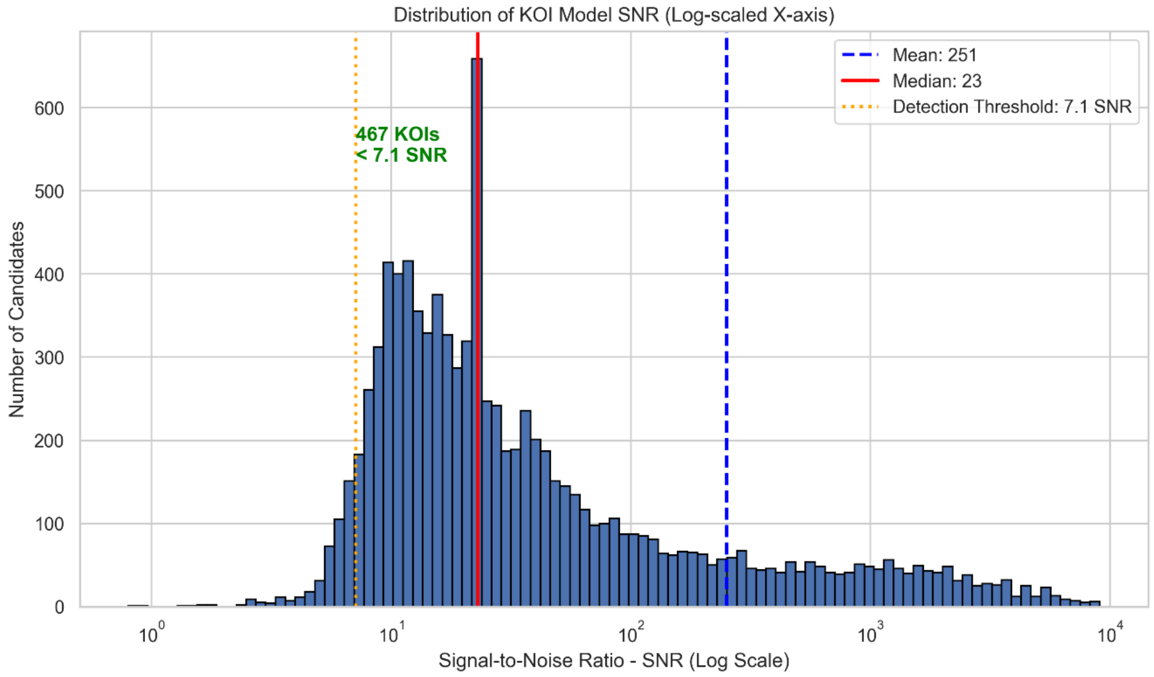


Figure 22: Column koi_model_snr histogr.

Chart shows:

- The chart axis is in logarithmic scale (as we have done before in some other columns).
- There are 467 KOIs under 7.1 snr. Low SNRs are often associated with false positives. A high koi_model_snr means the system is more confident there's a planet there.
- The median is at 23 snr, meaning that this is the typical snr of most KOI's (not affected by outliers).
- The mean is at 251, meaning that this high value is the impact of extreme high snr detections on the average.

This column could maybe be useful when distinguishing confirmed vs. false positives.

Column: koi_tce_plnt_num:

The column stands for TCE (threshold crossing event) planet number. It tells us which planet candidate in the system this particular entry is referring to. Some stars have multiple dips in light that look like potential planets, and each of the dips gets its own number. So, for example: Value nr 1 is the first candidate, nr 2 is the second candidate, etc. in a multi-planet system. (Archive, 2023). This column could be helpful if you want to keep track of which transit signal belongs to which planet candidate in the same system. This column could also be helpful to combinate with other columns in classifying confirmed vs. false positives because of multiple planets in the same system, the more planets in a system = the more likely there is a confirmed planet. That is my own opinion.

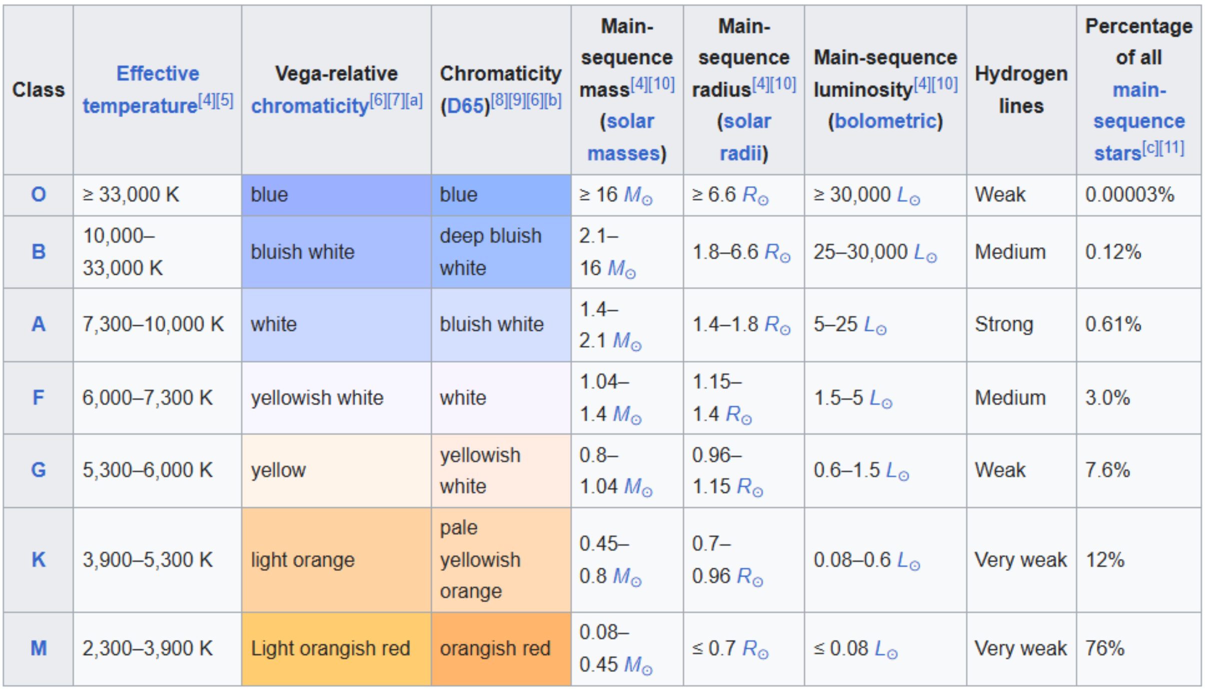
Column: koi_tce_delivname:

This column is of no interest to this project, and I will not refer to it. The column just shows the version of the pipeline data processing delivered.

6.8 PROPERTIES OF THE HOST STAR

Column: koi_steff:

This section of the column looks at the host star. This could be very important because of the star that the planets orbit around. This temperature of the host star is measured in Kelvin (K), like the column koi_teq (Wikipedia, Kelvin, 2024). It is all about the overall energy coming from the star's surface helping to define the star's type and spectral class. Here is a little summary from the stellar classification table in Wikipedia (Wikipedia, Stellar classification, 2024):



Class	Effective temperature[4][5]	Vega-relative chromaticity[6][7][a]	Chromaticity (D65)[8][9][6][b]	Main-sequence mass[4][10] (solar masses)	Main-sequence radius[4][10] (solar radii)	Main-sequence luminosity[4][10] (bolometric)	Hydrogen lines	Percentage of all main-sequence stars[c][11]
O	≥ 33,000 K	blue	blue	≥ 16 $M_\odot$	≥ 6.6 $R_\odot$	≥ 30,000 $L_\odot$	Weak	0.00003%
B	10,000–33,000 K	bluish white	deep bluish white	2.1–16 $M_\odot$	1.8–6.6 $R_\odot$	25–30,000 $L_\odot$	Medium	0.12%
A	7,300–10,000 K	white	bluish white	1.4–2.1 $M_\odot$	1.4–1.8 $R_\odot$	5–25 $L_\odot$	Strong	0.61%
F	6,000–7,300 K	yellowish white	white	1.04–1.4 $M_\odot$	1.15–1.4 $R_\odot$	1.5–5 $L_\odot$	Medium	3.0%
G	5,300–6,000 K	yellow	yellowish white	0.8–1.04 $M_\odot$	0.96–1.15 $R_\odot$	0.6–1.5 $L_\odot$	Weak	7.6%
K	3,900–5,300 K	light orange	pale yellowish orange	0.45–0.8 $M_\odot$	0.7–0.96 $R_\odot$	0.08–0.6 $L_\odot$	Very weak	12%
M	2,300–3,900 K	Light orangish red	orangish red	0.08–0.45 $M_\odot$	≤ 0.7 $R_\odot$	≤ 0.08 $L_\odot$	Very weak	76%

Figure 23: https://en.wikipedia.org/wiki/Stellar_classification

A breakdown of the table:

1. 2000 – 4000 K. These are cool stars.
2. 5000 – 6000 K. These are like our own sun's temperature of 5772 K (NASA, 2021).
3. 6000 – 7500 K. These are hotter stars.
4. 8000 K+. These stars are very hot.

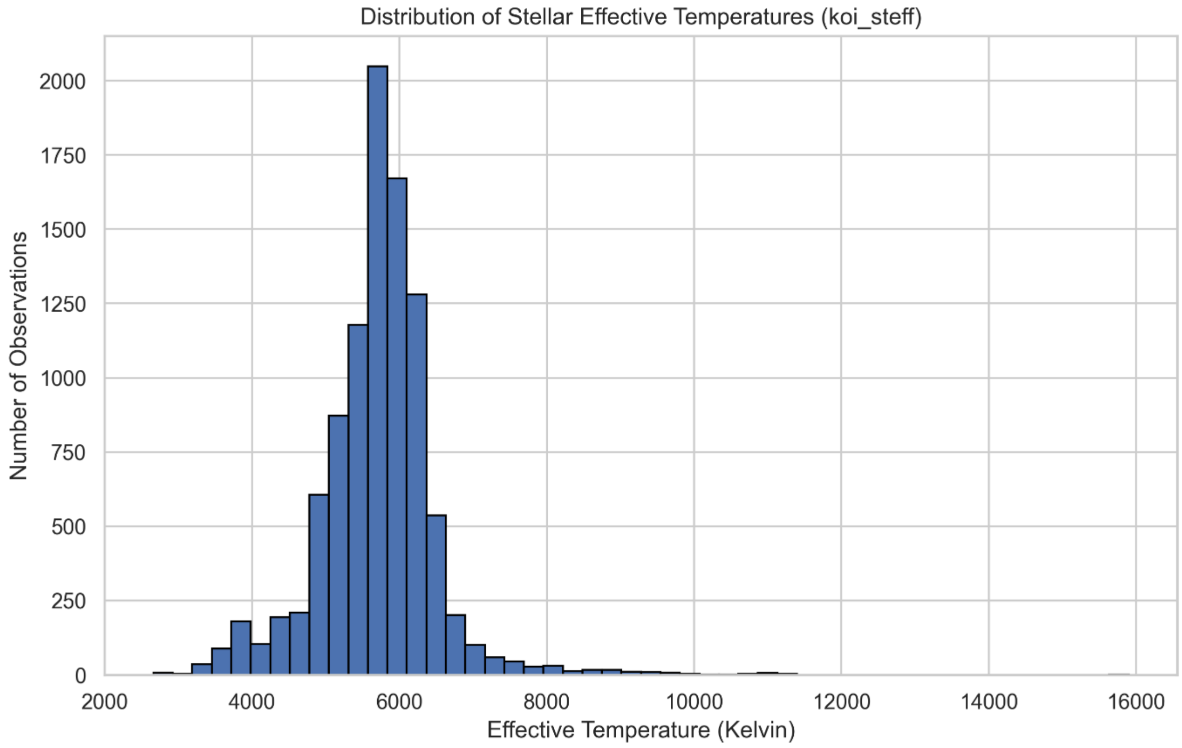
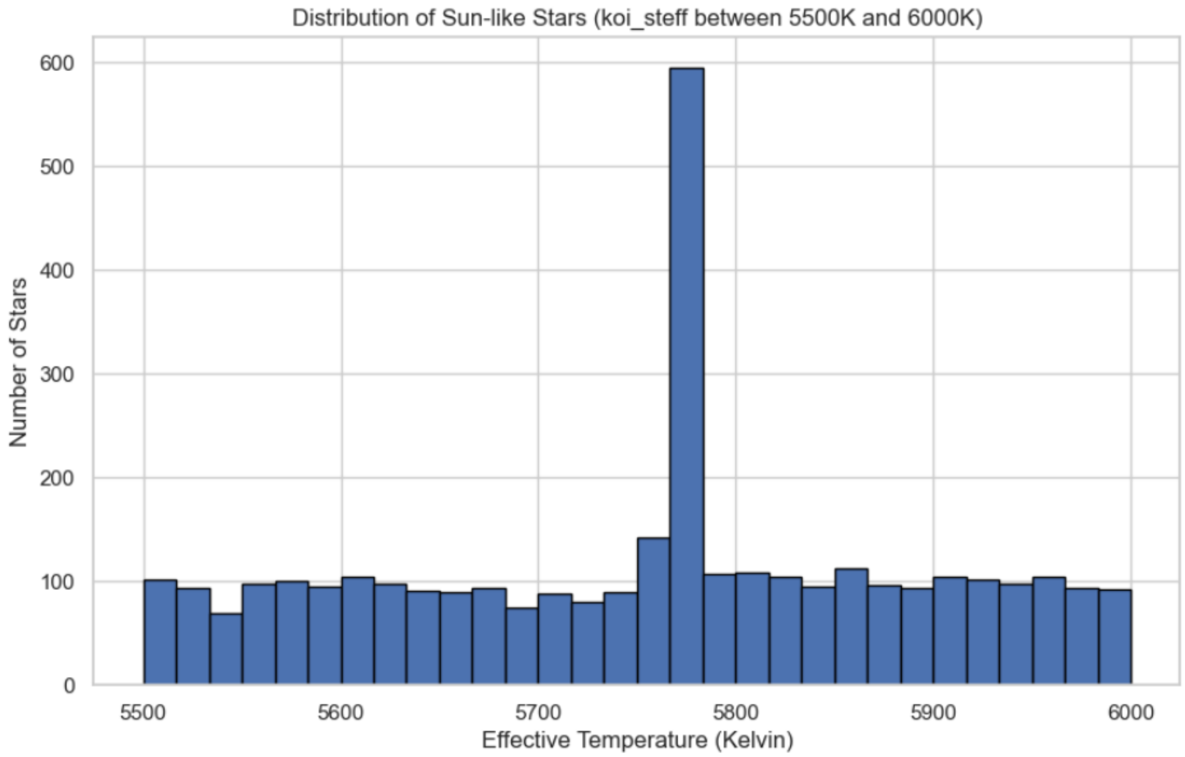


Figure 24: Column koi_staff histogr.

The chart shows:

- An x-axis with temperature in Kelvin and the y-axis shows how many stars fall into each temperature bin.
- The peak is somewhere around 5000 – 6000 K (similar to the sun’s temperature at 5772 K.
- A long tail stretches toward hotter stars up to 1000 + K.
- Fewer stars have very low temperatures under 3000 K. These could be faint stars that are harder for the Kepler Telescope to detect.

Let’s zoom in on the histograms to show stars with effective temperatures close to the Sun’s:



```
There are 3,411 sun-like stars in the chart (Effective Temperature: 5,500K to 6,000K)
```

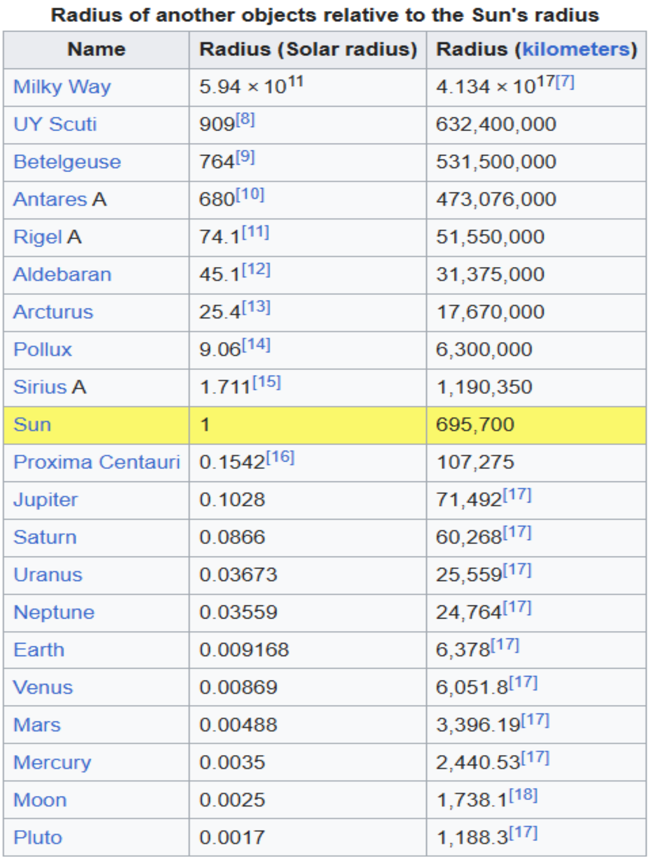
Figure 25: Column koi_steff zoomed

The chart shows:

- Each bar shows how many KOIs orbit stars in a given narrow temperature band within 5500 to 6000 K.
- This plot isolates just the stars that are like our Sun and there is a peak (the tall bar) near 5700 to 5800 K.
- There are 3411 sun-like stars in the chart.

These values could be important because the temperature of the star affects everything. Like in the column koi_insol on how much light and heat the planet receives. A planet orbiting a cool red dwarf might need to be super close to the star to get enough heat, or a planet orbiting a hot star might be far away but still be roasting. Maybe hotter stars have their habitable planets further out in the system?

Column: koi_slogg:



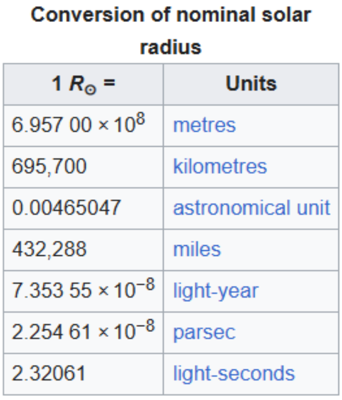
Radius of another objects relative to the Sun's radius

Name	Radius (Solar radius)	Radius (kilometers)
Milky Way	5.94 × 10^{11}	4.134 × 10^{17}[7]
UY Scuti	909[8]	632,400,000
Betelgeuse	764[9]	531,500,000
Antares A	680[10]	473,076,000
Rigel A	74.1[11]	51,550,000
Aldebaran	45.1[12]	31,375,000
Arcturus	25.4[13]	17,670,000
Pollux	9.06[14]	6,300,000
Sirius A	1.711[15]	1,190,350
Sun	1	695,700
Proxima Centauri	0.1542[16]	107,275
Jupiter	0.1028	71,492[17]
Saturn	0.0866	60,268[17]
Uranus	0.03673	25,559[17]
Neptune	0.03559	24,764[17]
Earth	0.009168	6,378[17]
Venus	0.00869	6,051.8[17]
Mars	0.00488	3,396.19[17]
Mercury	0.0035	2,440.53[17]
Moon	0.0025	1,738.1[18]
Pluto	0.0017	1,188.3[17]

Figure 26: https://en.wikipedia.org/wiki/Solar_radius

Slogg stands for “log g» and it is short for **logarithm of the gravitational acceleration.** It gives us the logarithmic surface gravity of the host star, and it is measured in log(cm/s^2). In other words, it’s a way of measuring how strong gravity is on the star’s surface, given values in log base 10 using centimeters per second squared as the unit.

The Sun (our Sun) for example has a surface gravity of about log g = 4.44. So, if we see a koi_slogg value close to that, we may be looking at a star that is pretty similar to our own star in terms of structure and compactness (Southworth, 2023). Giant stars have a log-g values of about 2.0 to 3.0 and supergiant stars have a very low surface gravity of <2.0 (Creevey, 2021). This information can help distinguish between what kind of star it is.



Conversion of nominal solar radius

1 $R_\odot$ =	Units
6.957 00 × 10^{8}	metres
695,700	kilometres
0.00465047	astronomical unit
432,288	miles
7.353 55 × 10^{-8}	light-year
2.254 61 × 10^{-8}	parsec
2.32061	light-seconds

Figure 27: https://en.wikipedia.org/wiki/Solar_radius

Column: koi_srad:

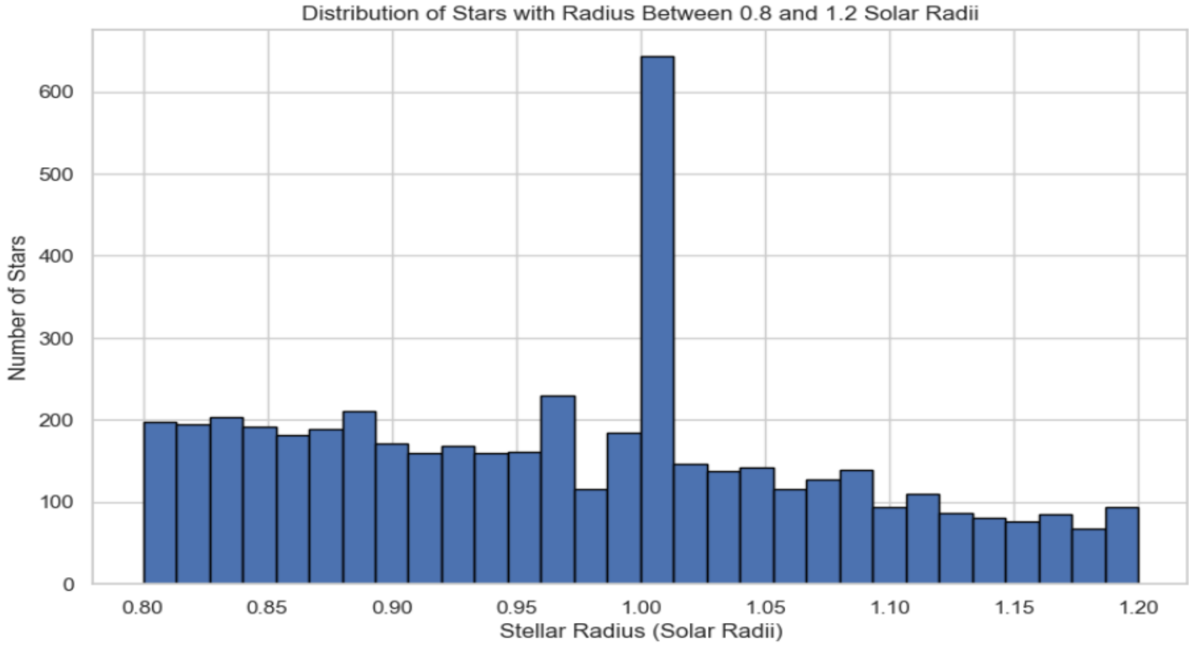
This column tells us about the **radius of the host star**, and it is expressed in units of the sun's radius (Archive, 2023). This is all about how big the star is, and it may be important because of the planet's size and temperature that is often measured relative to the stars it orbits (Mann, 2013).

The standard practice in astronomy when expressing stellar radii in units of our Sun's radius is detonated as 1R⊙ (figure 26). So, if the value is 1.0, then it is the same size as our star, the Sun.

- A 0.5 value is half the size of the Sun.
- A 2.0 value is twice as big as the Sun.
- A 10+ value – we are then looking at a giant or a supergiant star.

The deeper the transit is, the bigger the planet but only if the size of the star is known. We also remember that the planetary radius column koi_prad is calculated based on the depth of the light curve and the size of the star.

Let's show a histogram that displays the distribution of stellar radii with the range of 0.8 to 1.2 times the Suns's radius. Remember, our Sun's radius = 1.0.

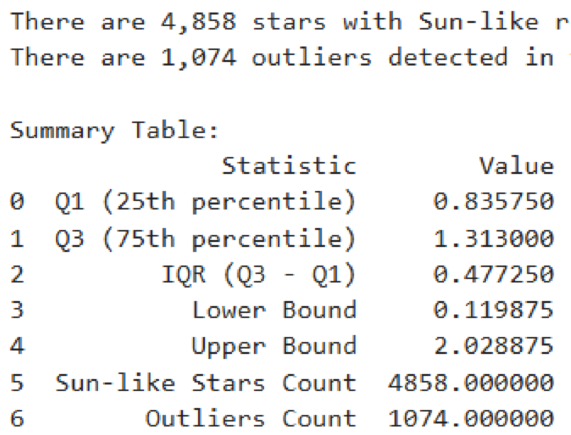


```
There are 4,858 stars with Sun-like radius (between 0.8 and 1.2 Solar radii)
```

Figure 28: Column koi_srad with sun-like radius

As we can see from the chart, the center is near 1.0. This could indicate that the Kepler dataset contains many stars that are similar to our own Sun.

I made a summary table that shows a descriptive statistics overview of the koi_srad column including the number of Sun-like stars and outliers based on the interquartile range (IQR) method:



```
There are 4,858 stars with Sun-like radius (between 0.8 and 1.2 Solar radii).
There are 1,074 outliers detected in the koi_srad column using the IQR method.

Summary Table:
              Statistic        Value
0  Q1 (25th percentile)     0.835750
1  Q3 (75th percentile)     1.313000
2          IQR (Q3 - Q1)     0.477250
3            Lower Bound     0.119875
4            Upper Bound     2.028875
5  Sun-like Stars Count  4858.000000
6        Outliers Count  1074.000000
```

Figure 29: Column koi_srad Statistic summary

- The Q1 (25th percentile) shows that 25 % of the dataset has a radius that is less than or equal to 0.83 solar radii. This tells us where lower quartile of the data lies. Mostly smaller stars than our Sun.

- The Q3 (75th percentile) shows that 75 % of the dataset has a radius that is less than or equal to 1.31 solar radii. This tells us the upper quartile values are just above or Sun's radii. It seems like many of them include G and F type stars (look at Figure 22, koi_steff).

- The IQR (Q3 – Q1) measures the spread of the middle 50 % of the data while ignoring outliers.

- The lower bound of 0.119 explains that any value below this is considered a lower outlier. The Upper bound of 2.028 explains that any value above this is considered an upper outlier. Maybe giant stars?

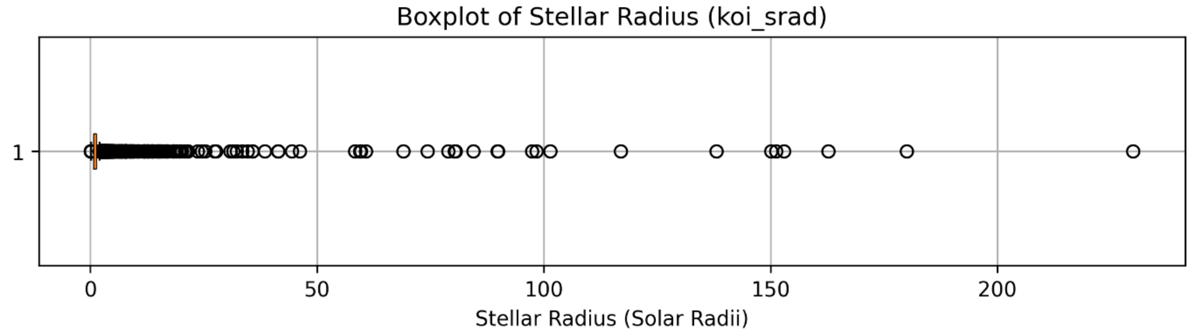


Figure 30: Column koi_srad boxpl. full

This boxplot shows how skewed/spread the full koi_srad dataset is. The long tail to the right shows very large stars. There is also a dense cluster near the 0, just like the chart with a radius between 0.8 to 1.2 solar radius.

Column: ra:

This column does not tell us anything about the physical properties of the star or planet. The column stands for **Right Ascension**, and it tells us the horizontal coordinates of the host star in the sky. It is measured in decimal degrees. I will not go further with this column because it has no function in finding differences between confirmed and false positives or habitability predictions.

Column: dec (Changed the column name to dec_deg because of SQL issues):

The word dec stands for **Declination**, which is the vertical coordinate on the celestial sphere. Just like column ra, dec is needed to pinpoint a star's location in space, longitude, and latitude. I will also not go further with this column because it won't help predict planet confirmation or habitability.

Column: koi_kepmag:

Koi_kepmag stands for **Kepler Magnitude**, which is a measure of how the host star appears to the Kepler Space Telescope. It is a part of Kepler's own custom photometric system and is specifically tuned to the Kepler's sensors. The values tell us that if the numbers are low, then it specifies brighter stars. If the numbers are high, then it specifies fainter stars (Wikipedia, Kepler space telescope, 2024).

- If the value in column koi_kepmag is 10, then it's a very bright star, high quality, low noise data, and ideal for detecting planetary transit.
- If the value in column koi_kepmag is from 14-16, then it probably is a moderate bright/faint star where the data quality is sufficient but may be affected by increased noise and potentially can complicate transit detection.
- If the value in column ko_kepmag is > 16, then it is probably fainter stars where the data is noisier. So detecting small planets may become significantly more challenging due to lower signal-to-noise ratios. When studying false positives, this column might help me notice that fainter stars have a higher rate of uncertainty.

6.9 HEATMAP CORRELATION OF CONFIRMED PLANETS VS. FALSE POSITIVE

To answer the question of what column features distinguish confirmed exoplanets from false positives, I chose to use a heatmap because it may give us a clear visual overview of how each column's features relate to each other. A heatmap could help highlight patterns by showing how strongly each variable is correlated with the column koi_disposition. This could help us spot which column might be useful for distinguishing real exoplanets from false positives, and which ones likely will not help us much.

That said, I did not start or rely on the heatmap alone. I wanted to better understand the dataset and its context by exploring almost each column individually. The reason for this was to give us a more complete picture of what each column represents, and how it was distributed.

In this heatmap, the columns that end in err1 and err2 were not included, because they only represent measurement errors and not actual values. Column koi_pdisposition was also taken out due to redundant with column koi_disposition. The columns kepid, kepoi_name, and kepler_name are just identifiers. The Candidate in the column koi_disposition is also not in this heatmap. The reason for this is that it is not ideal for this specific analysis. They haven't been ruled out or confirmed and they are unclear. This uncertainty makes them unreliable for correlation analysis, and they don't represent a clear label. That is why Candidate are not included. Let us look at the heatmap:

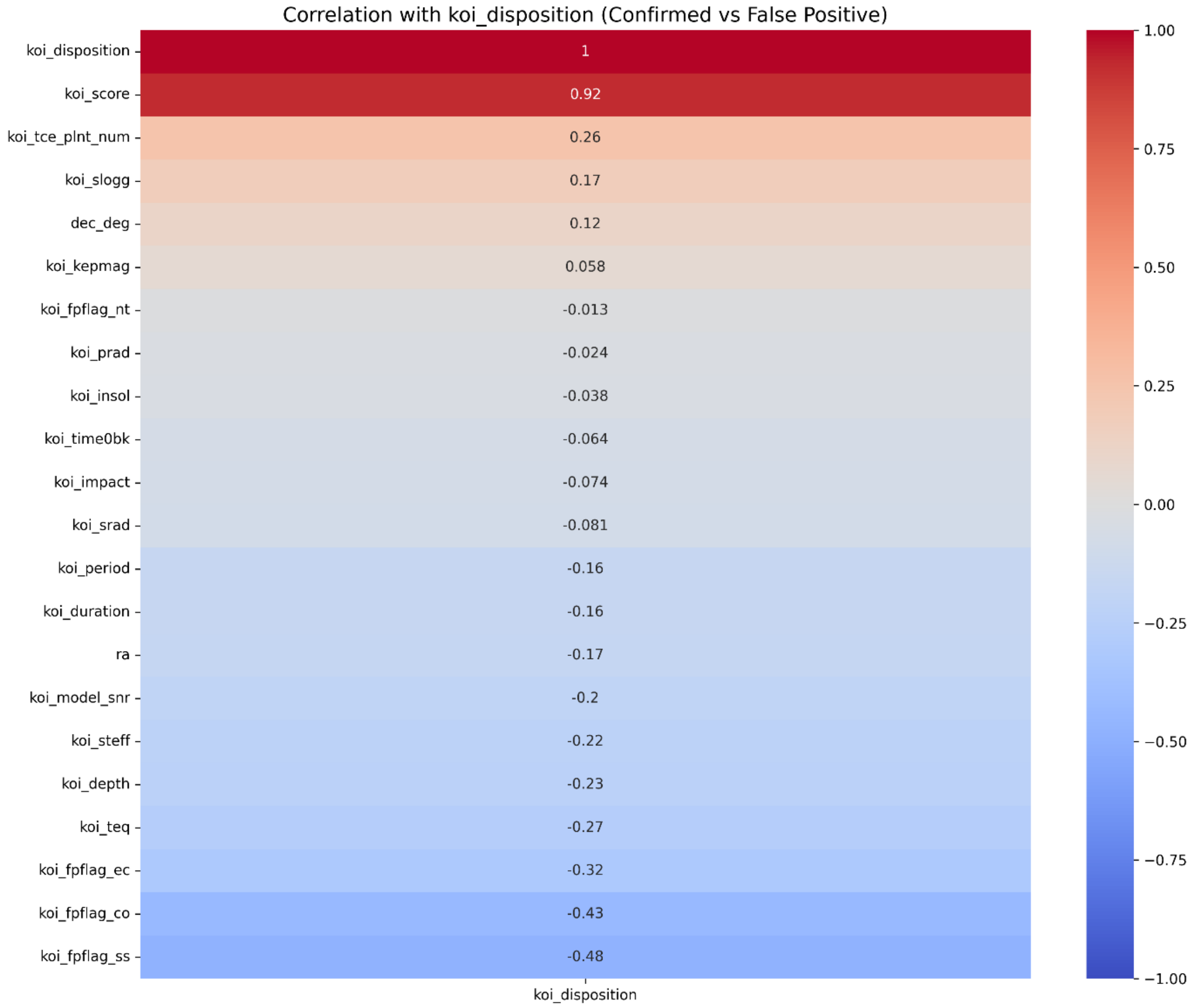


Figure 31: Heatmap of confirmed vs. false positive

The heatmap represents whether a KOI is a confirmed exoplanet (1) or a false positive (0). The values range from 1 to -1. Positive values in red means that the **higher** the values get, the more the column is associated with **confirmed exoplanets**.

The negative values in blue mean that the **lower** the values get, the more the planets are associated with **false positives**. Values that are close to 0 are weak or have no correlation. As we can see:

- The strongest positive correlation is the koi_score (0.92). This makes sense that this is highly correlated since this was calculated through the Kepler pipeline algorithms and validated by the experts on the likelihood of a real exoplanet.
- Koi_tce_plnt_num with 0.26 is the second most correlated with confirmed. As we mentioned earlier, confirmed planets may often appear in multi-planet systems, especially when the transit signals are consistent.
- The column koi_slogg correlates 0.17. As we earlier explained, this is the logarithm of the surface gravity of the star. Higher surface gravity may relate to stellar characteristics that reduce false positives.
- The strongest negatives are, without no surprise, the strongest negative correlation because the flags indicate a likely false positive.

Let us combine some columns based on this logic:

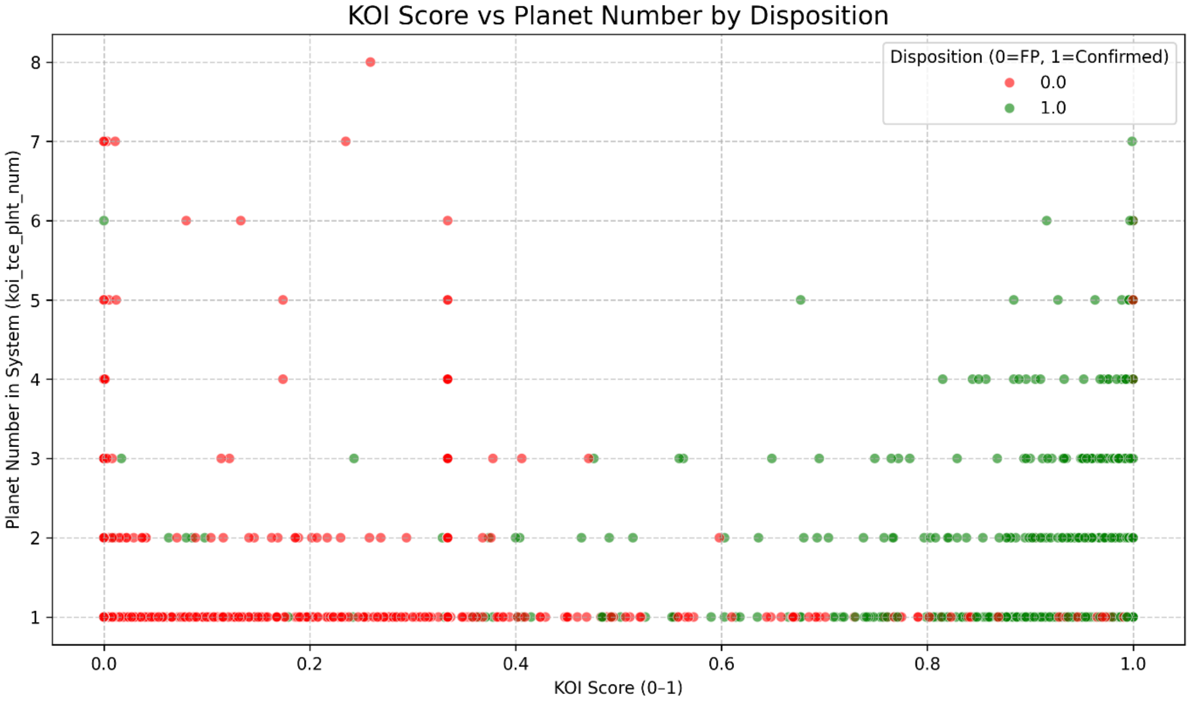


Figure 32: Scatterpl. Score vs. Planet Number

The chart shows:

- The X-axis is the koi_score and the Y-axis is the koi_tce_plnt_nm.
- The confirmed exoplanets cluster in the right corner. Often from multi planet systems > 1.
- The false positives dominate in the bottom left corner and almost always from single detection systems.
- This interprets the idea that high KOI confidence together with multi planet system is a good visual signal for confirmed exoplanets.

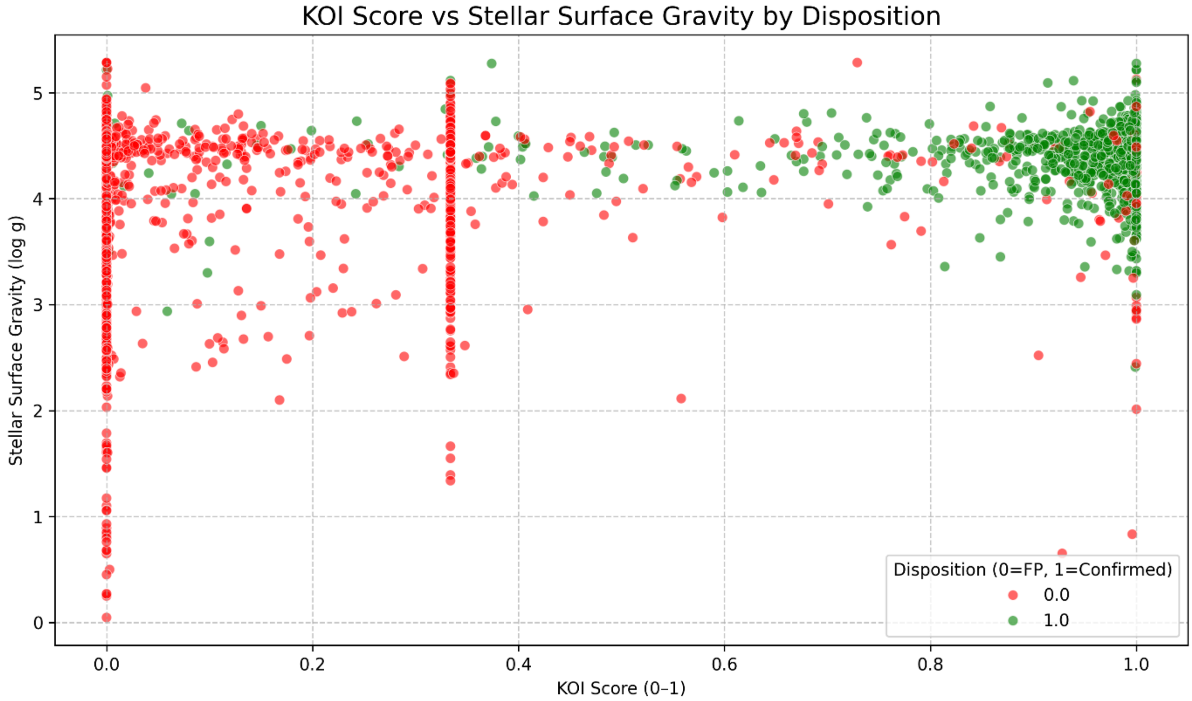


Figure 33: Scatterpl. koi_score and koi_slogg

The chart shows:

- The X-axis is the koi_score. Y-axis is the koi_slogg
- The confirmed planets are mostly found on the right side of the chart, close to 1.0. We mentioned earlier that the Sun (our Sun) for example has a surface gravity of about log g = 4.44. Most of the confirmed are around stars with moderate to high surface gravity 4 to 5.

- The false positives are at lower KOI scores and are around stars with slightly lower gravity. The column koi_slogg helps screen out unstable and complex stellar environments like giant stars or binaries that can be more prone to error.

The third combination we want to see is the correlation between koi_score and koi_model_snr with koi_disposition as the explained variable. This is because the column koi_model_snr stands for signal-to-noise ratio and detects how strong the detected transit signal is compared to the background noise. A high snr means the transit dip in brightness is strong and distinct, and a low snr is often hard to distinguish from noise, stellar activity, etc. and is likely to be false positive.

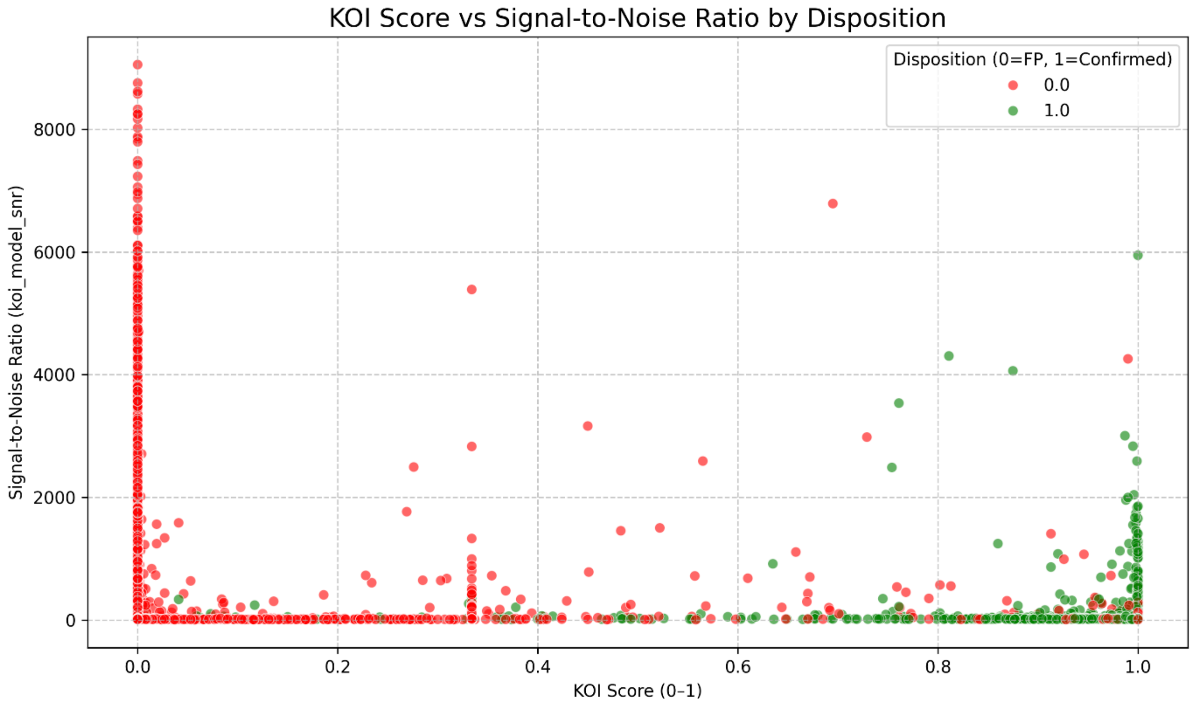


Figure 34: Scatterpl. koi_score and koi_model_snr

The scatterplot shows:

- The confirmed exoplanets are very clustered in the right quadrant.
- False positives are mainly appearing in the lower left.
- The two columns together with column koi_disposition form a clear separation line between real planets and false positives.

6.10 HEATMAP CORRELATION OF CONFIRMED EXOPLANETS ONLY

Now that we have seen what separates confirmed exoplanets and false positives from each other by using a heatmap and combining columns, let us now try to explore what features can be used to see potential habitability on the confirmed exoplanets by again using a heatmap. This is to help us identify which physical features are most relevant for studying potential habitability. Based on the EDA we did from most of the columns, not all columns are relevant for this purpose, so for this heatmap, I chose specific columns that explain the physical and orbital features that could maybe help us assess potential habitability in confirmed exoplanets. The columns are:

- Koi_prad - Planet radius in Earth radii. Planet size could be important for habitability. This could help us filter Earth-like candidates.
- Koi_teq – Equilibrium temperature in Kelvin. Too hot or too cold planets are maybe not habitable. This could identify if the planet lies in the habitable zone.
- Koi_insol – Stellar Isolation. This is relevant to our own Earth. This measures how much energy a planet receives from its star. Earth = 1.0 – planets close to this may be in the habitable zone.
- Koi_period – Orbital period in days. How long it takes a planet to orbit its star. Long periods could be planets that are further from its star.
- Koi_duration – the transit duration in hours. Can maybe help identify planets that are stable in orbits.
- Koi_depth – the transit depth in ppm. How much starlight is blocked during transit.
- Koi_impact – the impact parameter. Describes the path of the transit across the star.
- Koi_model_snr – Signal-to-noise ratio that measures how strong and clear the transit signal was. High snr was more trustworthy in detection.
- Koi_srad – stellar radius in solar radii. Sun-like stars, 1.0 solar radii, for a stable environment.
- Koi_steff – stellar effective temperature in Kelvin. This tells us how hot the star is. Could affect the habitable zone.
- Koi_slogg – stellar surface gravity (log g). This helps identify the star type.

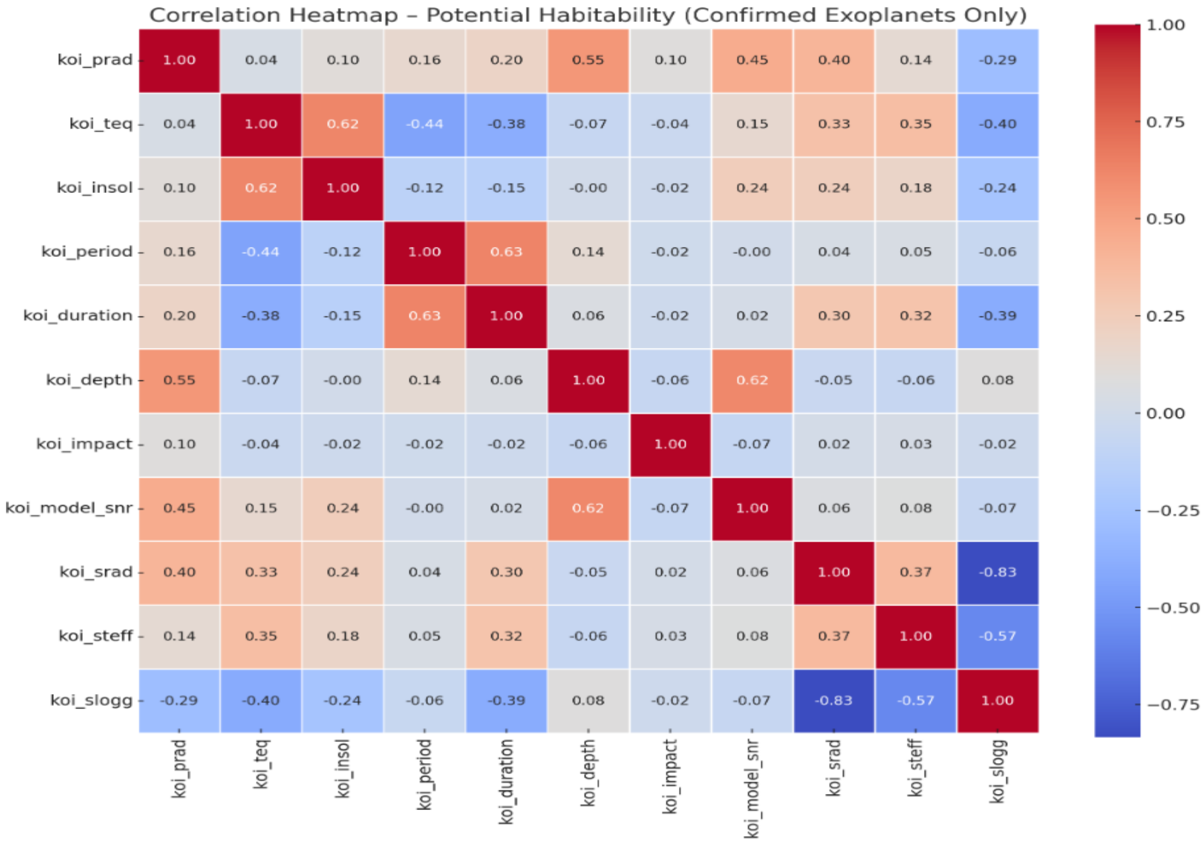


Figure 35: Heatmap on confirmed vs. explanatory columns

As we can see from the heatmap there are especially three combo columns that would be interesting to investigate a little bit more.

- koi_teq and koi_insol with correlation 0.62. These two combined could give us a strong signal for surface temperature.
- Koi_depth and koi_model_snr with correlation 0.62. Combination could be useful in detecting Earth-like planets.
- Koi_period and koi_duration with correlation 0.63. These two could maybe be useful in estimating orbital distance and transit time in planets that could lie in a habitable zone.

Let us start with koi_teq vs. koi_insol. Based on the EDA of those two columns, we highlighted that teq values of 250-300 K are kind of “Earth-like” planets (with no greenhouse effect). Insol column where a value of 1.0 means the planet receives the same amount of energy as Earth does from the sun (Archive, 2023), so the planet ranges within

0.25 to 4 (units) times Earth's stellar flux (Petigura, 2013). The chart with those estimates could look like this:

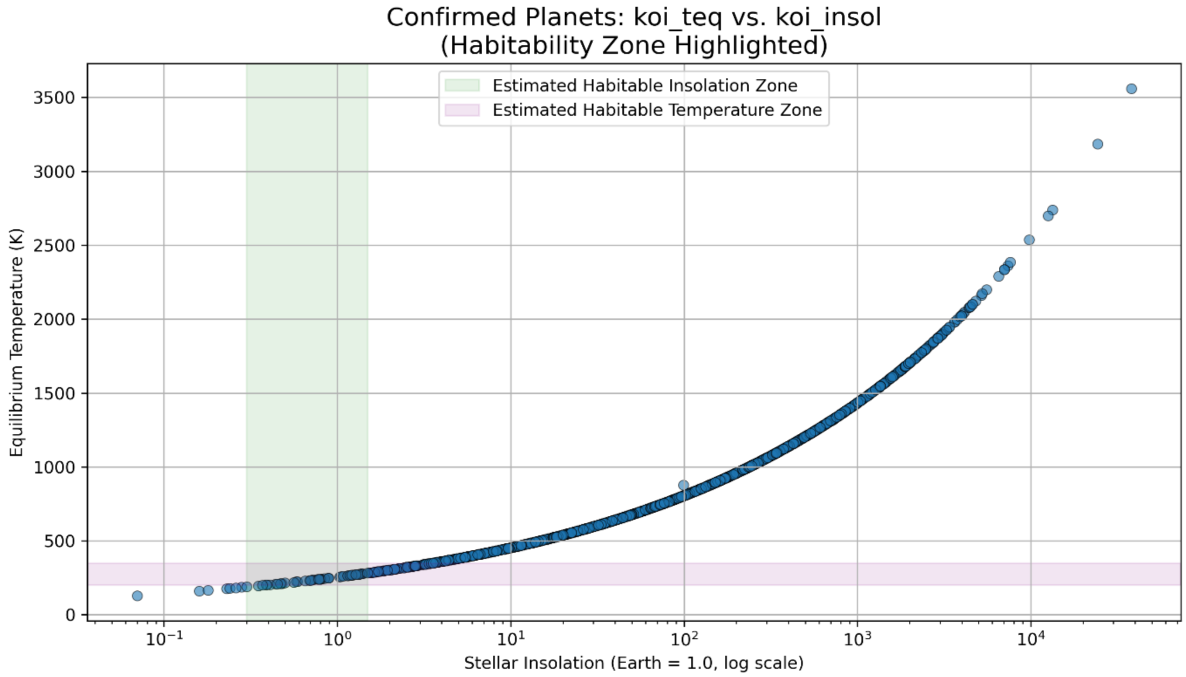


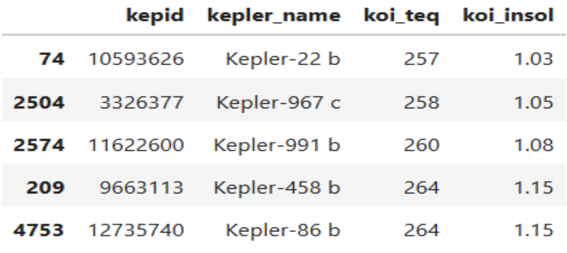
Figure 36: Scatterpl. koi_teq vs. koi_insol

- The green vertical band shows a stellar planet range shows a stellar isolation value between 0.3 and 1.5. Planets within this band could maybe be in the habitable zone.

- The purple horizontal band shows equilibrium temperatures range between 200 K and 350 K. Again, Earth is about 255 K without atmosphere greenhouse effect.

- The planets that are located within both bands could be the right distance from their star with a life-supporting temperature.

The top 5 planets that are within both bands:

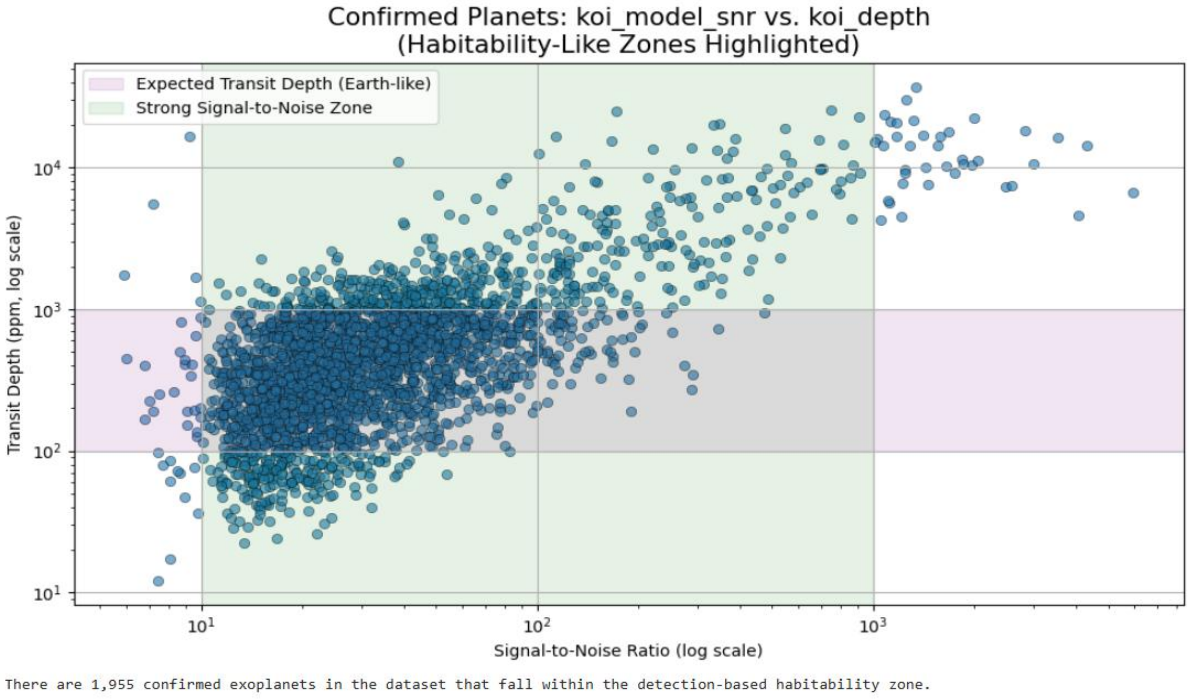


	kepid	kepler_name	koi_teq	koi_insol
74	10593626	Kepler-22 b	257	1.03
2504	3326377	Kepler-967 c	258	1.05
2574	11622600	Kepler-991 b	260	1.08
209	9663113	Kepler-458 b	264	1.15
4753	12735740	Kepler-86 b	264	1.15

Figure 37: Top 5 planets based on koi_teq and koi_insol

This list is based solely on temperature and stellar energy that is between values of: koi_teq 250 K to 300 K and koi_insol between 0.5 to 1.5. On that note, the value range may exclude other potential habitat-confirmed exoplanets.

The koi_depth vs. koi_model_snr based on the EDA of those two columns, we highlighted that depth values that are high > 1000 could depend on the size of the planet compared to the star. So, the bigger the ppm value the more it blocks the light of the star. Smaller ppm values like Earth-size only causes a tiny dip (Wikipedia, 2024). Column koi_model_snr where a value detection threshold of 7.1 means that any transit above the threshold was considered significant (high SNR signals mean a strong probability that it is a planet (Jenkins, 2019). If we use a value between 10 and 1000 ppm in the SNR and 100 to 1000 ppm in depth we get this chart:



```
There are 1,955 confirmed exoplanets in the dataset that fall within the detection-based habitability zone.
```

Figure 38: Scatterpl. koi_model_snr vs. koi_depth

The log scale y-axis is the distance between each step. So for example, from 10^2 to 10^3 the signal gets 10 X stronger. A value of 100 ppm is then 10^2 – Shallow transit that could be small planets. A value of 1000 ppm is then 10^3 – Deeper transit could be Earth-like to big Neptune planets.

The log scale x-axis measures the transit signal on how strong it is. So for example, a value of 10 SNR is then 10^1 – That could be just over the minimum acceptable signal. A value of 100 SNR is then 10^2 – That could be a strong signal/detection. 1000 SNR is then 10^3 which could be a very strong signal or maybe a large or a planet close to the star.

The chart also shows us the planets that are located within both bands, and there are 1955 confirmed exoplanets in this range.

The koi_duration vs. koi_period based on the EDA of those two columns, we highlighted that duration values are given in hours. How long it takes for a planet to pass in front of its star. The period column tells us about the orbital period on how long it takes to make one full trip around its star. The value is in days. If we use the same log g method between these two columns (like in the previous chart) with koi_period values between 50 and 500 days and koi_duration values between 5 and 15 hours, we get this chart:

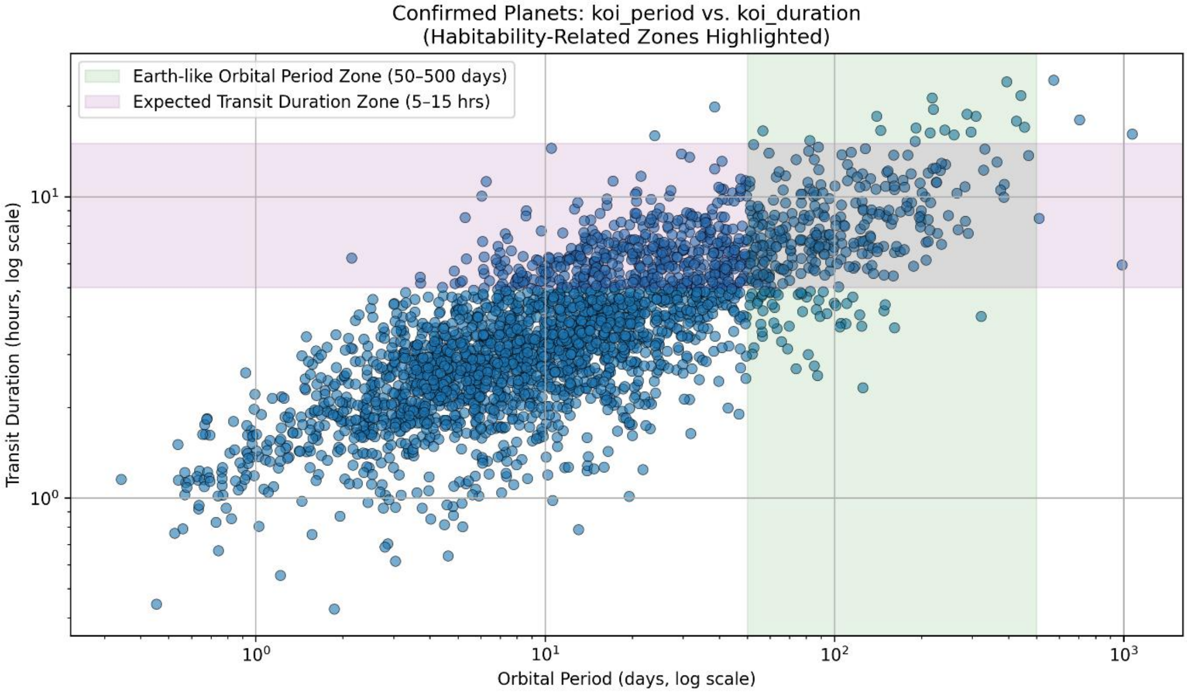
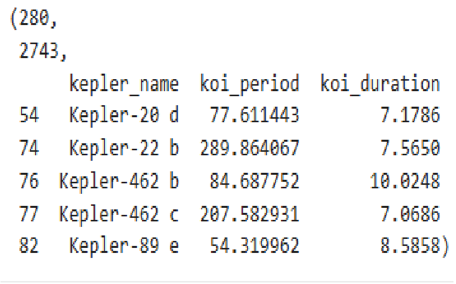


Figure 39: Scatterpl. koi_period vs. koi_duration

The log scale y-axis is the distance between each step. So for example, from 10^1 to 10^2 the signal gets 10 X stronger. A value of 10 hours is then 10^1 – A very short and fast-moving planet. A value of 100 hours is then 10^2 – A near-typical Earth-like transit.

The log scale x-axis measures how many days it takes to complete one full orbit around the star. So for example, a value of 10 days is then 10^1 – That's a 10-day orbit. A value of 100 days is then 10^2 – That's a 100-day orbit around its star.



```
(280,
 2743,
      kepler_name  koi_period  koi_duration
 54   Kepler-20 d   77.611443        7.1786
 74   Kepler-22 b  289.864067        7.5650
 76  Kepler-462 b   84.687752       10.0248
 77  Kepler-462 c  207.582931        7.0686
 82   Kepler-89 e   54.319962        8.5858)
```

Figure 40: List of confirmed planets in the habitability zone

Out of 2743 confirmed planets, 280 of them fall inside the overlapping habitability zone. The 5 rows under are example planets in the zone.

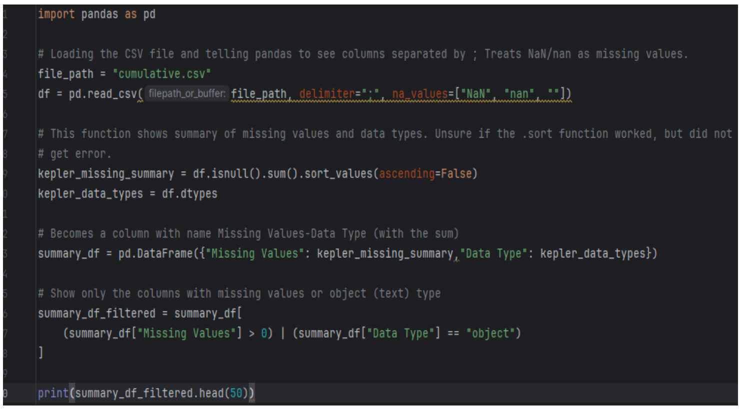
7 Data pre-processing

The Kepler data was downloaded directly from NASA's archive:

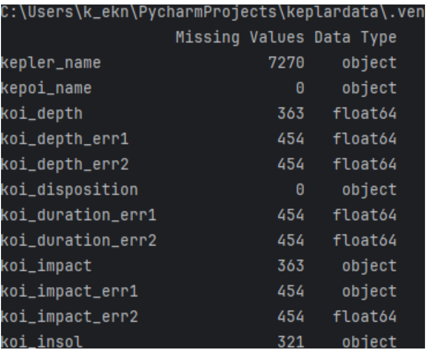
https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=cumulative

The dataset was first downloaded from Kaggle.com. When analyzing the data by manual scrutiny, I noticed a lot of errors. That is why I used NASA's archive instead.

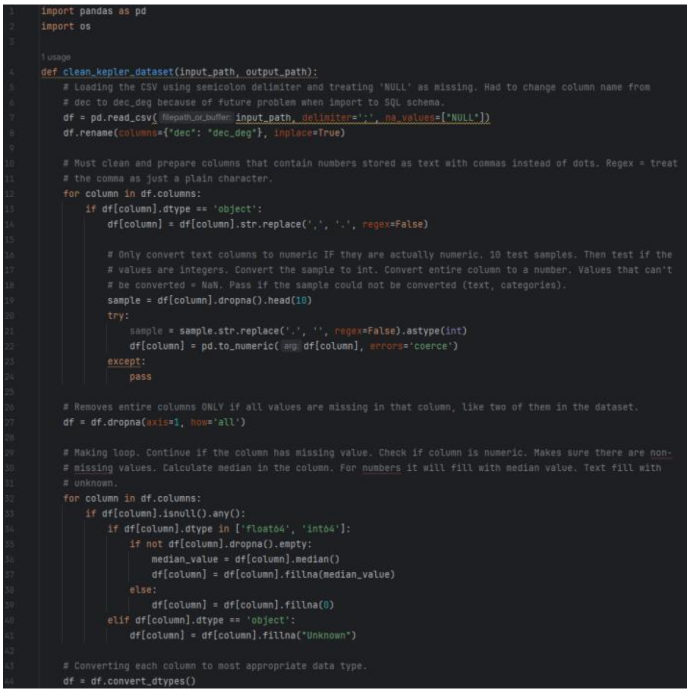
To see how the dataset was, I imported the CSV file (From text/CSV) in Excel's Power Query Editor and Transform Data. The data had many empty cells in almost every column just by visually investigating. In contrast to the Kaggle dataset, the NASA dataset wasn't too bad despite missing values. Used python for finding missing values:



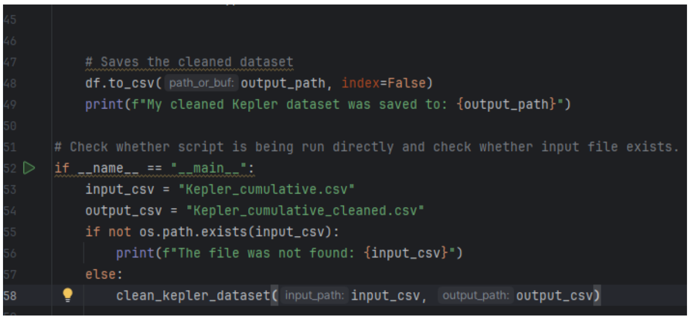
```
import pandas as pd

# Loading the CSV file and telling pandas to see columns separated by ; Treats NaN/nan as missing values.
file_path = "cumulative.csv"
df = pd.read_csv( filepath_or_buffer: file_path, delimiter=";", na_values=["NaN", "nan", ""])

# This function shows summary of missing values and data types. Unsure if the .sort function worked, but did not
# get error.
kepler_missing_summary = df.isnull().sum().sort_values(ascending=False)
kepler_data_types = df.dtypes

# Becomes a column with name Missing Values-Data Type (with the sum)
summary_df = pd.DataFrame({"Missing Values": kepler_missing_summary,"Data Type": kepler_data_types})

# Show only the columns with missing values or object (text) type
summary_df_filtered = summary_df[
    (summary_df["Missing Values"] > 0) | (summary_df["Data Type"] == "object")
]

print(summary_df_filtered.head(50))
```



```
C:\Users\k_ekn\PycharmProjects\keplardata\.ven
                  Missing Values Data Type
kepler_name                 7270    object
kepoi_name                     0    object
koi_depth                    363   float64
koi_depth_err1               454   float64
koi_depth_err2               454   float64
koi_disposition                0    object
koi_duration_err1            454   float64
koi_duration_err2            454   float64
koi_impact                   363    object
koi_impact_err1              454    object
koi_impact_err2              454   float64
koi_insol                    321    object
```



```
import pandas as pd
import os

1 usage
def clean_kepler_dataset(input_path, output_path):
    # Loading the CSV using semicolon delimiter and treating 'NULL' as missing. Had to change column name from
    # dec to dec_deg because of future problem when import to SQL schema.
    df = pd.read_csv( filepath_or_buffer: input_path, delimiter=';', na_values=["NULL"])
    df.rename(columns={"dec": "dec_deg"}, inplace=True)

    # Must clean and prepare columns that contain numbers stored as text with commas instead of dots. Regex = treat
    # the comma as just a plain character.
    for column in df.columns:
        if df[column].dtype == 'object':
            df[column] = df[column].str.replace(',', '.', regex=False)

            # Only convert text columns to numeric IF they are actually numeric. 10 test samples. Then test if the
            # values are integers. Convert the sample to int. Convert entire column to a number. Values that can't
            # be converted = NaN. Pass if the sample could not be converted (text, categories).
            sample = df[column].dropna().head(10)
            try:
                sample = sample.str.replace('.', '', regex=False).astype(int)
                df[column] = pd.to_numeric( arg: df[column], errors='coerce')
            except:
                pass

    # Removes entire columns ONLY if all values are missing in that column, like two of them in the dataset.
    df = df.dropna(axis=1, how='all')

    # Making loop. Continue if the column has missing value. Check if column is numeric. Makes sure there are non-
    # missing values. Calculate median in the column. For numbers it will fill with median value. Text fill with
    # unknown.
    for column in df.columns:
        if df[column].isnull().any():
            if df[column].dtype in ['float64', 'int64']:
                if not df[column].dropna().empty:
                    median_value = df[column].median()
                    df[column] = df[column].fillna(median_value)
                else:
                    df[column] = df[column].fillna(0)
            elif df[column].dtype == 'object':
                df[column] = df[column].fillna("Unknown")

    # Converting each column to most appropriate data type.
    df = df.convert_dtypes()
```

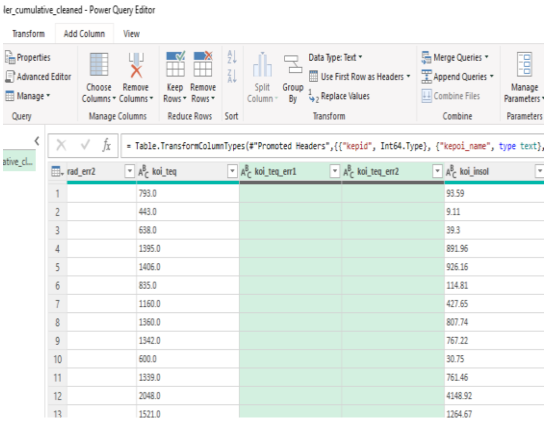


```


    # Saves the cleaned dataset
    df.to_csv( path_or_buf: output_path, index=False)
    print(f"My cleaned Kepler dataset was saved to: {output_path}")

# Check whether script is being run directly and check whether input file exists.
if __name__ == "__main__":
    input_csv = "Kepler_cumulative.csv"
    output_csv = "Kepler_cumulative_cleaned.csv"
    if not os.path.exists(input_csv):
        print(f"The file was not found: {input_csv}")
    else:
        clean_kepler_dataset( input_path: input_csv,  output_path: output_csv)
```

There were 2 columns with no values at all, as you can see in the code (left image). I chose to not include them in the dataset.

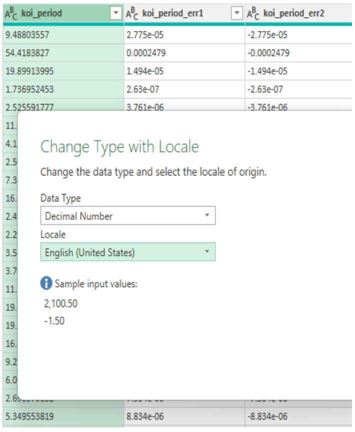
Text columns that had NaN values like kepler_name were replaced with "Unknown". Missing values in the numeric columns were replaced with median values. Categorical columns were replaced with median values (though I think I made a little mistake, maybe I should have replaced them with mode values). The Python script (image on the left side) with some detailed markdowns for explanation.

You can also see that the numeric columns are text-type. Almost all of the numeric values had datatype text. Had to replace the values in Power Quary so that it is possible to work with the dataset for instance on a spreadsheet.



```
= Table.TransformColumnTypes(#"Promoted Headers",{{"kepid", Int64.Type}, {"kepoi_name", type text},
```

	rad_err2	koi_teq	koi_teq_err1	koi_teq_err2	koi_insol
1		793.0			93.59
2		443.0			9.11
3		638.0			39.3
4		1395.0			891.96
5		1406.0			926.16
6		835.0			114.81
7		1160.0			427.65
8		1360.0			807.74
9		1342.0			767.22
10		600.0			30.75
11		1339.0			761.46
12		2048.0			4148.92
13		1521.0			1264.67



koi_period	koi_period_err1	koi_period_err2
9.48803557	2.775e-05	-2.775e-05
54.4183827	0.0002479	-0.0002479
19.89913995	1.494e-05	-1.494e-05
1.736952453	2.63e-07	-2.63e-07
2.525591777	3.761e-06	-3.761e-06
11.		
4.1		
2.5		
7.3		
16.		
2.4		
2.2		
3.5		
3.7		
11.		
19.		
19.		
16.		
9.2		
6.0		
2.6		
5.349553819	8.834e-06	-8.834e-06

Change Type with Locale

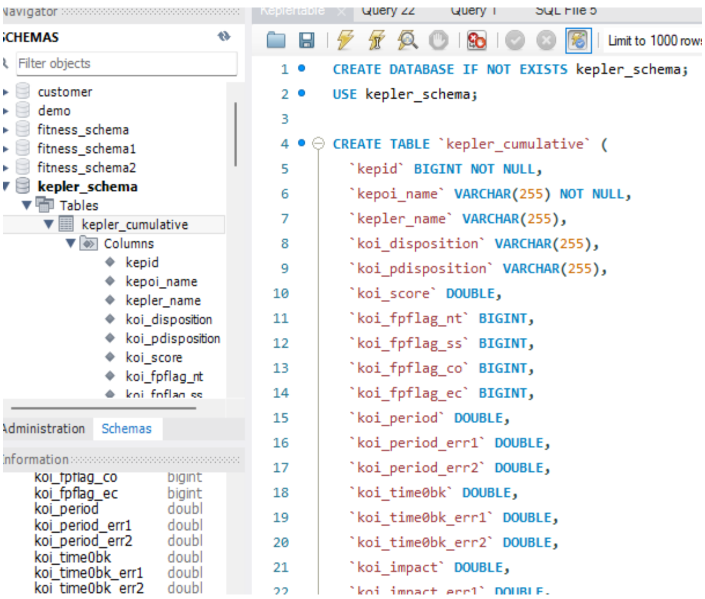
Change the data type and select the locale of origin.

Data Type: Decimal Number

Locale: English (United States)

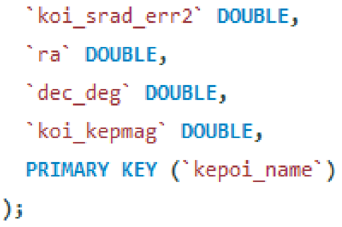
Sample input values:
2,100.50
-1.50

The next step was to import the newly cleaned CSV file into a database using SQL. Made a new schema called kepler_schema, then created a table called kepler_cumulative. Did a lot of tweaking in finding the right datatypes and constraints for the columns. There was a total of 5 object columns. The table is shown below:



```
koi_fpflag_co      bigint
koi_fpflag_ec      bigint
koi_period         doubl
koi_period_err1    doubl
koi_period_err2    doubl
koi_time0bk        doubl
koi_time0bk_err1   doubl
koi_time0bk_err2   doubl
```

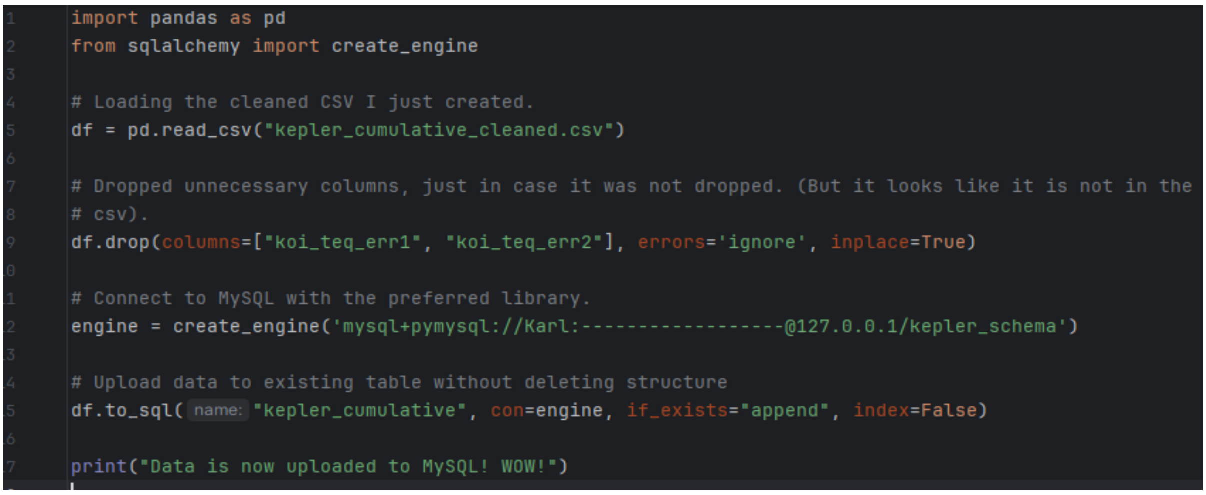
```
CREATE DATABASE IF NOT EXISTS kepler_schema;
USE kepler_schema;

CREATE TABLE `kepler_cumulative` (
  `kepid` BIGINT NOT NULL,
  `kepoi_name` VARCHAR(255) NOT NULL,
  `kepler_name` VARCHAR(255),
  `koi_disposition` VARCHAR(255),
  `koi_pdisposition` VARCHAR(255),
  `koi_score` DOUBLE,
  `koi_fpflag_nt` BIGINT,
  `koi_fpflag_ss` BIGINT,
  `koi_fpflag_co` BIGINT,
  `koi_fpflag_ec` BIGINT,
  `koi_period` DOUBLE,
  `koi_period_err1` DOUBLE,
  `koi_period_err2` DOUBLE,
  `koi_time0bk` DOUBLE,
  `koi_time0bk_err1` DOUBLE,
  `koi_time0bk_err2` DOUBLE,
  `koi_impact` DOUBLE,
  `koi_impact_err1` DOUBLE,
```



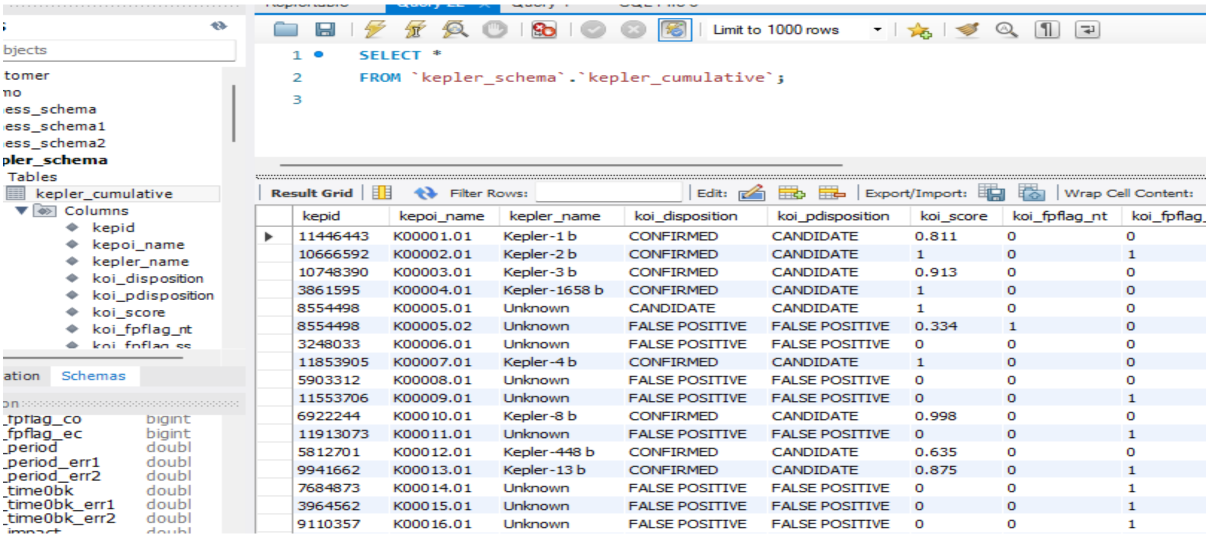
```
  `koi_srad_err2` DOUBLE,
  `ra` DOUBLE,
  `dec_deg` DOUBLE,
  `koi_kepmag` DOUBLE,
  PRIMARY KEY (`kepoi_name`)
);
```

Some of the columns have datatypes as DOUBLE to store numbers with decimal point double precision because of the wide numeric range in some of the columns. Marked the column “kepoi_name” as the unique identifier. By the way, there were no duplicates in the dataset.

Finally, imported the cleaned CSV into the SQL schema.



```python
import pandas as pd
from sqlalchemy import create_engine

# Loading the cleaned CSV I just created.
df = pd.read_csv("kepler_cumulative_cleaned.csv")

# Dropped unnecessary columns, just in case it was not dropped. (But it looks like it is not in the
# csv).
df.drop(columns=["koi_teq_err1", "koi_teq_err2"], errors='ignore', inplace=True)

# Connect to MySQL with the preferred library.
engine = create_engine('mysql+pymysql://Karl:-----------------@127.0.0.1/kepler_schema')

# Upload data to existing table without deleting structure
df.to_sql( name: "kepler_cumulative", con=engine, if_exists="append", index=False)

print("Data is now uploaded to MySQL! WOW!")
```



```
bjects
tomer
no
ess_schema
ess_schema1
ess_schema2
pler_schema
 Tables
  kepler_cumulative
   Columns
    kepid
    kepoi_name
    kepler_name
    koi_disposition
    koi_pdisposition
    koi_score
    koi_fpflag_nt
    koi_fpflag_ss
```

```
_fpflag_co       bigint
_fpflag_ec       bigint
_period          doubl
_period_err1     doubl
_period_err2     doubl
_time0bk         doubl
_time0bk_err1    doubl
_time0bk_err2    doubl
_impact          doubl
```

```sql
SELECT *
FROM `kepler_schema`.`kepler_cumulative`;
```

kepid	kepoi_name	kepler_name	koi_disposition	koi_pdisposition	koi_score	koi_fpflag_nt	koi_fpflag_
11446443	K00001.01	Kepler-1 b	CONFIRMED	CANDIDATE	0.811	0	0
10666592	K00002.01	Kepler-2 b	CONFIRMED	CANDIDATE	1	0	1
10748390	K00003.01	Kepler-3 b	CONFIRMED	CANDIDATE	0.913	0	0
3861595	K00004.01	Kepler-1658 b	CONFIRMED	CANDIDATE	1	0	0
8554498	K00005.01	Unknown	CANDIDATE	CANDIDATE	1	0	0
8554498	K00005.02	Unknown	FALSE POSITIVE	FALSE POSITIVE	0.334	1	0
3248033	K00006.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	0
11853905	K00007.01	Kepler-4 b	CONFIRMED	CANDIDATE	1	0	0
5903312	K00008.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	0
11553706	K00009.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	1
6922244	K00010.01	Kepler-8 b	CONFIRMED	CANDIDATE	0.998	0	0
11913073	K00011.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	1
5812701	K00012.01	Kepler-448 b	CONFIRMED	CANDIDATE	0.635	0	0
9941662	K00013.01	Kepler-13 b	CONFIRMED	CANDIDATE	0.875	0	1
7684873	K00014.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	1
3964562	K00015.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	1
9110357	K00016.01	Unknown	FALSE POSITIVE	FALSE POSITIVE	0	0	1

8 Measuring performance

As this project aims to build classification models to help sort the confirmed exoplanets from false positives by using the Kepler dataset, it can be important to define in advance how model performance will be measured. Since confirmed vs. false positive is a binary classification problem, these evaluation metrics have been selected:

Accuracy: because this measures how correct the model is. This, by the way, can also be misleading if the dataset is imbalanced (higher numbers of false planets than confirmed).

Precision: It can be important to minimize the number of false positives. In this dataset, it could ensure that we don't wrongly label non-planets as confirmed exoplanets.

Recall: We must measure how many planets we successfully have identified. This could be useful if we aim to get them all, even if it means more false positives.

F1 Score: We then need a balance that considers precision and recall. This could be relevant when false positives and false negatives are important to control.

(Accuracy, Precision, Recall, and F1 Score: Information is taken from https://noroff.bravais.com/s/p0pRPjqWnsj68qgoOZHY (BDT) Model evaluation and selection).

Roc-Auc Score: This can evaluate the model and its ability to distinguish between the classes across all thresholds (Team, 2025). It can give a sense of how well the model separates confirmed planets from the false positives overall.

All of these metrics are selected to ensure a balanced model. Once I have trained the classification model, the dataset will be split into a training set and a test set to validate its model performance. Maybe we can also apply a cross-validation to avoid overfitting.

9 Big data/machine learning discussion

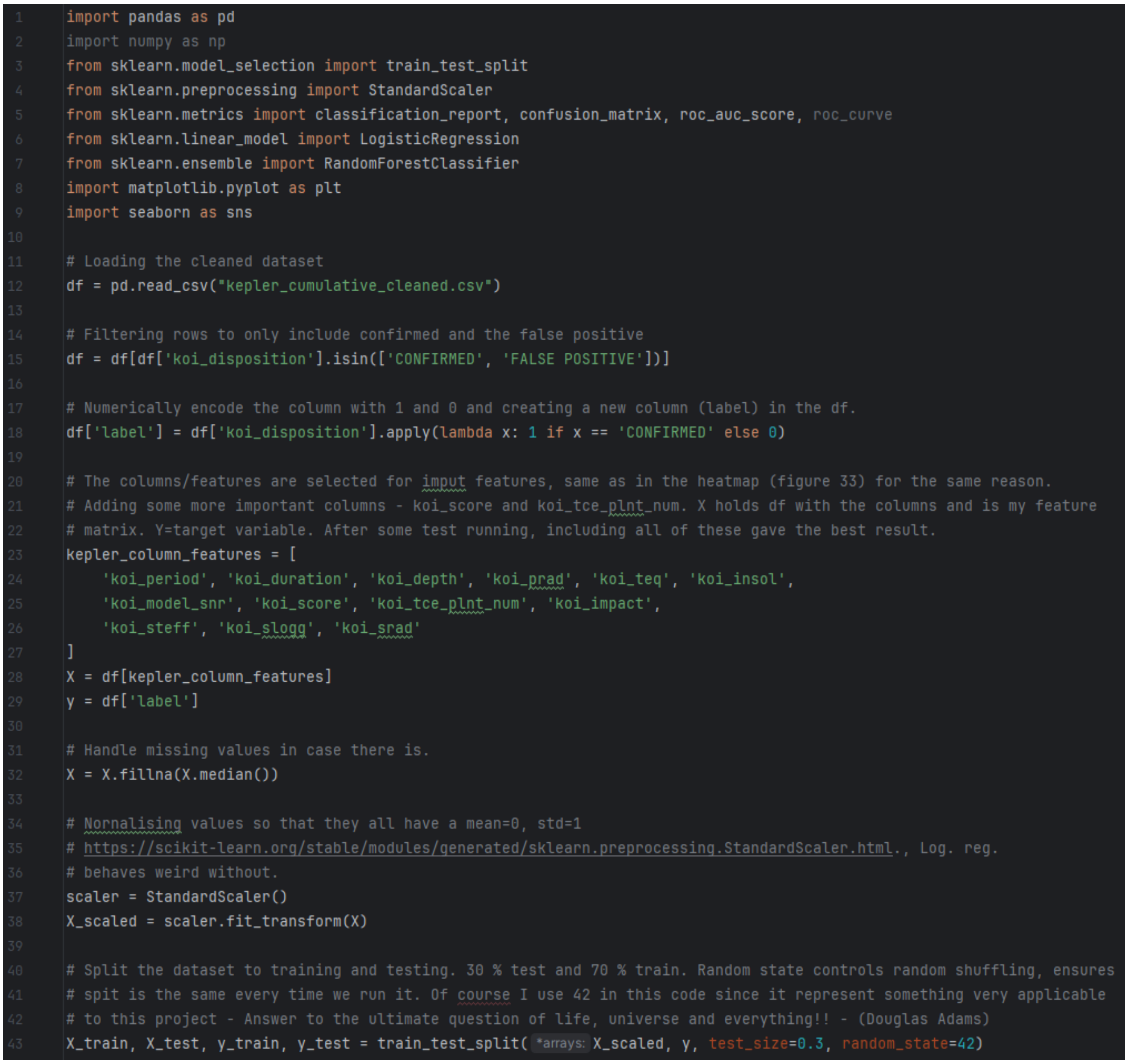
To classify the Kepler Objects of Interest (KOI's) as confirmed exoplanets or false positives, we have to apply machine learning by using Python/Jupyter Notebook. Given the relatively large and complex structure of the KOI dataset with over 9500 rows and 47 columns, manual analysis alone may not be consistent enough. Machine learning will hopefully provide this project with a better tool to detect patterns and make predictions.

I am planning to use classification algorithms such as Logistic Regression and Random Forest to train predictive models. The models will have to learn the features of the columns such as orbital period, planetary radius, equilibrium temperature, signal-to-noise ratio, and more. The target or the explained variable is the koi_disposition which was converted to binary labels where 0 = False Positive, 1 = Confirmed.

As mentioned, the platform for this task is Jupyter Notebook with Python. This is because it is flexible and is integrated with data science libraries such as Pandas, Scikit-learn, Seaborn, and Matplotlib, from importing the data, and cleaning it, through column selection, training, and performance evaluation.

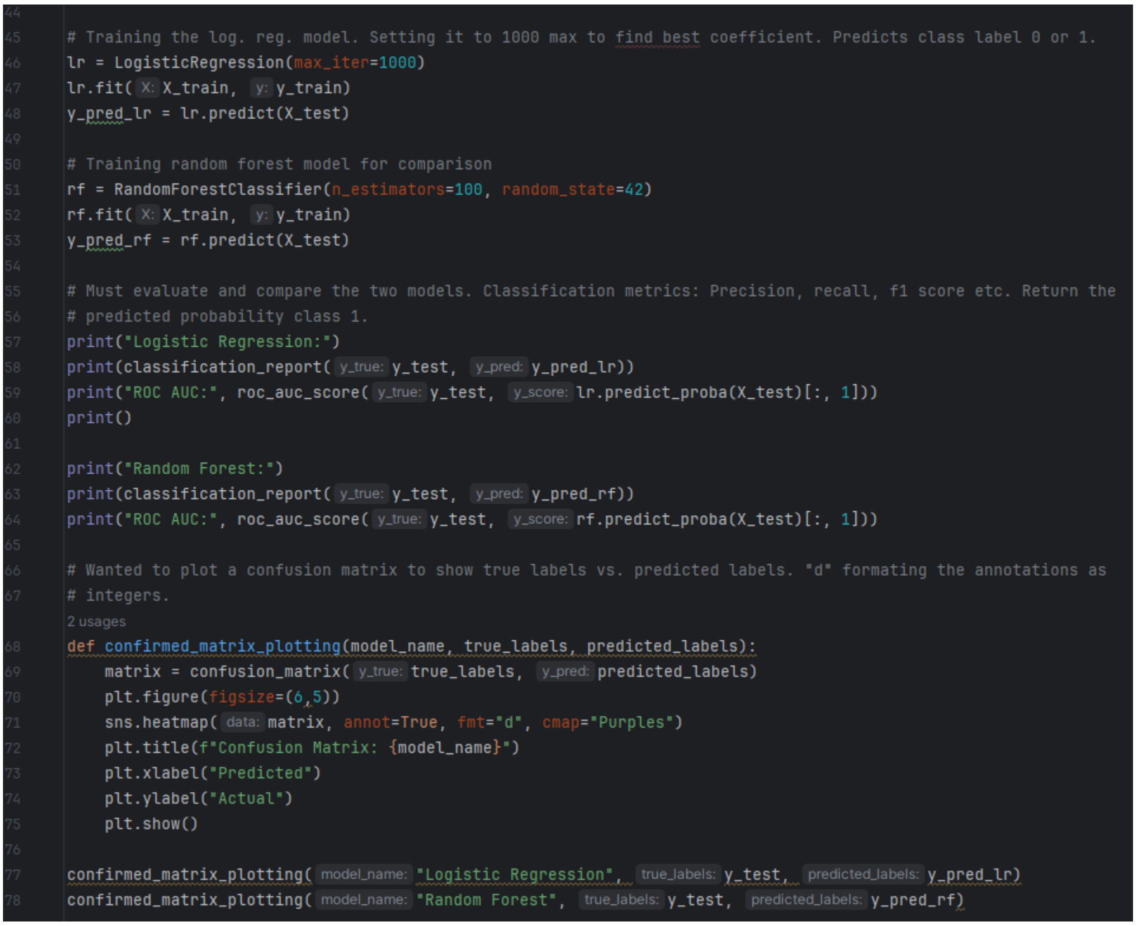
After the model has been trained, its performance will be assessed in Accuracy, Precision, Recall, F1 Score, and ROC-AUC. This can ensure that the results will be reliable.

Let us train the model for prediction:



```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
import seaborn as sns

# Loading the cleaned dataset
df = pd.read_csv("kepler_cumulative_cleaned.csv")

# Filtering rows to only include confirmed and the false positive
df = df[df['koi_disposition'].isin(['CONFIRMED', 'FALSE POSITIVE'])]

# Numerically encode the column with 1 and 0 and creating a new column (label) in the df.
df['label'] = df['koi_disposition'].apply(lambda x: 1 if x == 'CONFIRMED' else 0)

# The columns/features are selected for imput features, same as in the heatmap (figure 33) for the same reason.
# Adding some more important columns - koi_score and koi_tce_plnt_num. X holds df with the columns and is my feature
# matrix. Y=target variable. After some test running, including all of these gave the best result.
kepler_column_features = [
    'koi_period', 'koi_duration', 'koi_depth', 'koi_prad', 'koi_teq', 'koi_insol',
    'koi_model_snr', 'koi_score', 'koi_tce_plnt_num', 'koi_impact',
    'koi_steff', 'koi_slogg', 'koi_srad'
]
X = df[kepler_column_features]
y = df['label']

# Handle missing values in case there is.
X = X.fillna(X.median())

# Nornalising values so that they all have a mean=0, std=1
# https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html., Log. reg.
# behaves weird without.
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split the dataset to training and testing. 30 % test and 70 % train. Random state controls random shuffling, ensures
# spit is the same every time we run it. Of course I use 42 in this code since it represent something very applicable
# to this project - Answer to the ultimate question of life, universe and everything!! - (Douglas Adams)
X_train, X_test, y_train, y_test = train_test_split( *arrays: X_scaled, y, test_size=0.3, random_state=42)
```

Figure 41: Python script - Log. Reg. and Rand. Forest part 1

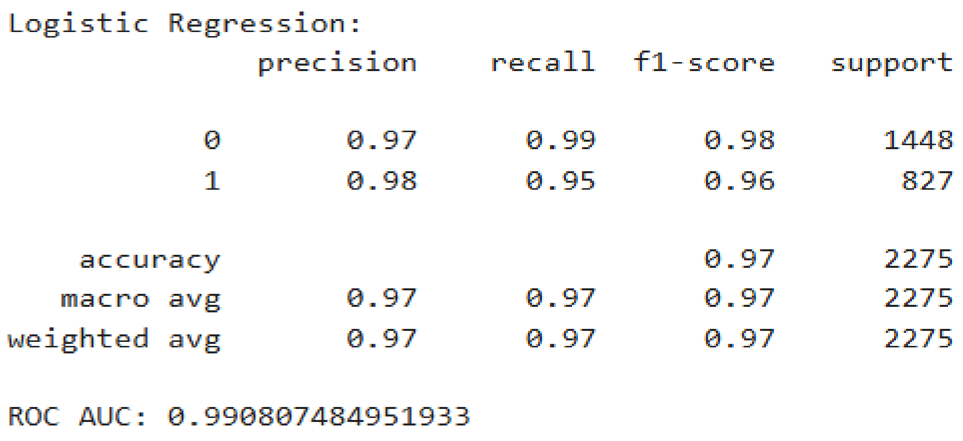


```
# Training the log. reg. model. Setting it to 1000 max to find best coefficient. Predicts class label 0 or 1.
lr = LogisticRegression(max_iter=1000)
lr.fit( X: X_train,  y: y_train)
y_pred_lr = lr.predict(X_test)

# Training random forest model for comparison
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit( X: X_train,  y: y_train)
y_pred_rf = rf.predict(X_test)

# Must evaluate and compare the two models. Classification metrics: Precision, recall, f1 score etc. Return the
# predicted probability class 1.
print("Logistic Regression:")
print(classification_report( y_true: y_test,  y_pred: y_pred_lr))
print("ROC AUC:", roc_auc_score( y_true: y_test,  y_score: lr.predict_proba(X_test)[:, 1]))
print()

print("Random Forest:")
print(classification_report( y_true: y_test,  y_pred: y_pred_rf))
print("ROC AUC:", roc_auc_score( y_true: y_test,  y_score: rf.predict_proba(X_test)[:, 1]))

# Wanted to plot a confusion matrix to show true labels vs. predicted labels. "d" formating the annotations as
# integers.
2 usages
def confirmed_matrix_plotting(model_name, true_labels, predicted_labels):
    matrix = confusion_matrix( y_true: true_labels,  y_pred: predicted_labels)
    plt.figure(figsize=(6,5))
    sns.heatmap( data: matrix, annot=True, fmt="d", cmap="Purples")
    plt.title(f"Confusion Matrix: {model_name}")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.show()

confirmed_matrix_plotting( model_name: "Logistic Regression",  true_labels: y_test,  predicted_labels: y_pred_lr)
confirmed_matrix_plotting( model_name: "Random Forest",  true_labels: y_test,  predicted_labels: y_pred_rf)
```

Figure 42: Python script - Log. Reg. and Rand. Forest part 2

Building this script was quite challenging, but at the end it worked!

Logistic Regression result:



```
Logistic Regression:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1448
           1       0.98      0.95      0.96       827

    accuracy                           0.97      2275
   macro avg       0.97      0.97      0.97      2275
weighted avg       0.97      0.97      0.97      2275

ROC AUC: 0.990807484951933
```

Figure 43: Result of Logistic Regression

Let us break down this result:

False Positives = 0:

- **Precision:** Of all the KOIs **predicted** as false positives, the model shows that 97 % were correct.
- **Recall:** Of all the **actual** false positives, the model caught 99 %.
- **F1 Score:** It gave us an almost perfect balance of 98 %.
- Support: There were 1448 false positives in the test.

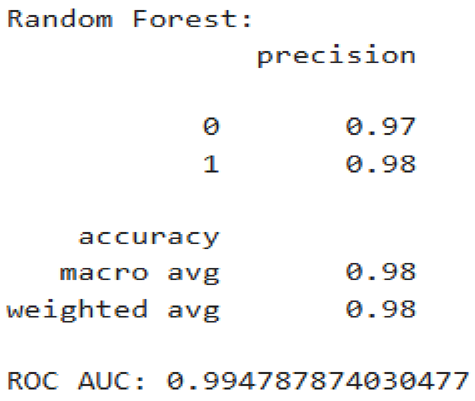
Confirmed Exoplanets = 1:

- **Precision:** Of all the KOIs **predicted** as confirmed exoplanets, the model shows that 98 % were correct.
- **Recall:** Of all the **actual** confirmed exoplanets, the model identified 95 % successfully.
- **F1 Score:** It gave us 96 %, so a few real exoplanets were missed.
- **Support:** There were 827 confirmed exoplanets in the test.

Accuracy: 97 %, shows the percentage of all the predictions that were correct. That is regardless of the class. Total of 2275 test examples.

ROC AUC: Shows a 99 % confidence and accuracy in the classification of confirmed exoplanets vs. false positives. This is crazy good!

Random Forest result:



```
Random Forest:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1448
           1       0.98      0.95      0.97       827

    accuracy                           0.98      2275
   macro avg       0.98      0.97      0.97      2275
weighted avg       0.98      0.98      0.98      2275

ROC AUC: 0.994787874030477
```

Figure 44: Result of Random Forest

Let us break down this result:

False Positives = 0:

- **Precision:** Of all the KOIs **predicted** as false positives, the model shows that 97 % were correct.
- **Recall:** Of all the **actual** false positives, the model caught 99 %.
- **F1 Score:** It gave us an almost perfect balance of 98 %.
- **Support:** There were 1448 false positives in the test.

Confirmed Exoplanets = 1:

- **Precision:** Of all the KOIs **predicted** as confirmed exoplanets, the model shows that 98 % were correct.
- **Recall:** Of all the **actual** confirmed exoplanets, the model identified 95 % successfully.
- **F1 Score:** It gave us 97 %, (1 % more accurate than logistic regression) so a few real exoplanets were missed.
- **Support:** There were 827 confirmed exoplanets in the test.

Accuracy: 98 %, (1 % more accurate than logistic regression) shows the percentage of all the predictions that were correct. That is regardless of the class.

ROC AUC: Shows a 99 % (lightly more confidence/accurate 0.9947 than logistic regression) confidence and accuracy in the classification of confirmed exoplanets vs. false positive. Still is crazy good!

Let’s see the confusion matrix:

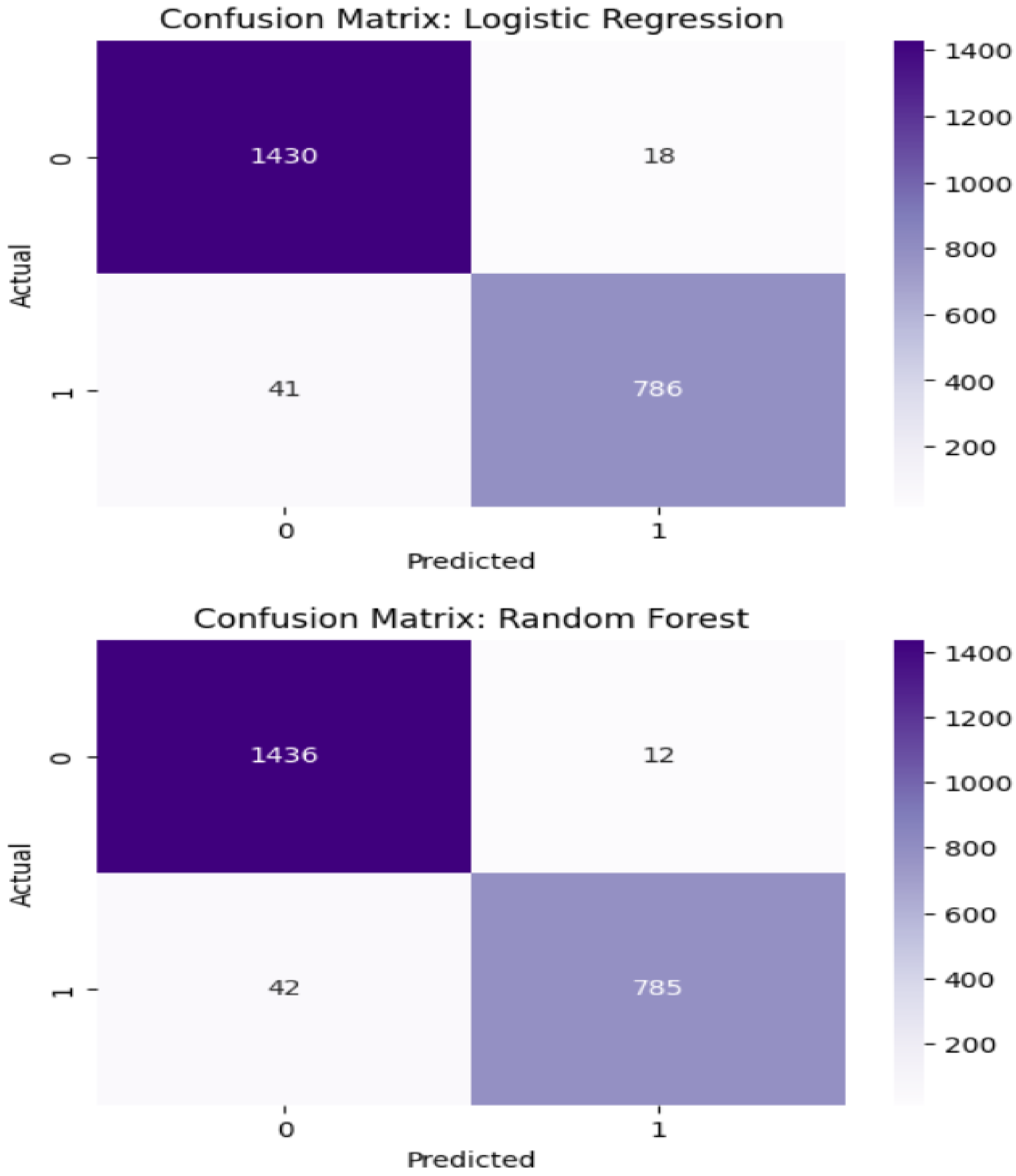
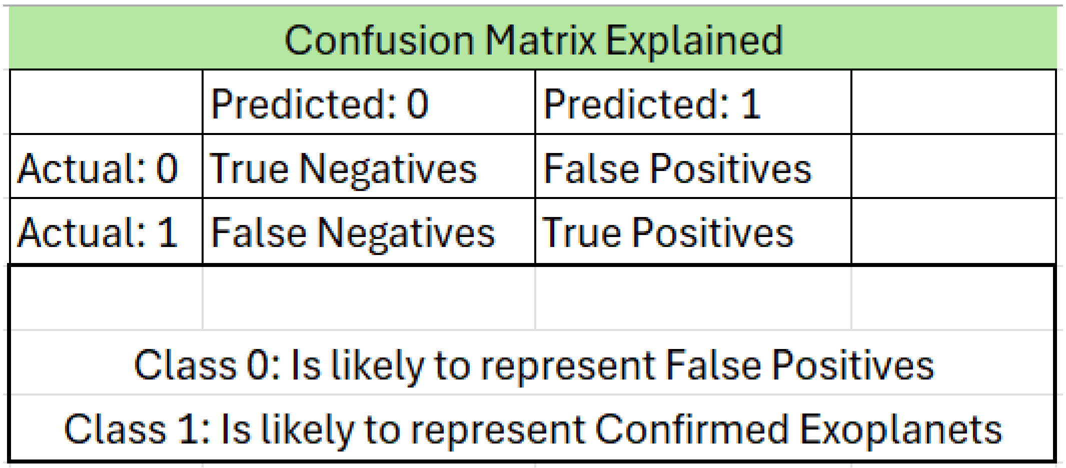


Figure 45: Confusion Matrix

This confusion matrix will help us evaluate the performance of both models a little more. It compares the actual values vs. the predicted values.



Confusion Matrix Explained

	Predicted: 0	Predicted: 1	
Actual: 0	True Negatives	False Positives	
Actual: 1	False Negatives	True Positives	

Class 0: Is likely to represent False Positives

Class 1: Is likely to represent Confirmed Exoplanets

Figure 46: Confusion Matrix explained

Logistic Regression interpretation of the confusion matrix:

- **True Negatives:** There were 1430 correctly predicted false positives.
- **False Positives:** There were 18 that were incorrectly predicted as confirmed, but were false.
- **False Negatives:** There were 41 incorrectly predicted as false but was actually confirmed.
- **True Negatives:** There were 786 that were correctly predicted as confirmed planets.

Random Forest interpretation of the confusion matrix:

- **True Negatives:** There were 1436 correctly predicted false positives.
- **False Positives:** There were 12 that were incorrectly predicted as confirmed, but were false.
- **False Negatives:** There were 42 incorrectly predicted as false but was actually confirmed.
- **True Negatives:** There were 785 that were correctly predicted as confirmed planets.

Overall: Random Forest has a slightly better performance in terms of reducing false positives, so this model is the best-preforming classifier for the task.

Now our next goal is to classify whether a confirmed exoplanet is potentially habitable. We have to base this on two very important columns. These two columns have been selected after exploring and investigating columns, plus the result from the heatmap in Figure 34. Let us recap the two columns:

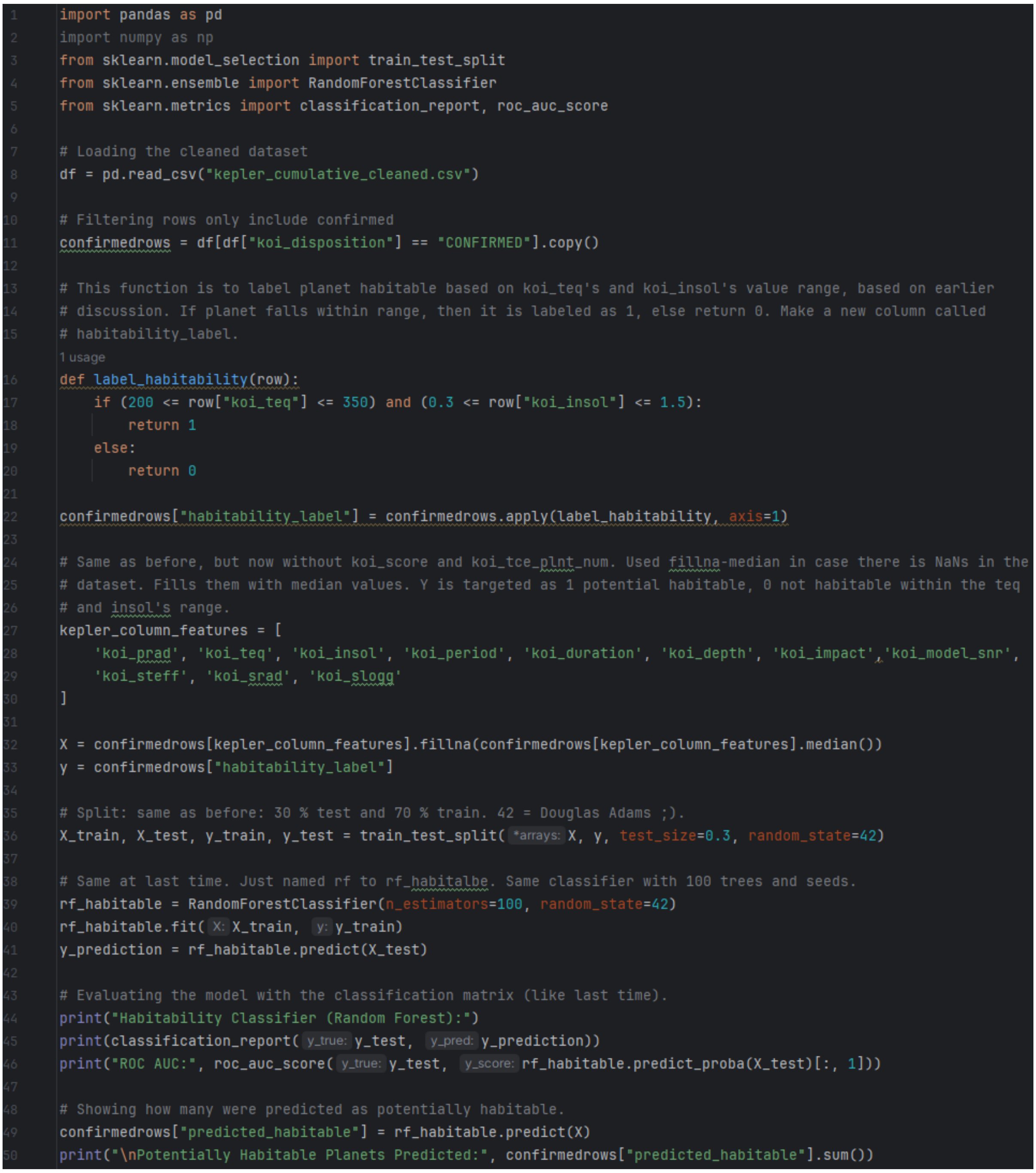
- **Koi_teq – Equilibrium temperature in Kelvin**. Too hot or too cold planets are maybe not habitable. This could identify if the planet lies in the habitable zone.
- **Koi_insol – Stellar Isolation**. This is relevant to our own Earth. This measures how much energy a planet receives from its star. Earth = 1.0 – planets close to this may be in the habitable zone.

- **koi_teq and koi_insol with correlation 0.62. These two combined could give us a strong signal for surface temperature.**

As in the scatterplot in Figure 35, we will use the same criteria.

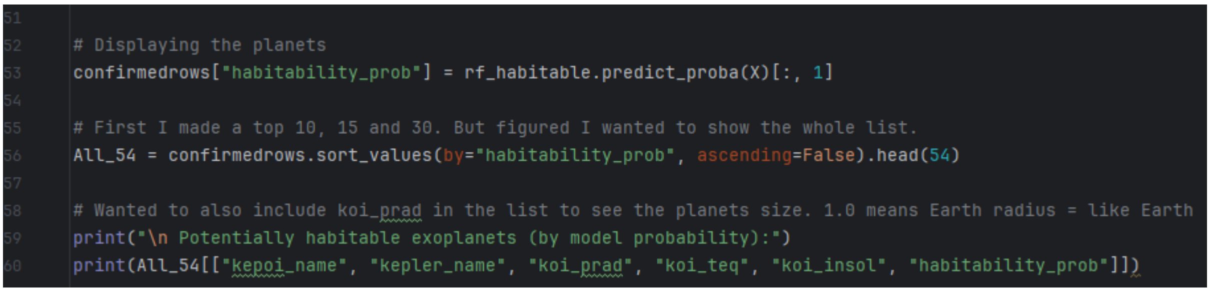
- Teq (Equilibrium Temperature) values between 200 K and 350 K.
- Insol (Steller isolation) between 0.3 and 1.5 Earth units.
- Koi_disposition is only labeled as confirmed exoplanets. The features are the same as the heatmap in Figure 34.

The code is visualized and explained on the next page. For this training model, we choose Random Forest.



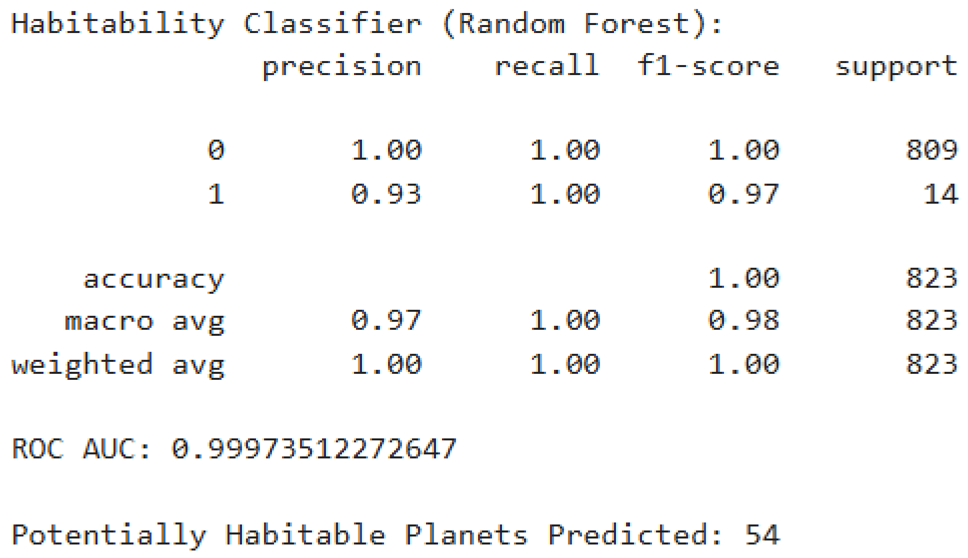
```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, roc_auc_score

# Loading the cleaned dataset
df = pd.read_csv("kepler_cumulative_cleaned.csv")

# Filtering rows only include confirmed
confirmedrows = df[df["koi_disposition"] == "CONFIRMED"].copy()

# This function is to label planet habitable based on koi_teq's and koi_insol's value range, based on earlier
# discussion. If planet falls within range, then it is labeled as 1, else return 0. Make a new column called
# habitability_label.
def label_habitability(row):
    if (200 <= row["koi_teq"] <= 350) and (0.3 <= row["koi_insol"] <= 1.5):
        return 1
    else:
        return 0

confirmedrows["habitability_label"] = confirmedrows.apply(label_habitability, axis=1)

# Same as before, but now without koi_score and koi_tce_plnt_num. Used fillna-median in case there is NaNs in the
# dataset. Fills them with median values. Y is targeted as 1 potential habitable, 0 not habitable within the teq
# and insol's range.
kepler_column_features = [
    'koi_prad', 'koi_teq', 'koi_insol', 'koi_period', 'koi_duration', 'koi_depth', 'koi_impact','koi_model_snr',
    'koi_steff', 'koi_srad', 'koi_slogg'
]

X = confirmedrows[kepler_column_features].fillna(confirmedrows[kepler_column_features].median())
y = confirmedrows["habitability_label"]

# Split: same as before: 30 % test and 70 % train. 42 = Douglas Adams ;).
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Same at last time. Just named rf to rf_habitalbe. Same classifier with 100 trees and seeds.
rf_habitable = RandomForestClassifier(n_estimators=100, random_state=42)
rf_habitable.fit(X_train, y_train)
y_prediction = rf_habitable.predict(X_test)

# Evaluating the model with the classification matrix (like last time).
print("Habitability Classifier (Random Forest):")
print(classification_report(y_test, y_prediction))
print("ROC AUC:", roc_auc_score(y_test, rf_habitable.predict_proba(X_test)[:, 1]))

# Showing how many were predicted as potentially habitable.
confirmedrows["predicted_habitable"] = rf_habitable.predict(X)
print("\nPotentially Habitable Planets Predicted:", confirmedrows["predicted_habitable"].sum())
```

Figure 47. Python script training model habitability part 1



```python

# Displaying the planets
confirmedrows["habitability_prob"] = rf_habitable.predict_proba(X)[:, 1]

# First I made a top 10, 15 and 30. But figured I wanted to show the whole list.
All_54 = confirmedrows.sort_values(by="habitability_prob", ascending=False).head(54)

# Wanted to also include koi_prad in the list to see the planets size. 1.0 means Earth radius = like Earth
print("\n Potentially habitable exoplanets (by model probability):")
print(All_54[["kepoi_name", "kepler_name", "koi_prad", "koi_teq", "koi_insol", "habitability_prob"]])
```

Figure 48: Python script training model habitability part 2



```
Habitability Classifier (Random Forest):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       809
           1       0.93      1.00      0.97        14

    accuracy                           1.00       823
   macro avg       0.97      1.00      0.98       823
weighted avg       1.00      1.00      1.00       823

ROC AUC: 0.99973512272647

Potentially Habitable Planets Predicted: 54
```

Figure 49: Result of the habitiability classifier Random Forest

Let us break down the result:

Not habitable = 0:

- **Precision:** Every time the model predicted not habitable, it gave us a 100 %, meaning that it was correct 100 % of the time. Recall and F1 Score did the same.
- **Support:** The model found all 809 not habitable planets.

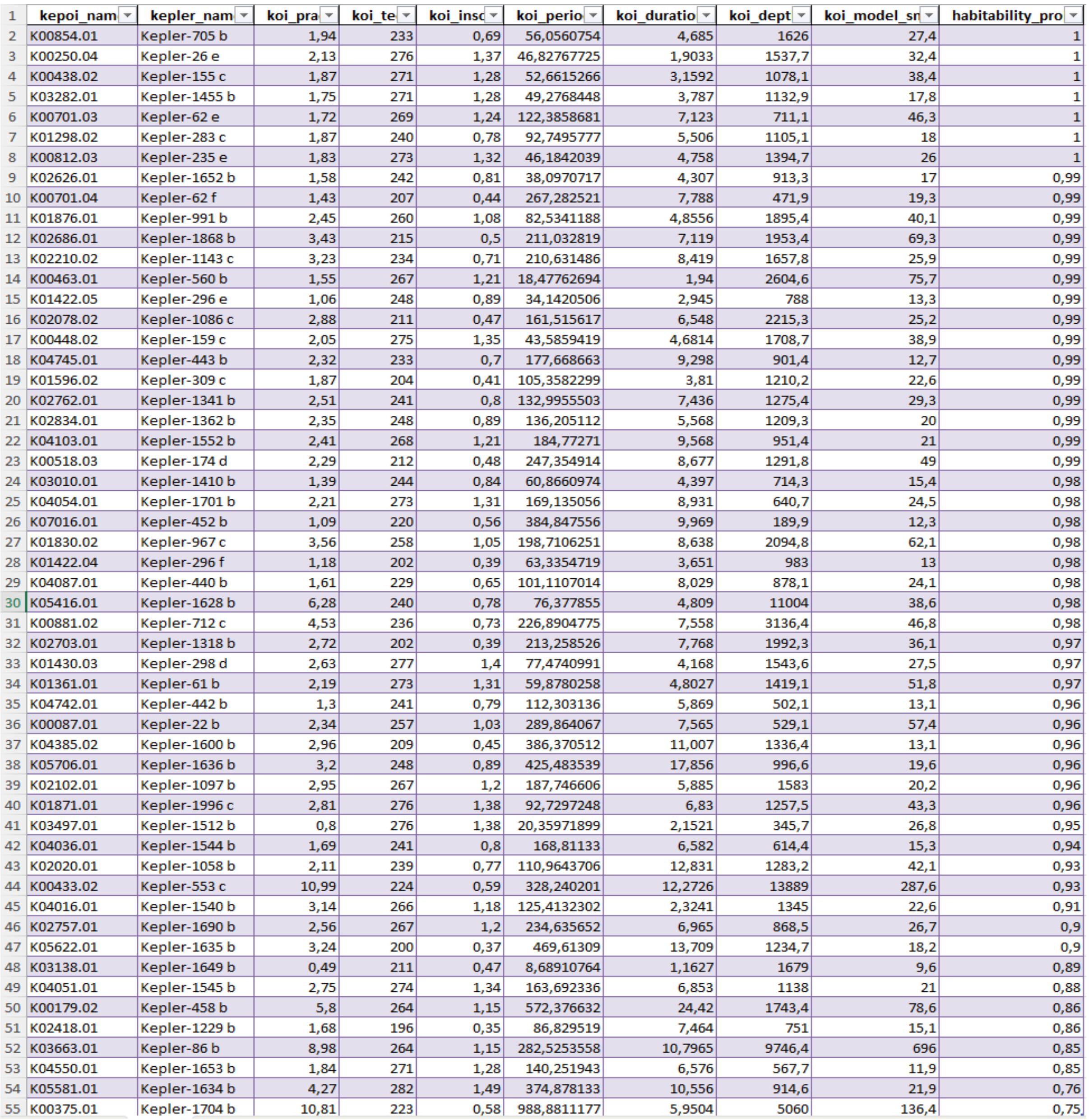
Potentially Habitable = 1:

- **Precision:** Of all the KOIs **predicted** as habitable, the model shows that 93 % were potentially habitable.
- **Recall:** 100 % shows that it caught all 14 habitable planets in the test set. (I tried to make a script that could show the 14 planets, but I failed in making the code).
- **F1 Score:** It gave us 97 %, a very good balance between precision and recall.
- **Support:** Says that there were 809 not habitable planets and 14 habitable planets.

Accuracy: 100 %. All 823 predictions were correct.

ROC AUC: Shows a 99 % (0.9997). A nearly perfect model that separates habitable from nonhabitable planets in the classification threshold.

It also says that there are potentially 54 habitable planets predicted. This means that 14 habitable planets have been labeled as true in the test set, but it also predicted that 54 planets in total are potentially. I choose to rather include all 54 potentially habitable planets rather than just focusing on the 14, even though habitability_label = 1 is based on the range from koi_teq and koi_insol (remember, it is just based on temperature and isolation). I don't want to rule out those who have been labeled as potentially habitable planets in fear of missing out on their potential. Here is the list of all 54 potentially habitable exoplanets by model probability:



1	kepoi_nam	kepler_nam	koi_pra	koi_te	koi_insc	koi_perio	koi_duratio	koi_dept	koi_model_sn	habitability_pro
2	K00854.01	Kepler-705 b	1,94	233	0,69	56,0560754	4,685	1626	27,4	1
3	K00250.04	Kepler-26 e	2,13	276	1,37	46,82767725	1,9033	1537,7	32,4	1
4	K00438.02	Kepler-155 c	1,87	271	1,28	52,6615266	3,1592	1078,1	38,4	1
5	K03282.01	Kepler-1455 b	1,75	271	1,28	49,2768448	3,787	1132,9	17,8	1
6	K00701.03	Kepler-62 e	1,72	269	1,24	122,3858681	7,123	711,1	46,3	1
7	K01298.02	Kepler-283 c	1,87	240	0,78	92,7495777	5,506	1105,1	18	1
8	K00812.03	Kepler-235 e	1,83	273	1,32	46,1842039	4,758	1394,7	26	1
9	K02626.01	Kepler-1652 b	1,58	242	0,81	38,0970717	4,307	913,3	17	0,99
10	K00701.04	Kepler-62 f	1,43	207	0,44	267,282521	7,788	471,9	19,3	0,99
11	K01876.01	Kepler-991 b	2,45	260	1,08	82,5341188	4,8556	1895,4	40,1	0,99
12	K02686.01	Kepler-1868 b	3,43	215	0,5	211,032819	7,119	1953,4	69,3	0,99
13	K02210.02	Kepler-1143 c	3,23	234	0,71	210,631486	8,419	1657,8	25,9	0,99
14	K00463.01	Kepler-560 b	1,55	267	1,21	18,47762694	1,94	2604,6	75,7	0,99
15	K01422.05	Kepler-296 e	1,06	248	0,89	34,1420506	2,945	788	13,3	0,99
16	K02078.02	Kepler-1086 c	2,88	211	0,47	161,515617	6,548	2215,3	25,2	0,99
17	K00448.02	Kepler-159 c	2,05	275	1,35	43,5859419	4,6814	1708,7	38,9	0,99
18	K04745.01	Kepler-443 b	2,32	233	0,7	177,668663	9,298	901,4	12,7	0,99
19	K01596.02	Kepler-309 c	1,87	204	0,41	105,3582299	3,81	1210,2	22,6	0,99
20	K02762.01	Kepler-1341 b	2,51	241	0,8	132,9955503	7,436	1275,4	29,3	0,99
21	K02834.01	Kepler-1362 b	2,35	248	0,89	136,205112	5,568	1209,3	20	0,99
22	K04103.01	Kepler-1552 b	2,41	268	1,21	184,77271	9,568	951,4	21	0,99
23	K00518.03	Kepler-174 d	2,29	212	0,48	247,354914	8,677	1291,8	49	0,99
24	K03010.01	Kepler-1410 b	1,39	244	0,84	60,8660974	4,397	714,3	15,4	0,98
25	K04054.01	Kepler-1701 b	2,21	273	1,31	169,135056	8,931	640,7	24,5	0,98
26	K07016.01	Kepler-452 b	1,09	220	0,56	384,847556	9,969	189,9	12,3	0,98
27	K01830.02	Kepler-967 c	3,56	258	1,05	198,7106251	8,638	2094,8	62,1	0,98
28	K01422.04	Kepler-296 f	1,18	202	0,39	63,3354719	3,651	983	13	0,98
29	K04087.01	Kepler-440 b	1,61	229	0,65	101,1107014	8,029	878,1	24,1	0,98
30	K05416.01	Kepler-1628 b	6,28	240	0,78	76,377855	4,809	11004	38,6	0,98
31	K00881.02	Kepler-712 c	4,53	236	0,73	226,8904775	7,558	3136,4	46,8	0,98
32	K02703.01	Kepler-1318 b	2,72	202	0,39	213,258526	7,768	1992,3	36,1	0,97
33	K01430.03	Kepler-298 d	2,63	277	1,4	77,4740991	4,168	1543,6	27,5	0,97
34	K01361.01	Kepler-61 b	2,19	273	1,31	59,8780258	4,8027	1419,1	51,8	0,97
35	K04742.01	Kepler-442 b	1,3	241	0,79	112,303136	5,869	502,1	13,1	0,96
36	K00087.01	Kepler-22 b	2,34	257	1,03	289,864067	7,565	529,1	57,4	0,96
37	K04385.02	Kepler-1600 b	2,96	209	0,45	386,370512	11,007	1336,4	13,1	0,96
38	K05706.01	Kepler-1636 b	3,2	248	0,89	425,483539	17,856	996,6	19,6	0,96
39	K02102.01	Kepler-1097 b	2,95	267	1,2	187,746606	5,885	1583	20,2	0,96
40	K01871.01	Kepler-1996 c	2,81	276	1,38	92,7297248	6,83	1257,5	43,3	0,96
41	K03497.01	Kepler-1512 b	0,8	276	1,38	20,35971899	2,1521	345,7	26,8	0,95
42	K04036.01	Kepler-1544 b	1,69	241	0,8	168,81133	6,582	614,4	15,3	0,94
43	K02020.01	Kepler-1058 b	2,11	239	0,77	110,9643706	12,831	1283,2	42,1	0,93
44	K00433.02	Kepler-553 c	10,99	224	0,59	328,240201	12,2726	13889	287,6	0,93
45	K04016.01	Kepler-1540 b	3,14	266	1,18	125,4132302	2,3241	1345	22,6	0,91
46	K02757.01	Kepler-1690 b	2,56	267	1,2	234,635652	6,965	868,5	26,7	0,9
47	K05622.01	Kepler-1635 b	3,24	200	0,37	469,61309	13,709	1234,7	18,2	0,9
48	K03138.01	Kepler-1649 b	0,49	211	0,47	8,68910764	1,1627	1679	9,6	0,89
49	K04051.01	Kepler-1545 b	2,75	274	1,34	163,692336	6,853	1138	21	0,88
50	K00179.02	Kepler-458 b	5,8	264	1,15	572,376632	24,42	1743,4	78,6	0,86
51	K02418.01	Kepler-1229 b	1,68	196	0,35	86,829519	7,464	751	15,1	0,86
52	K03663.01	Kepler-86 b	8,98	264	1,15	282,5253558	10,7965	9746,4	696	0,85
53	K04550.01	Kepler-1653 b	1,84	271	1,28	140,251943	6,576	567,7	11,9	0,85
54	K05581.01	Kepler-1634 b	4,27	282	1,49	374,878133	10,556	914,6	21,9	0,76
55	K00375.01	Kepler-1704 b	10,81	223	0,58	988,8811177	5,9504	5060	136,4	0,75

Figure 50. List of potential habitable planets from the training result of Random Forest

10 Discussion

This project demonstrates that machine learning models may play an important role in helping astronomers classify exoplanet candidates and identify those worth further investigation. With Kepler's large data detections, manually distinguishing confirmed planets from false positives is no longer scalable. In this project, both the Logistic Regression and the Random Forest models surprisingly achieved very good performance, with Random Forest showing a bit better result. This could suggest that columns like KOI score, signal-to-noise ratio, and planetary radius hold meaningful predictions for determining confirmation status.

In addition, the project also explored potential habitability planets using criteria based on only equilibrium temperature and stellar isolation. The model also performed well and gave us 54 candidates with physical traits similar to our own Earth (Earth-like conditions).

Yes, it was promising, but it is important to remain cautious. The predictions are only good as the assumption used by defining habitability solely on temperature and stellar isolation. They do not account for essential but missing variables such as atmosphere, magnetic field, or surface composition. The idea for this did not only come from exploring the dataset. It is also inspired by Ravi Kumar Kopparapu and colleagues who refined the boundaries of the habitable zone using climate models. They included stellar luminosity and atmospheric conditions. They tried to show a more accurate estimate of habitable zones around different types of stars (Kopparapu 2013; Kane S. H.-G., 2016).

A critical take is that though ML helps make sense of complex datasets, it simplifies the physical reality of exoplanets. A model does not "know" what makes a planet habitable, it only learns from the labels and numbers we define. We must acknowledge these limitations when interacting with the results.

11 Conclusion

My conclusion to this project is that through data cleaning, exploratory data analysis, and machine learning, I made a reasonably accurate model for classifying confirmed exoplanets and assessing potential habitability based on measurable parameters. The models performed very well and quite accurately, showing that the Kepler dataset contains enough structure to allow reliable predictions. At least within the scope of the columns available.

That said, the project was absolutely not without its challenges. Dealing with missing values, inconsistent data formats, and some unclear definitions of the columns required patience and careful judgment. Not to mention the coding part when making a predicting model, and even the dashboard for visualization. Reading the definition for each column was also quite challenging. I spent a lot of time understanding the vast specter of the Kepler Exoplanet Archive.

The criteria for defining habitability were also limited to just two columns/parameters. While scientifically justifiable, this is admittedly a simplified view of what makes a planet habitable. I should have gone deeper using a more multi-factor approach, but my expertise had its limitations. Maybe if I had enough time, I could have made a more accurate model for finding habitable planets.

Reflecting on this absolutely exciting journey, I have learned how valuable it is to deeply explore a dataset before modeling. Taking the time to understand each column, rather than rushing into machine learning, provided me with a context that shaped every decision I made afterward. I have also gained a greater appreciation for the bridge between data science and space science. How rows in a spreadsheet can represent entire worlds.

References

BIBLIOGRAPHY

Archive, N. E. (2023, December 14). *Planetary Systems (PS) API Table Column Definitions*. Retrieved from NASA Exoplanet Archive: https://exoplanetarchive.ipac.caltech.edu/docs/API_PS_columns.html

AstroPhil. (2022, Mai 13). *Youtube*. Retrieved from Impact Parameter: Inclination Of A Transiting Exoplanet: https://www.youtube.com/watch?v=JMjEFgnifXE

Astrophysics, C. f. (2021, October 4). *The Atmosphere of an Ultra-Hot Jupiter Exoplanet*. Retrieved from Center for Astrophysics: https://www.cfa.harvard.edu/news/atmosphere-ultra-hot-jupiter-exoplanet

Boyce-Astro. (n.d.). *What is BJD?* Retrieved from Boyce-Astro: https://boyce-astro.org/wp-content/uploads/BRIEF-Video-Lesson-TIME-What-is-BJD.pdf

Creevey, O. H. (2021). The Gaia-ESO Survey: Empirical calibration of stellar surface gravity using asteroseismology. *Astronomy & Astrophysics*, A161. Retrieved from https://www.aanda.org/articles/aa/full_html/2021/07/aa40935-21/aa40935-21.html

Eastman, J. S. (2010). *Explanation of BJD*. Retrieved from Astronomy Utilities - Ohio State University: https://astroutils.astronomy.osu.edu/time/bjd_explanation.html

Gary, J. (2010, March 10). *Lecture 14 – Planetary Temperatures*. Retrieved from NJIT Department of Physics – Physics 320: https://web.njit.edu/~gary/320/Lecture14.html

Institute, N. E. (2024, December 11). *NASA Exoplanet Archive*. Retrieved from NASA Exoplanet Archive: https://exoplanetarchive.ipac.caltech.edu/index.html

Jenkins, J. e. (2019). *Kepler Data Processing Handbook: Presearch Data Conditioning Pipeline Module.* Moffett Field, California: NASA Ames Research Center. Retrieved from https://ntrs.nasa.gov/api/citations/20190002637/downloads/20190002637.pdf

Jet Propulsion Laboratory, California Institute of Technology. (2015, January 6). *NASA's Kepler Marks 1,000th Exoplanet Discovery, Uncovers More Small Worlds in Habitable Zones*. Retrieved from NASA Jet Propulsion Laboratory: https://www.jpl.nasa.gov/news/nasas-kepler-marks-1000th-exoplanet-discovery-uncovers-more-small-worlds-in-habitable-zones/

Kane, S. H. (2016). *A Catalog of Kepler Habitable Zone Exoplanet Candidates*. Retrieved from The Astrophysical Journal: https://arxiv.org/pdf/1608.00620.pdf

Kane, S. H.-G. (2016, August 2). *A Catalog of Kepler Habitable Zone Exoplanet Candidates*. Retrieved from arXiv: https://arxiv.org/pdf/1608.00620

Kopparapu, R. K. (2013, January 29). *Habitable Zones Around Main-Sequence Stars*. Retrieved from arXiv.org: https://arxiv.org/abs/1301.6674

Mandel, K. &. (2002, December). Analytic Light Curves for Planetary Transit Searches. *The Astrophysical Journal, 580*(2), L171-L175. doi: 10.1086/345520

Mann, A. (2013, February 20). *Tiny Mercury-Sized Exoplanet Spotted in Distant Solar System*. Retrieved from Wired: https://www.wired.com/2013/02/mercury-sized-exoplanet

NASA. (2021, October 1). *Sun Fact Sheet*. Retrieved from NASA National Space Science Data Center (NSSDC): https://nssdc.gsfc.nasa.gov/planetary/factsheet/sunfact.html

NASA. (2023, December 12). *Kepler Mission - In Depth*. Retrieved from NASA Science: https://science.nasa.gov/mission/kepler/in-depth/#key-facts

NASA Science. (2023, October 5). *What is the Habitable Zone?* Retrieved from NASA Science - Exoplanets: https://science.nasa.gov/resource/what-is-the-habitable-zone/

NASA Science. (2024, April 11). *What's a Transit?* Retrieved from NASA Science – Exoplanets: https://science.nasa.gov/exoplanets/whats-a-transit/

NASA, E. S. (2022, May 10). *Kepler Mission*. Retrieved from NASA Exoplanet Archive - NASA Exoplanet Science Institute: https://exoplanetarchive.ipac.caltech.edu/docs/KeplerMission.html

NASA, S. (2023, October 25). *Transit Method*. Retrieved from Transit Method: https://science.nasa.gov/mission/roman-space-telescope/transit-method/

Petigura, E. H. (2013). Prevalence of Earth-size planets orbiting Sun-like stars. *Proceedings of the National Academy of Sciences of the United States of America, 110*(48), 19273–19278. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3845182/

Quintana, E. B. (2014). *An Earth-Sized Planet in the Habitable Zone of a Cool Star*. Retrieved from Science: https://arxiv.org/abs/1404.5667

ScienceDirect. (n.d.). *Kurtosis – Neuroscience*. Retrieved from ScienceDirect: https://www.sciencedirect.com/topics/neuroscience/kurtosis

Shallue, C. a. (2017, December 13). *Identifying Exoplanets with Deep Learning*. Retrieved from arXiv.org: https://arxiv.org/abs/1712.05044

Southworth, J. (2023, October 12). *Stellar and planetary physical constants used in TEPCat*. Retrieved from Transiting Extrasolar Planet Catalogue (TEPCat): https://www.astro.keele.ac.uk/jkt/tepcat/html-constants.html

Team, E. A. (2025, January 9). *How to explain the ROC curve and ROC AUC score?* Retrieved from Evidently AI: https://www.evidentlyai.com/classification-metrics/explain-roc-curve

Twicken, J. e. (2019). *Kepler Data Processing Handbook: Data Validation.* Moffett Field, California: NASA Ames Research Center. Retrieved from https://ntrs.nasa.gov/api/citations/20190002432/downloads/20190002432.pdf

University of Nebraska-Lincoln. (2017, March). *Transit Depths and Light Curves*. Retrieved from Astronomy Education at UNL: https://astro.unl.edu/newRTs/Transits/background/Transit1.html

Wikipedia. (2024, December 10). *Earth radius*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Earth_radius

Wikipedia. (2024, December 20). *Kelvin*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Kelvin

Wikipedia. (2024, December 10). *Kepler space telescope*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Kepler_space_telescope

Wikipedia. (2024, May 4). *Methods of detecting exoplanets*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Methods_of_detecting_exoplanets

Wikipedia. (2024, December 15). *Planetary equilibrium temperature*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Planetary_equilibrium_temperature

Wikipedia. (2024, December 10). *Stellar classification*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Stellar_classification

Yang, K. X. (2014, April 18). *Habitability of Kepler-186f*. Retrieved from Beyond Earthly Skies: https://beyondearthlyskies.blogspot.com/2014/04/habitability-of-kepler-186f.html

TABLE OF FIGURES: