july 10th, 2024

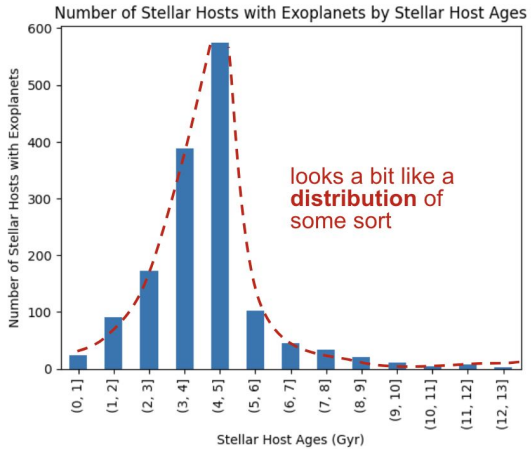
exoplanet classification research

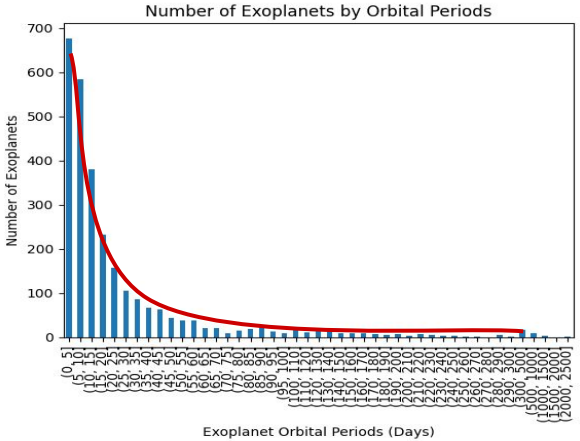
what i worked on

- inspecting some of the graphs created last week (importantly, the stellar age, the orbital period graphs) for further details.
  - e.g. trying different scales for different axes to get a closer look at graphs with logarithmic trends
- building **decision tree ML model** to better see “important” features when studying HZ planets, and analyzing this data.

two weeks ago…

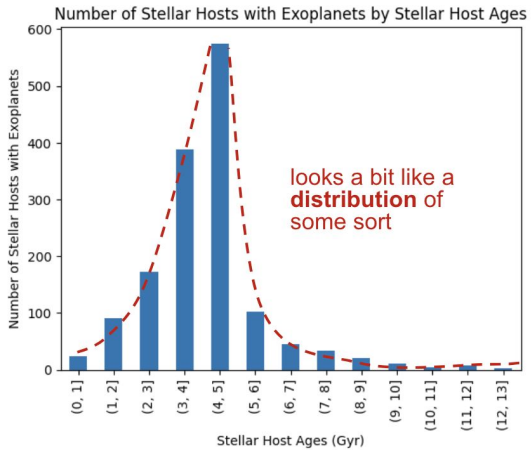
i found some really interesting patterns that appeared when comparing certain characteristics and exoplanet frequency.

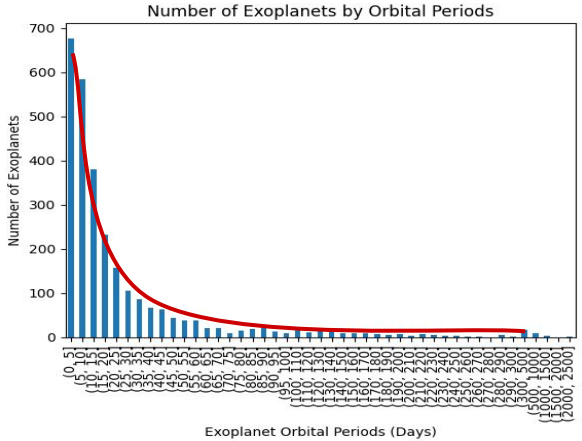




two weeks ago…

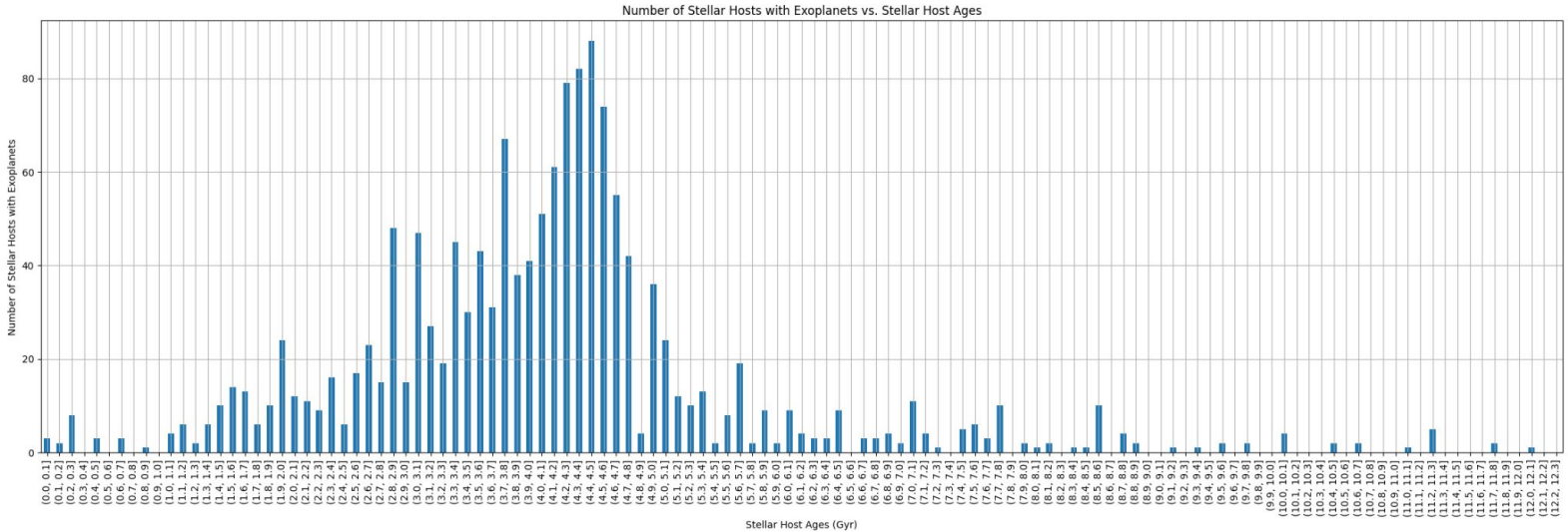
the problem: these graphs were all pretty small, as i meant for them to be samples. stellar hosts might seem like it has a pattern on the surface, but when we spread it out things might look different!!



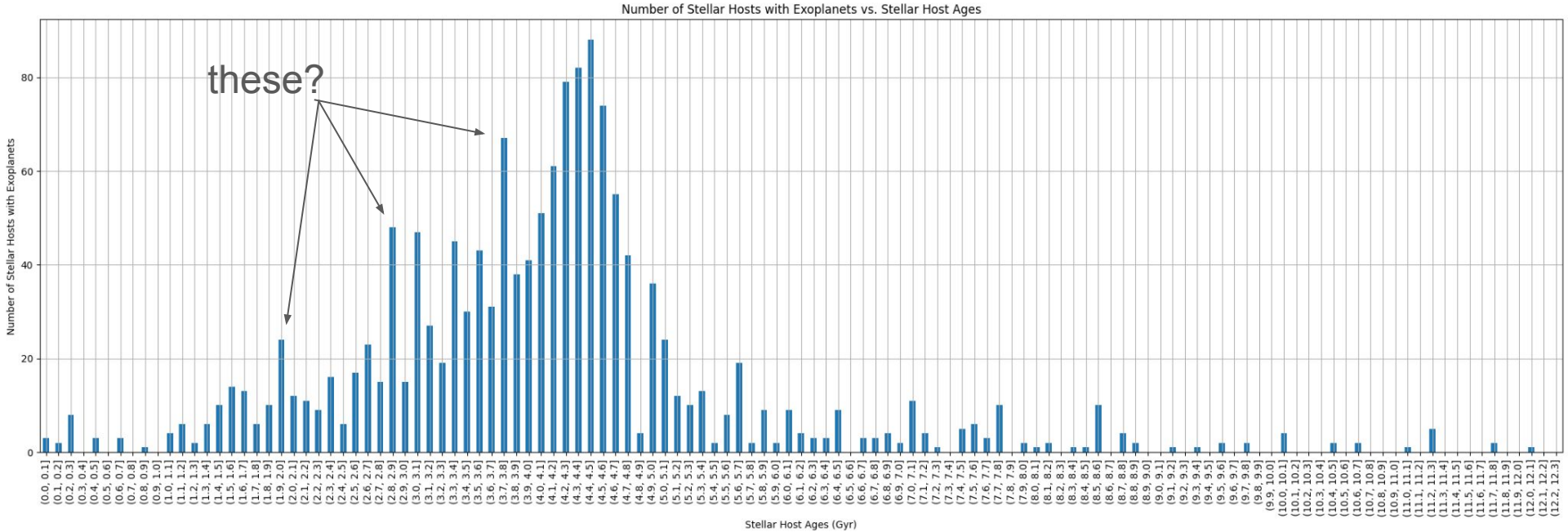


working on the stellar age graph

- so i made it bigger.
- **MUCH BIGGER.**
  made the scale for the x-axis a lot smaller, which results in this:

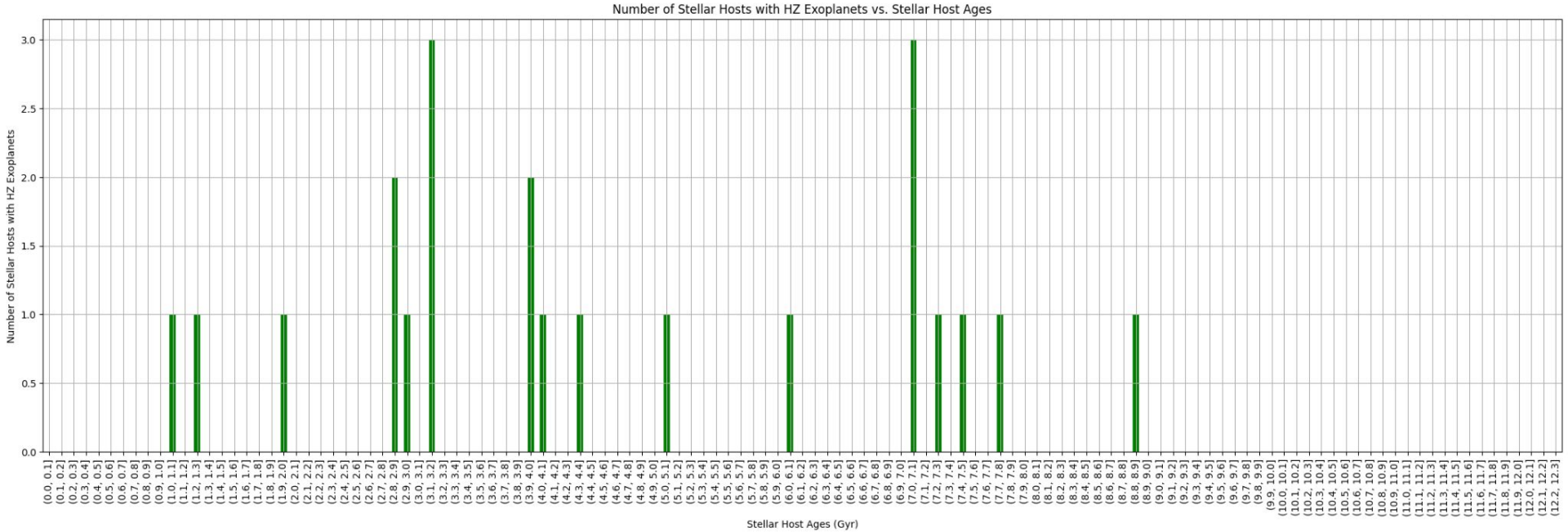


a closer look (# of exoplanets vs. stellar host ages)



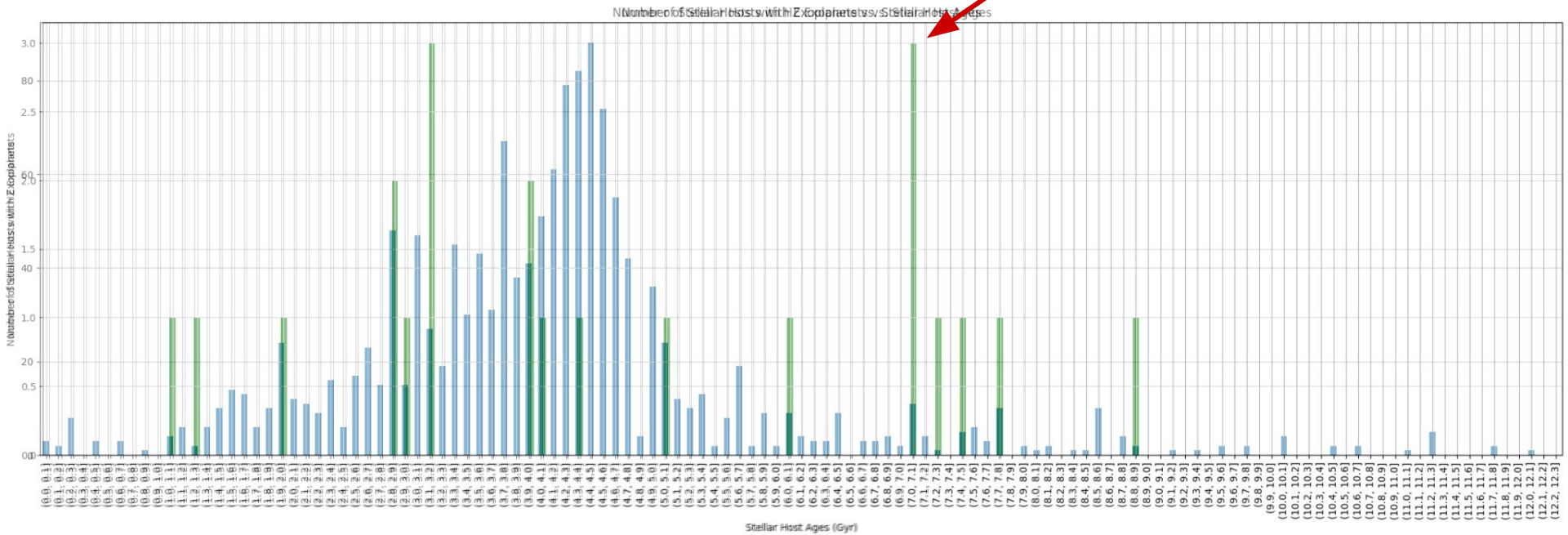
scale: increments of 0.1 Gyr

a closer look (# of HZ exoplanets vs. stellar host ages)



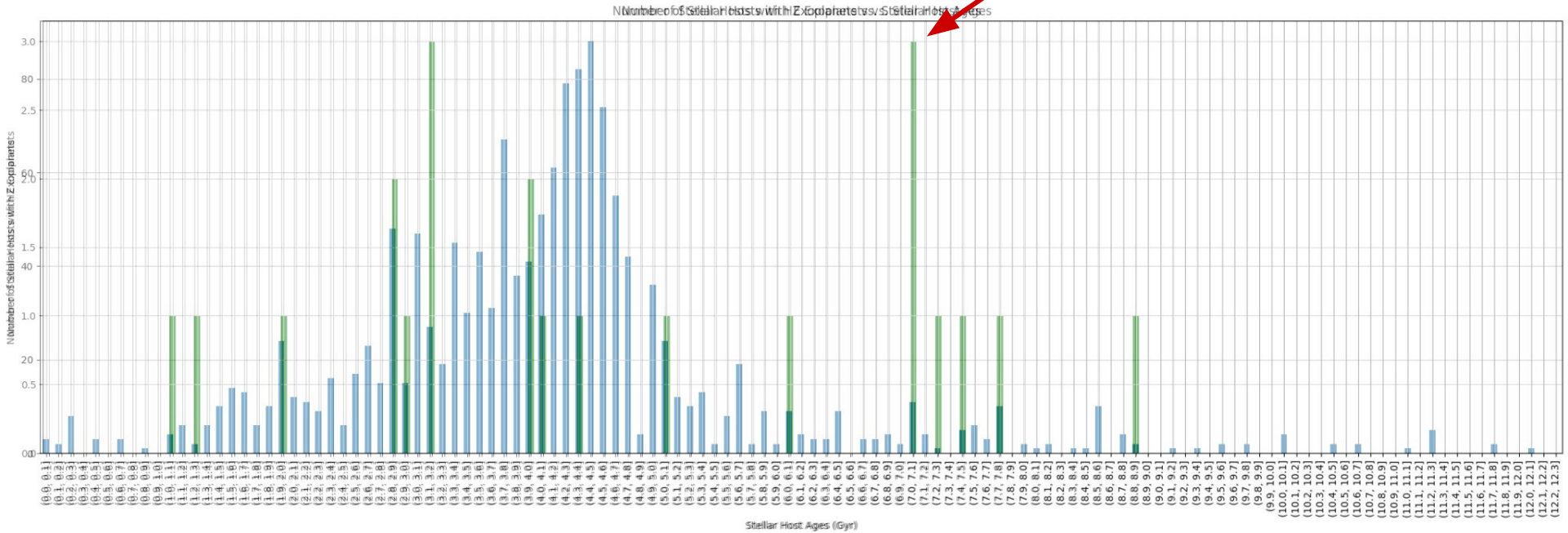
scale: increments of 0.1 Gyr

a closer look (overlayed)



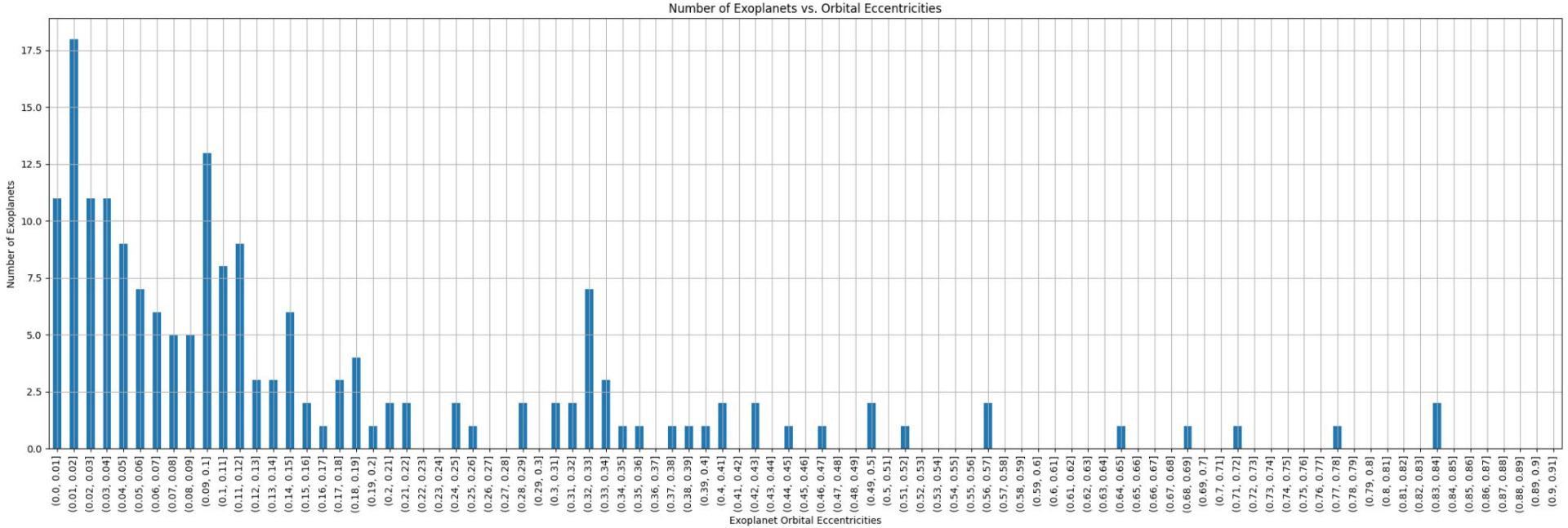
scale: increments of 0.1 Gyr

a closer look (overlayed)

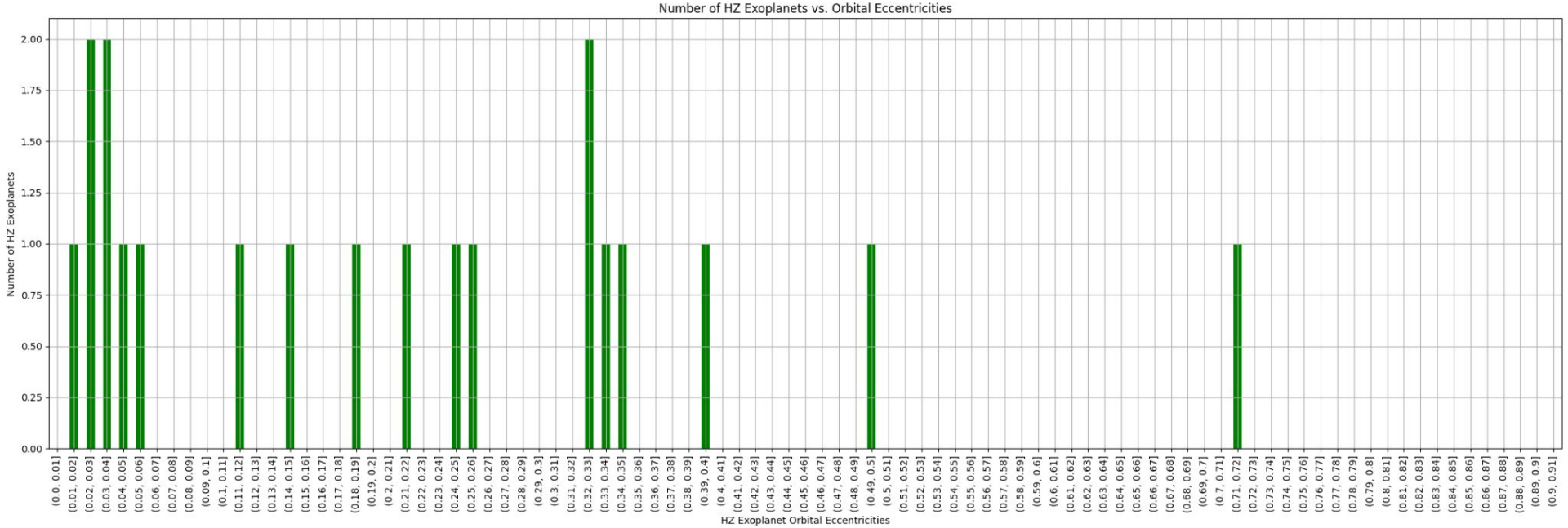


observations: stellar age definitely seems to affect the likelihood of habitable exoplanets, as the curve of the HZ graph does not follow the general trend of the general graph.

i did the same for a few others



i did the same for a few others



observations: orbital eccentricity also seems to have a little importance to the freq. of HZ exoplanets as it also doesn't follow the general pattern of the general graph.

at this point…

you can start to understand that there are a lot of really cool relationships…

at this point…

you can start to understand that there are a lot of really cool relationships…

BUT ALSO A LOT OF DIFFERENT POSSIBLE GRAPHS TO MAKE.

at this point…

you can start to understand that there are a lot of really cool relationships…

BUT ALSO A LOT OF DIFFERENT POSSIBLE GRAPHS TO MAKE.

so my new objective … was to make a **decision tree classifier** to help pinpoint important features to study.

habitable zone determinations

data from [kepler confirmed exoplanet archive]

Kepler Mission Counts

Confirmed Planets Discovered by Kepler [2]	2774
Candidates and Confirmed in Habitable Zone [1, 3] (180 K < Equilibrium (T) < 310 K) or (0.25 < Insolation (Earth flux) < 2.2)	361

two ways to identify habitable zone exoplanets (according to NASA exopl. archive):

TEMPERATURE

INSOLATION

any one of them being met is enough (?)

decision tree classifier – data labeling

- data source: [Confirmed Planets Discovered By Kepler](#) (2773 items)
- labeling data as below (hz criteria according to NASA exopl. archive)

```
[13] exoplanets_data.loc[((~np.isnan(exoplanets_data['pl_eqt'])) & (exoplanets_data['pl_eqt'] > 180) & (exoplanets_data['pl_eqt'] < 310)), 'hz_label_by_eqt'] = 1
     exoplanets_data.loc[((~np.isnan(exoplanets_data['pl_eqt'])) & (exoplanets_data['pl_eqt'] <= 180) | (exoplanets_data['pl_eqt'] >= 310)), 'hz_label_by_eqt'] = 0
     exoplanets_data['hz_label_by_eqt'].value_counts()

     hz_label_by_eqt
     0.0    193
     1.0     20
     Name: count, dtype: int64
```

equilibrium temperature fiddling – if within range, set hz_label_by_eqt 1, otherwise 0. there are **20** in the correct zone

```
[14] exoplanets_data.loc[((~np.isnan(exoplanets_data['pl_insol'])) & (exoplanets_data['pl_insol'] > 0.25) & (exoplanets_data['pl_insol'] < 2.2)), 'hz_label_by_insol'] =
     exoplanets_data.loc[((~np.isnan(exoplanets_data['pl_insol'])) & (exoplanets_data['pl_insol'] <= 0.25) | (exoplanets_data['pl_insol'] >= 2.2)), 'hz_label_by_insol']
     exoplanets_data['hz_label_by_insol'].value_counts()

     hz_label_by_insol
     0.0    135
     1.0     22
     Name: count, dtype: int64
```

earth flux insolation fiddling – if within range, set hz_label_by_insol 1, otherwise 0. there are **22** in the correct zone

```
[15] exoplanets_data.loc[(((~np.isnan(exoplanets_data['hz_label_by_eqt'])) & (exoplanets_data['hz_label_by_eqt'] == 1)) | ((~np.isnan(exoplanets_data['hz_label_by_insol'
     exoplanets_data.loc[(((~np.isnan(exoplanets_data['hz_label_by_eqt'])) & (exoplanets_data['hz_label_by_eqt'] == 0)) & ((~np.isnan(exoplanets_data['hz_label_by_insol'
     exoplanets_data.loc[(((~np.isnan(exoplanets_data['hz_label_by_eqt'])) & (exoplanets_data['hz_label_by_eqt'] == 0)) & (np.isnan(exoplanets_data['hz_label_by_insol'])
     exoplanets_data.loc[((np.isnan(exoplanets_data['hz_label_by_eqt'])) & ((~np.isnan(exoplanets_data['hz_label_by_insol'])) & (exoplanets_data['hz_label_by_insol'] ==
     exoplanets_data['hz_label'].value_counts()

     hz_label
     0.0    213
     1.0     31
     Name: count, dtype: int64
```

both of them together – if EITHER of the above are set to 1, the “big” label - hz_label is set to 1, otherwise 0.

decision tree classifier – data cleaning & preparation

- casting all numerical values to floats + setting empty fields to NaN
- cleaning data:
  - dropping rows without labels,
  - dropping irrelevant data fields (e.g., exoplanet name, stellar host name, etc.)
  - dropping data fields with too many missing values
  - dropping redundant data fields (e.g. exoplanet radius in Jupiter scales - there is an exoplanet radius data field in Earth scale)
  - dropping data fields that are used for labeling (e.g. exoplanet equilibrium temperature, exoplanet insolution flux)
- imputation - filling in missing values with the mean (same thing as the KNN classifier)
- standard scalar – scale features to standardized values (same thing as the KNN classifier)

decision tree classifier – training data stats

using the labels we have created, training the model using **213 negative and 31 positive samples,** which will be **oversampled** later to account for skew. we have **19 features** in the training.

```
[ ]  # all columns except "hz_lable" become features in the data for model training and testing.
     # "hz_label" is the label in the data for model training and testing
     features = training_data.drop(['hz_label'], axis = 1)
     labels = training_data.hz_label
```

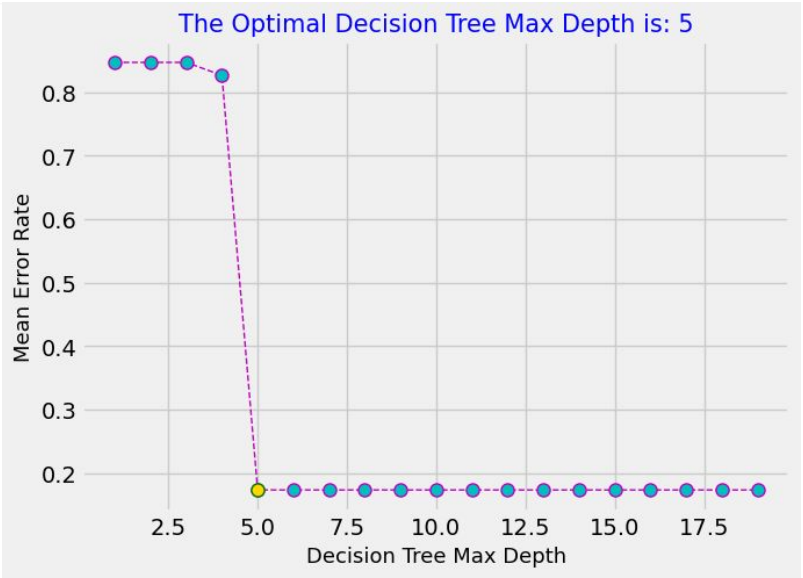
```
[ ]  # we have 213 negative and 31 positive samples in the training data
     # the training data is inbalanced; we'll use oversampling to handle this later on
     labels.value_counts()
```

```
hz_label
0.0    213
1.0     31
Name: count, dtype: int64
```

decision tree classifier – optimal max tree depth

i then found the **optimal max tree depth** by finding the lowest average error rate after training the models (for each max tree depth in the range of [1,19], trained models 100 times, each time with data randomly shuffled and then split with 60% as training data and 40% as testing data), and the optimal max tree depth turned out to be **5.**

```
Searching the optimal decision tree max depth in between 1 ~ 19

    In progress...
    For decision tree with max depth 1, mean prediction error rate = 0.847
    For decision tree with max depth 2, mean prediction error rate = 0.847
    For decision tree with max depth 3, mean prediction error rate = 0.847
    For decision tree with max depth 4, mean prediction error rate = 0.827
    For decision tree with max depth 5, mean prediction error rate = 0.173
    For decision tree with max depth 6, mean prediction error rate = 0.173
    For decision tree with max depth 7, mean prediction error rate = 0.173
    For decision tree with max depth 8, mean prediction error rate = 0.173
    For decision tree with max depth 9, mean prediction error rate = 0.173
    For decision tree with max depth 10, mean prediction error rate = 0.173
    For decision tree with max depth 11, mean prediction error rate = 0.173
    For decision tree with max depth 12, mean prediction error rate = 0.173
    For decision tree with max depth 13, mean prediction error rate = 0.173
    For decision tree with max depth 14, mean prediction error rate = 0.173
    For decision tree with max depth 15, mean prediction error rate = 0.173
    For decision tree with max depth 16, mean prediction error rate = 0.173
    For decision tree with max depth 17, mean prediction error rate = 0.173
    For decision tree with max depth 18, mean prediction error rate = 0.173
    For decision tree with max depth 19, mean prediction error rate = 0.173

    Done! The Optimal Decision Tree Max Depth is: 5
```



decision tree classifier – set up training

```
[ ]  # split data with into training and testing sets
     features_train, features_test, labels_train, labels_test = train_test_split(features,
                                                                                  labels,
                                                                                  test_size=split_test_data_percentage,
                                                                                  random_state=0,
                                                                                  shuffle=True)

     # standadize the scales of features
     features_train_sc = stand_scaler.fit_transform(features_train)
     features_test_sc = stand_scaler.transform(features_test)

     # randomly oversample the positive samples to balance training data
     ros = RandomOverSampler()
     features_train_sc_ros, labels_train_ros = ros.fit_resample(features_train_sc, labels_train)
```

decision tree classifier – training the model!! :D

- using the optimal max tree depth
- using entropy as the criterion to select features and thresholds to split and build the tree hierarchy

```
[ ] # train a Decision Tree classifier with the training data
    decision_tree_classifier = DecisionTreeClassifier(criterion='entropy', max_depth=optimal_max_depth, random_state=0)
    decision_tree_classifier.fit(features_train_sc_ros, labels_train_ros)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=5, random_state=0)
```

decision tree classifier – accuracy, precision, recall, etc.

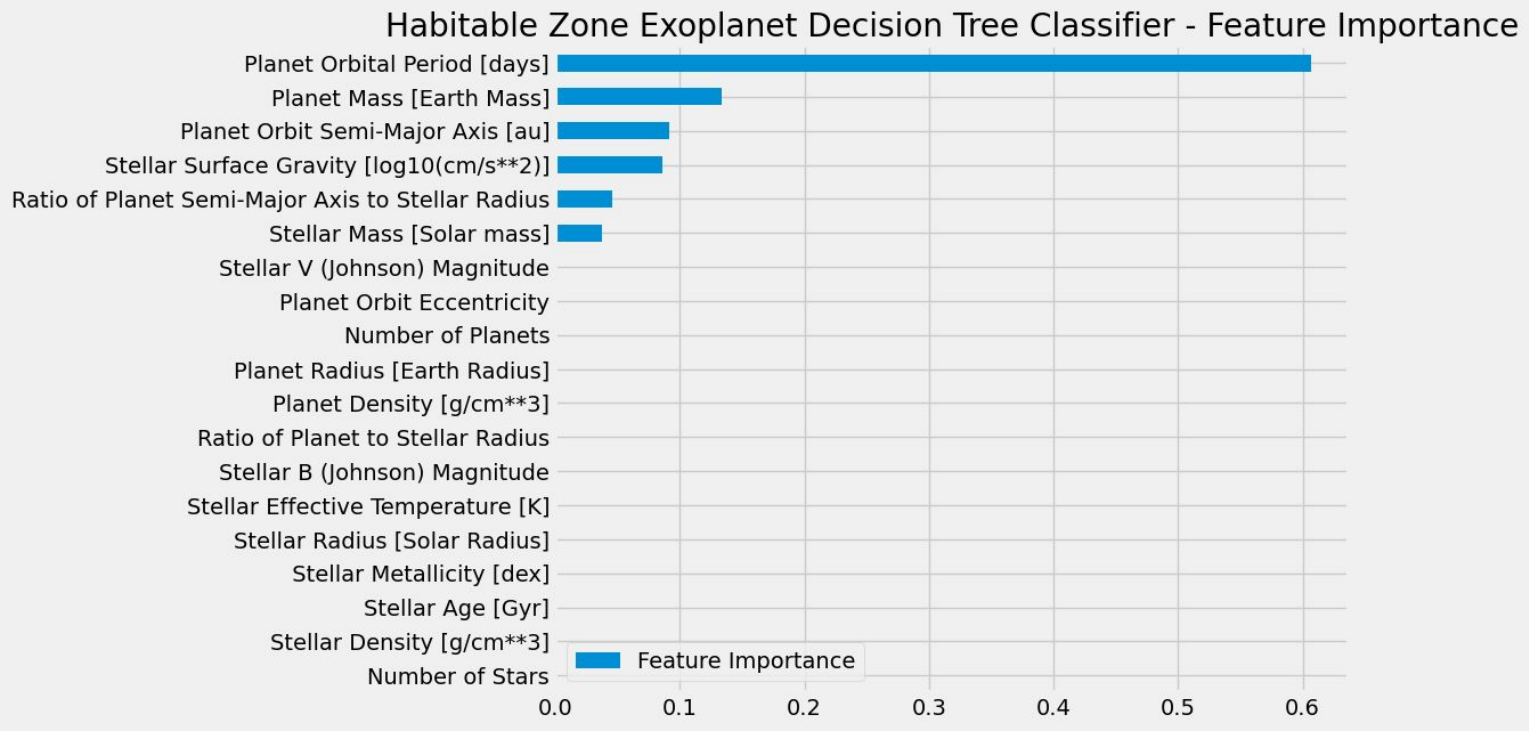
```
[ ]  # calculate accuracy, precision, recall, and F-1 scores for the Decision Tree classifier
     print("Decision Tree Classifier Accuracy: ", accuracy_score(labels_test, labels_pred))
     print()
     print("Decision Tree Classification Report :\n",(classification_report(labels_test,labels_pred)))
```

```
Decision Tree Classifier Accuracy:  0.8877551020408163

Decision Tree Classification Report :
              precision    recall  f1-score   support

         0.0       0.91      0.96      0.94        83
         1.0       0.70      0.47      0.56        15

    accuracy                           0.89        98
   macro avg       0.80      0.72      0.75        98
weighted avg       0.88      0.89      0.88        98
```

compare with: **KNN HZ exoplanet classifier**

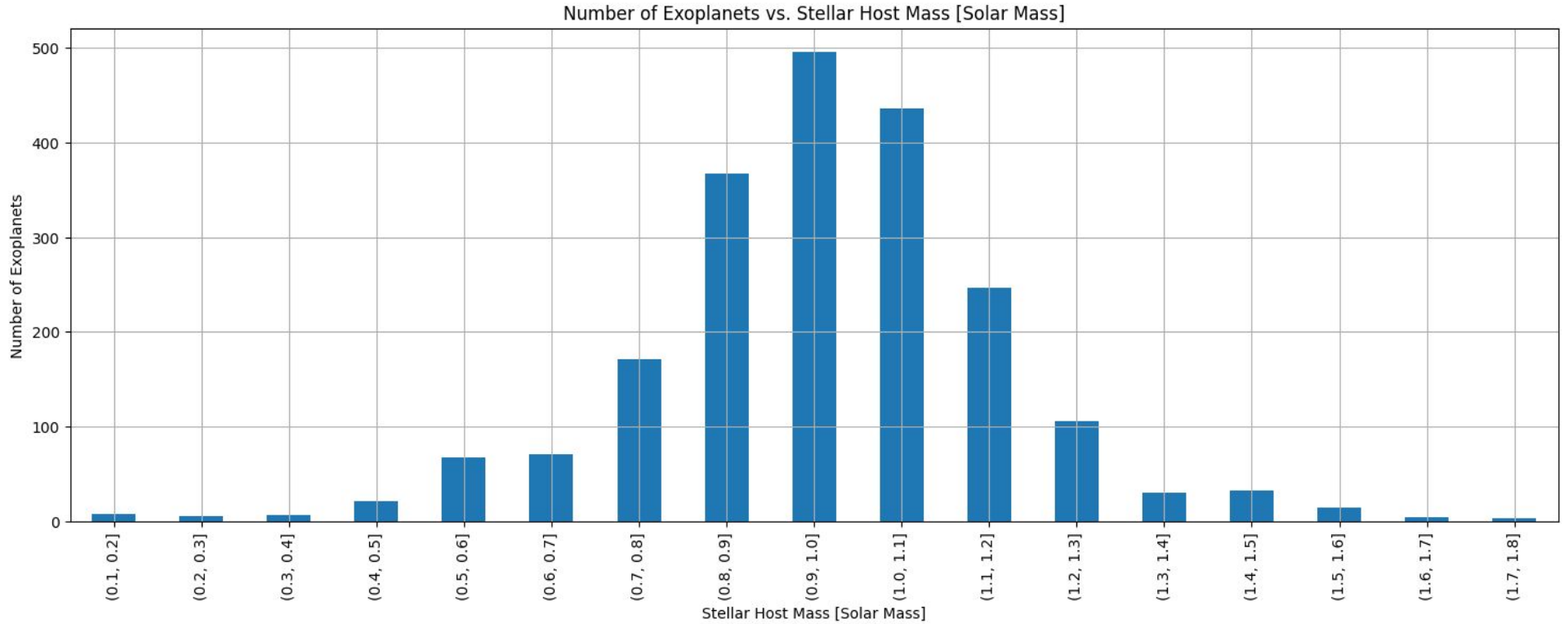
```
KNN Classifier Accuracy:  0.8648648648648649

KNN Classifier Classification Report :
              precision    recall  f1-score   support

         0.0       0.89      0.95      0.92        62
         1.0       0.62      0.42      0.50        12

    accuracy                           0.86        74
   macro avg       0.76      0.68      0.71        74
weighted avg       0.85      0.86      0.85        74
```

decision tree classifier – feature importance



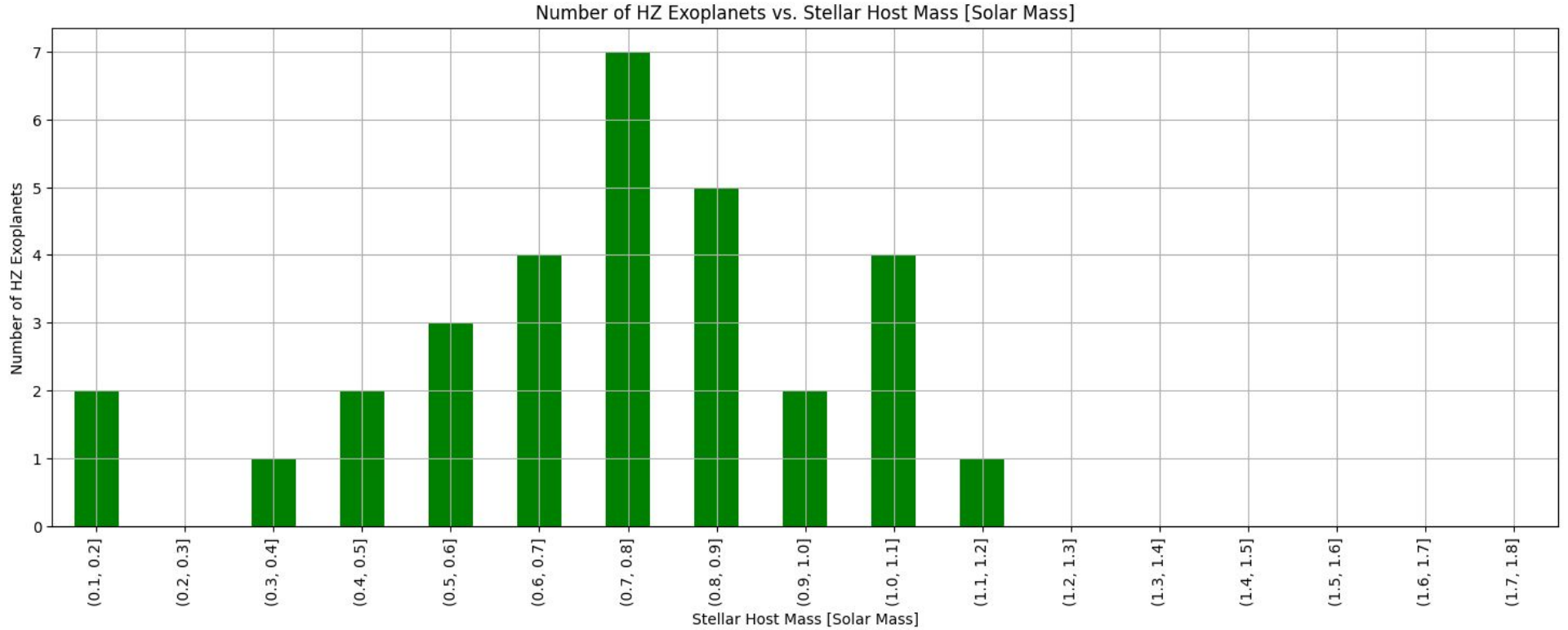
making spicier graphs of the important identified feat.

6. stellar mass – number of exoplanets



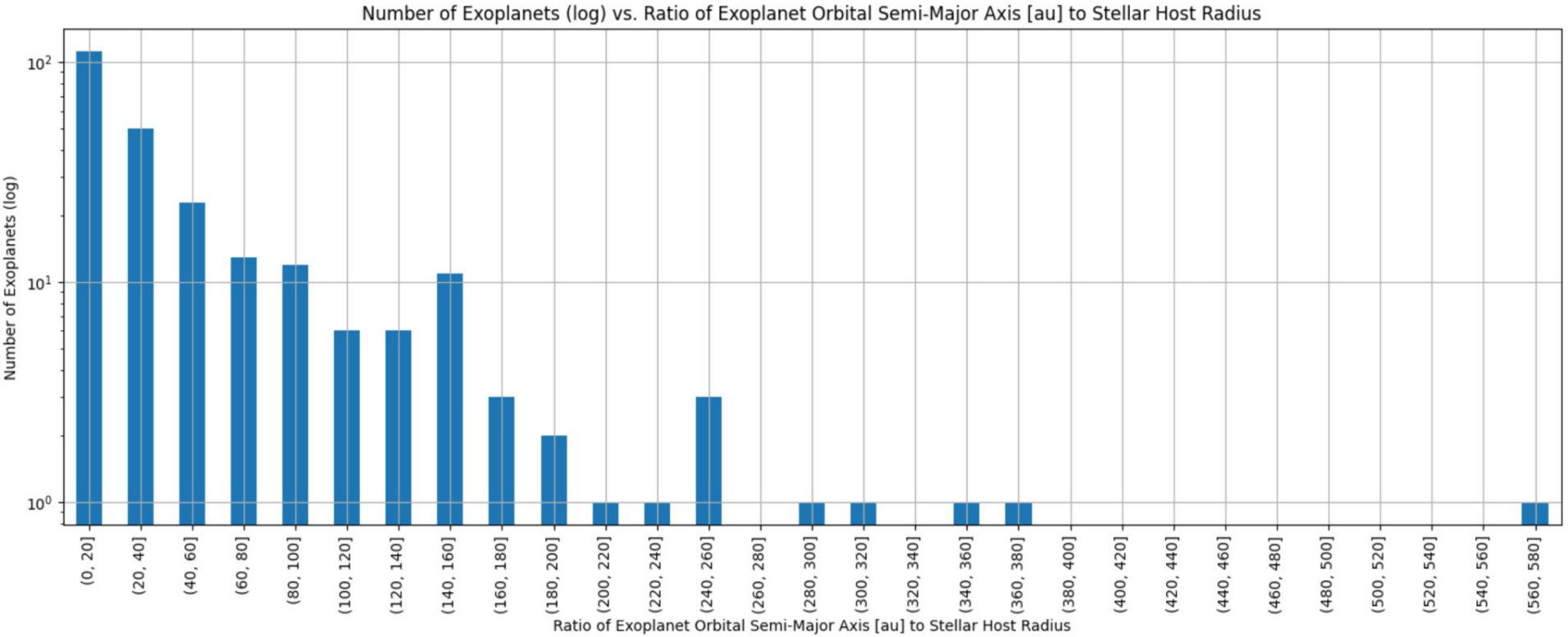
making spicier graphs of the important identified feat.

6. stellar mass – number of **HZ** exoplanets



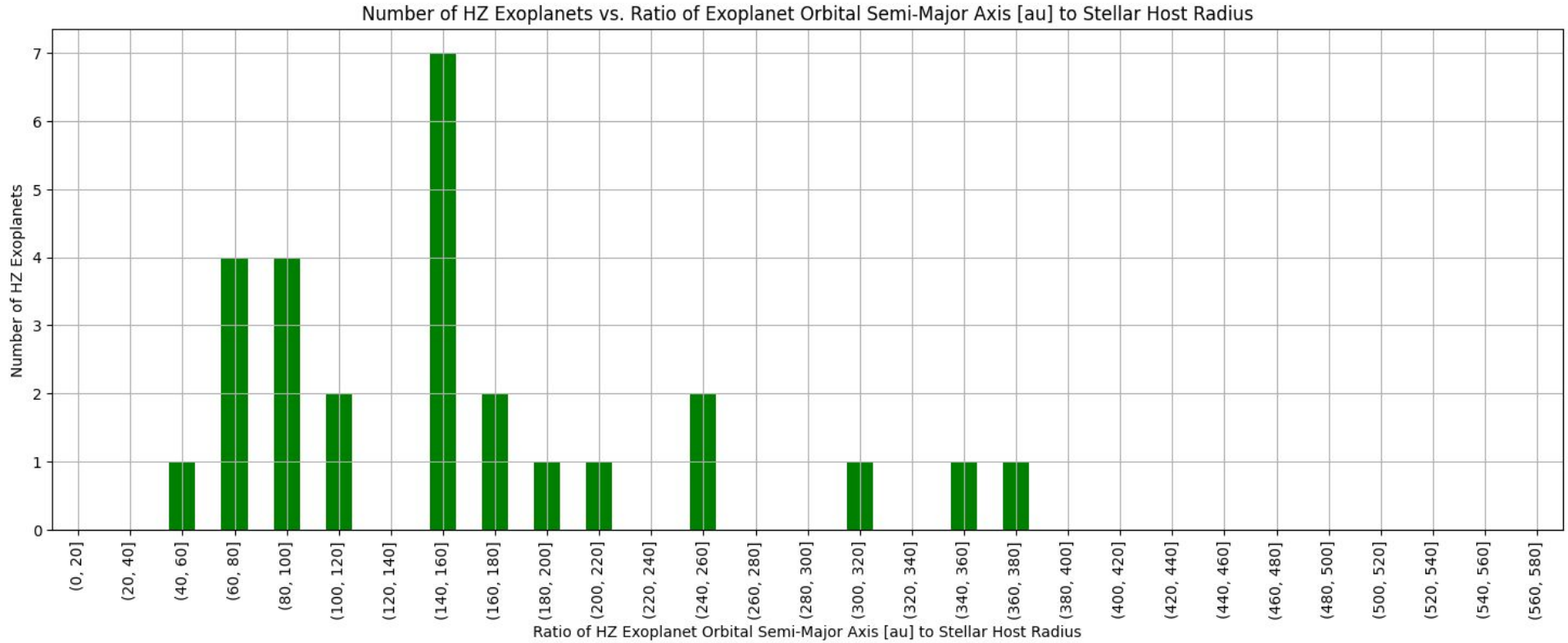
making spicier graphs of the important identified feat.

5. ratio of planet semi-major axis to stellar radius – number of exoplanets **(log scale)**



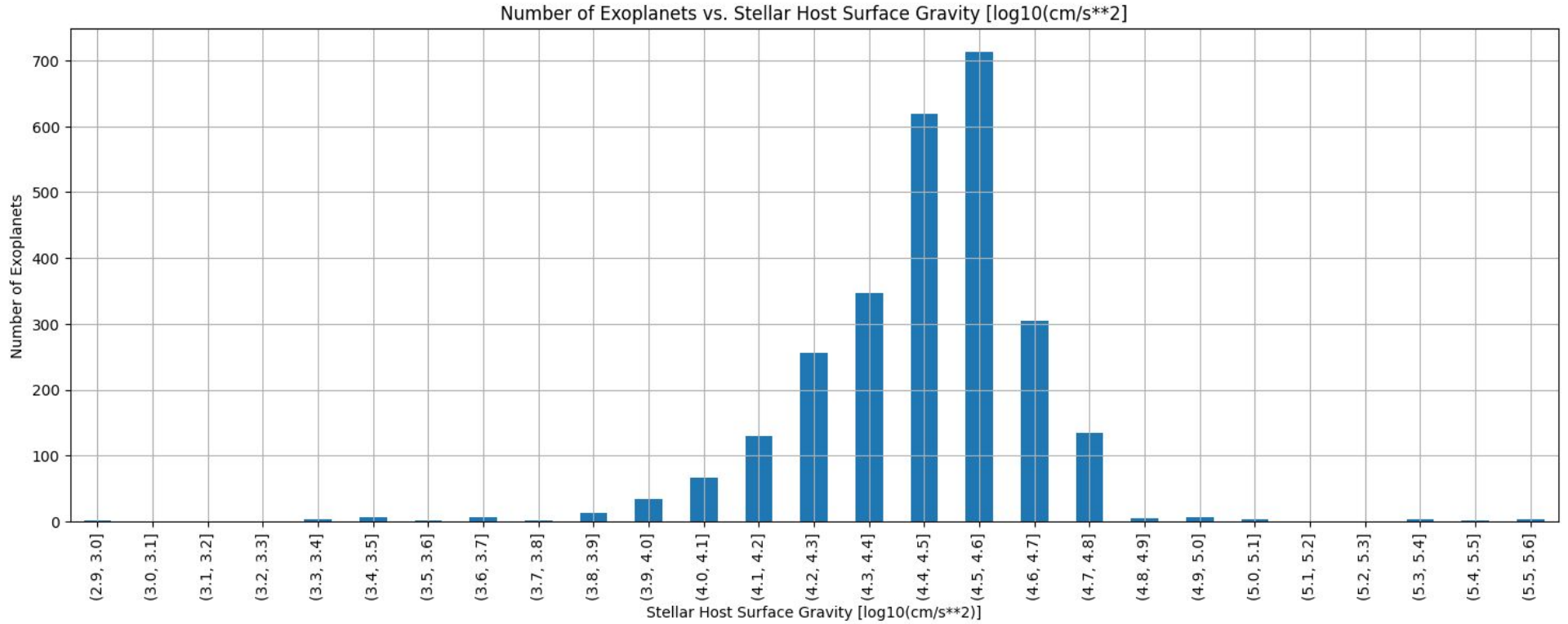
making spicier graphs of the important identified feat.

5. ratio of planet semi-major axis to stellar radius – number of **HZ** exoplanets



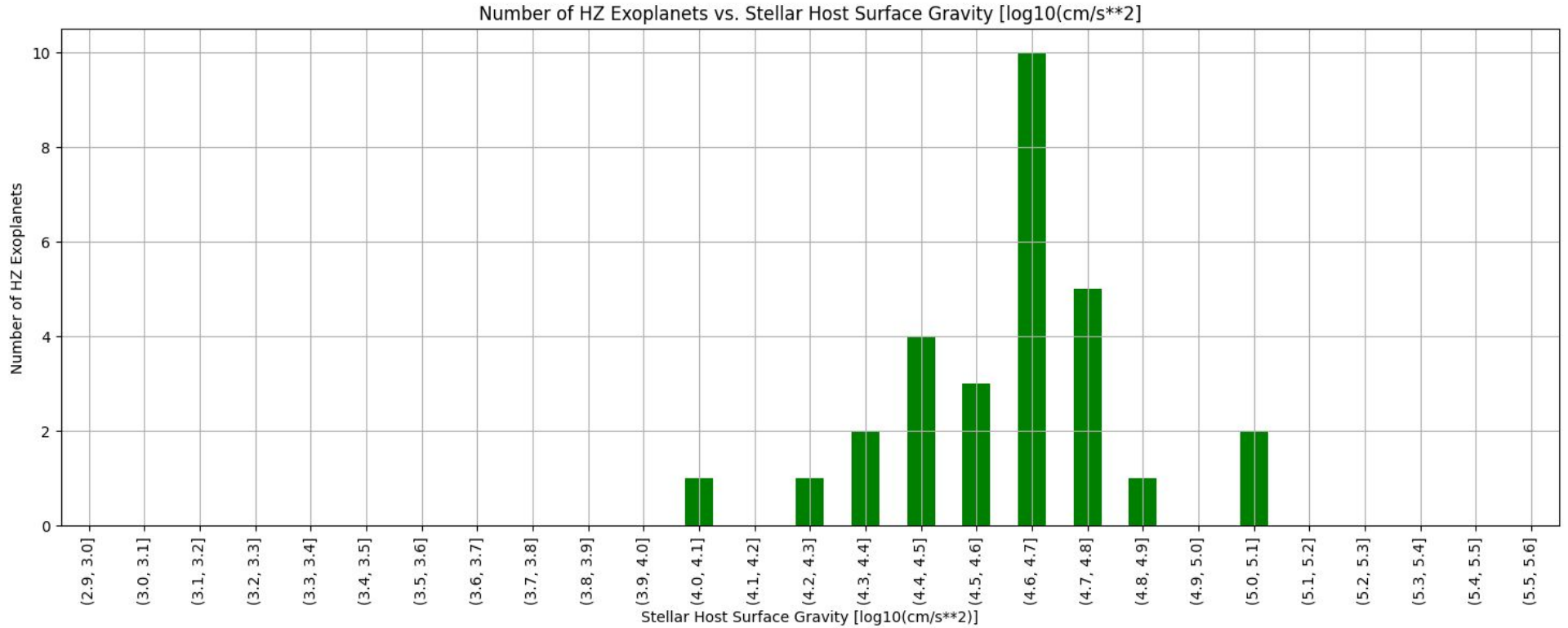
making spicier graphs of the important identified feat.

4. stellar surface gravity – number of exoplanets



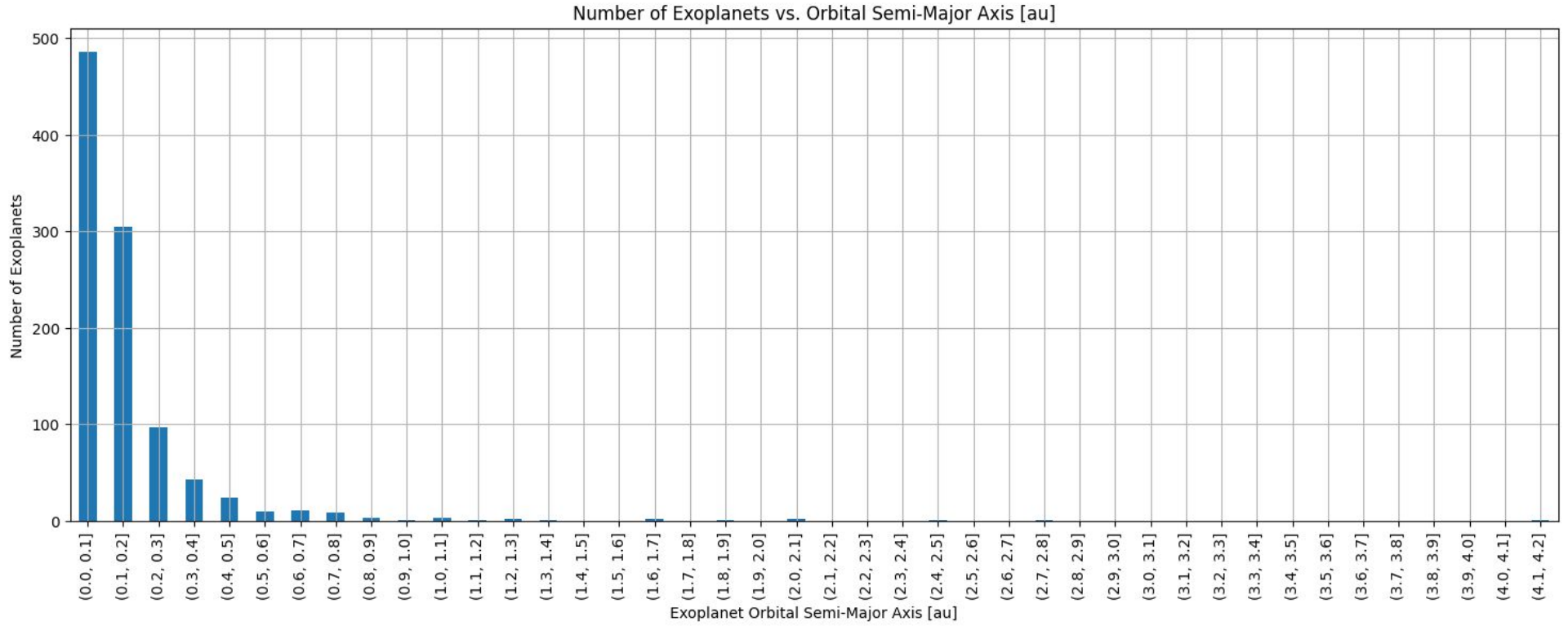
making spicier graphs of the important identified feat.

4. stellar surface gravity – number of **HZ** exoplanets



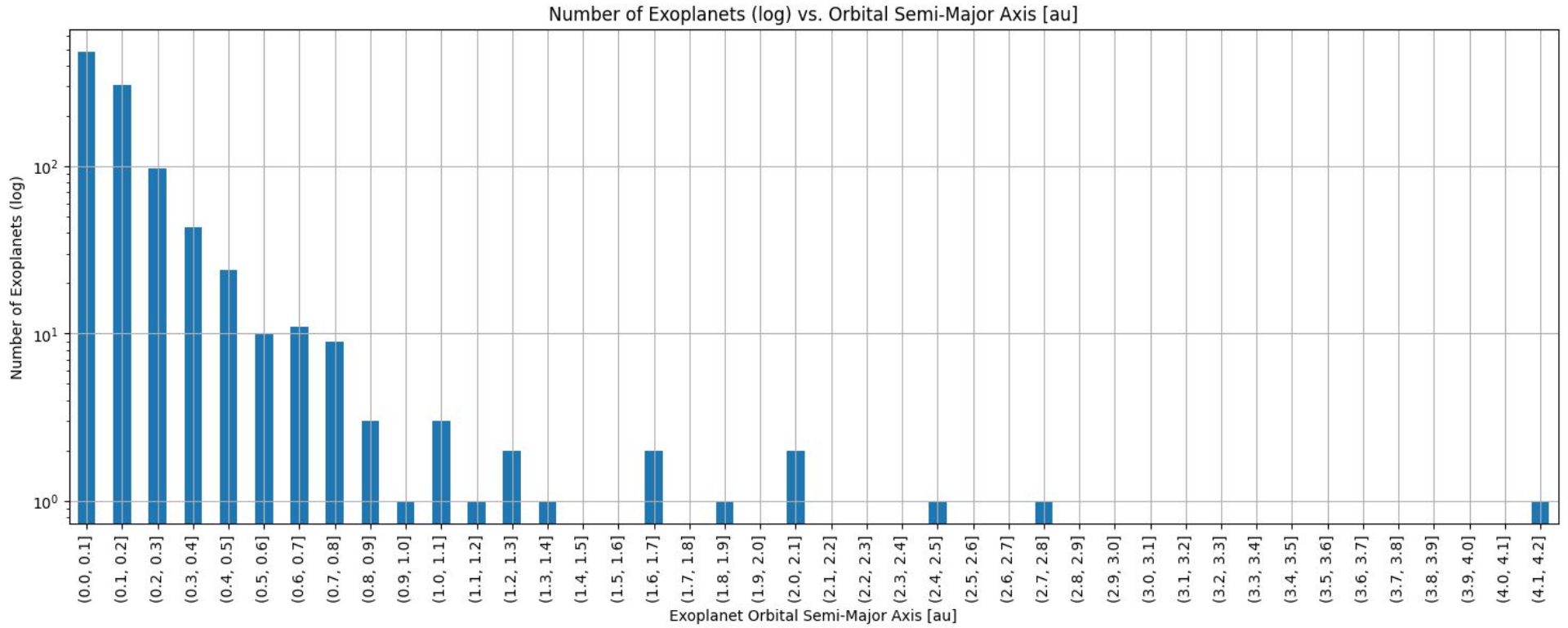
making spicier graphs of the important identified feat.

3. planet orbit (semi-major axis) – number of exoplanets



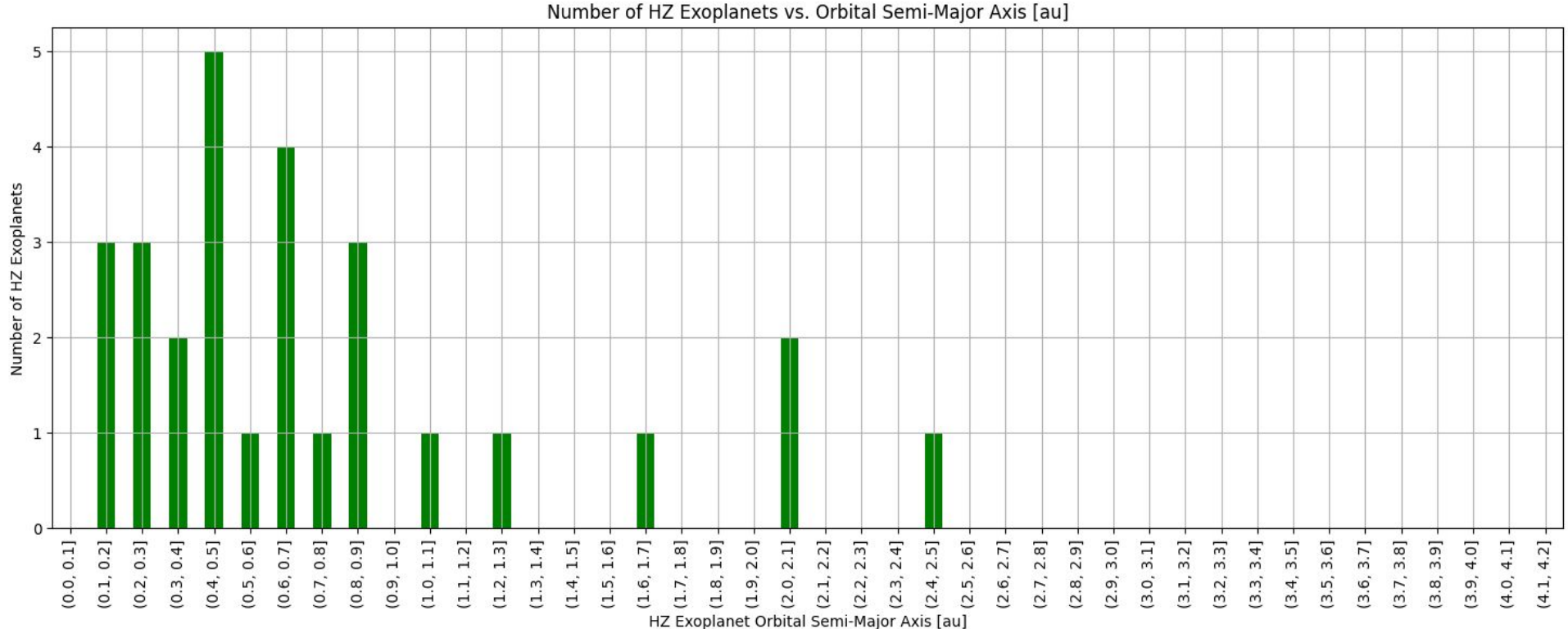
making spicier graphs of the important identified feat.

3. planet orbit (semi-major axis) – number of exoplanets **(log scale)**



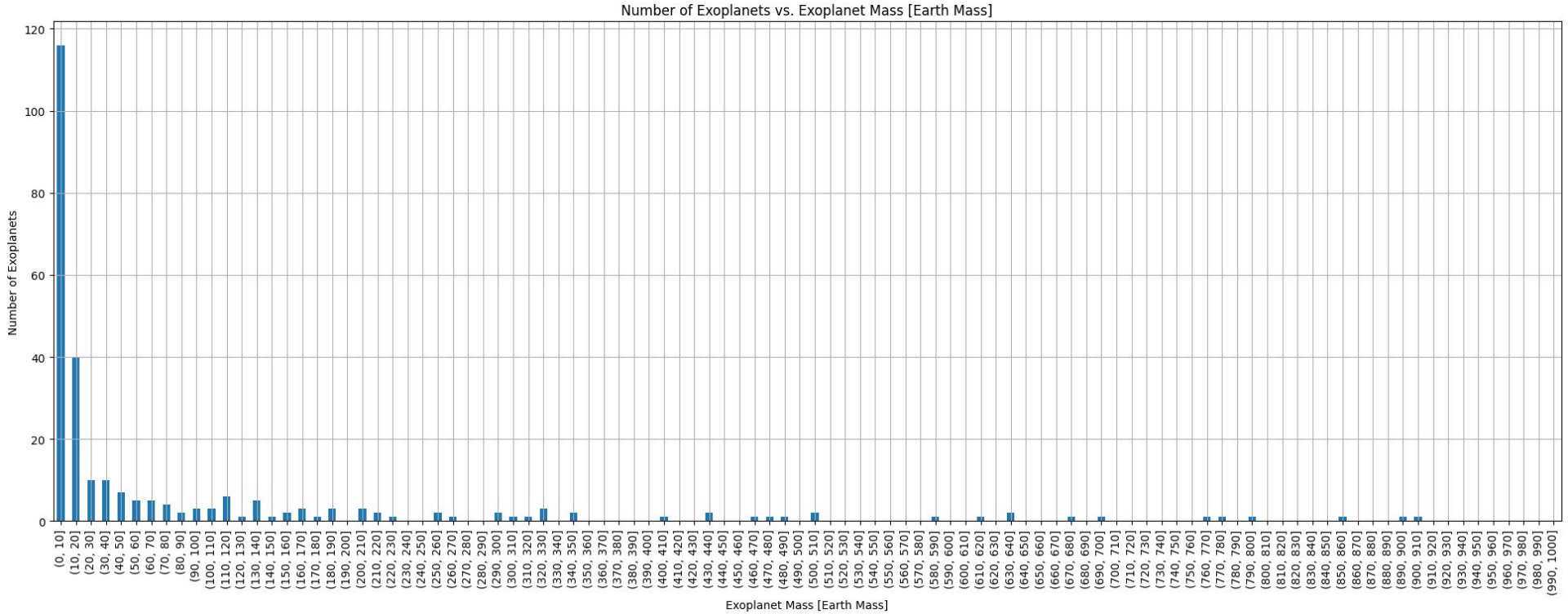
making spicier graphs of the important identified feat.

3. planet orbit (semi-major axis) – number of **HZ** exoplanets



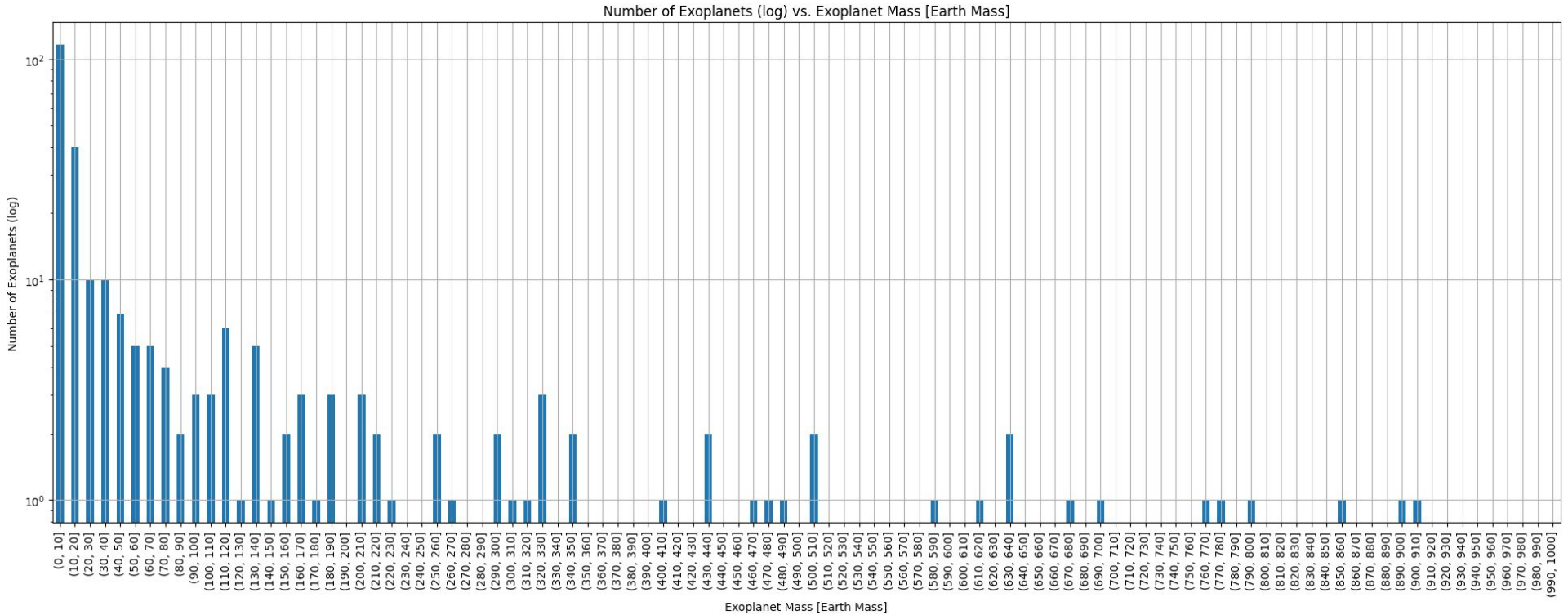
making spicier graphs of the important identified feat.

2. planet mass – number of exoplanets



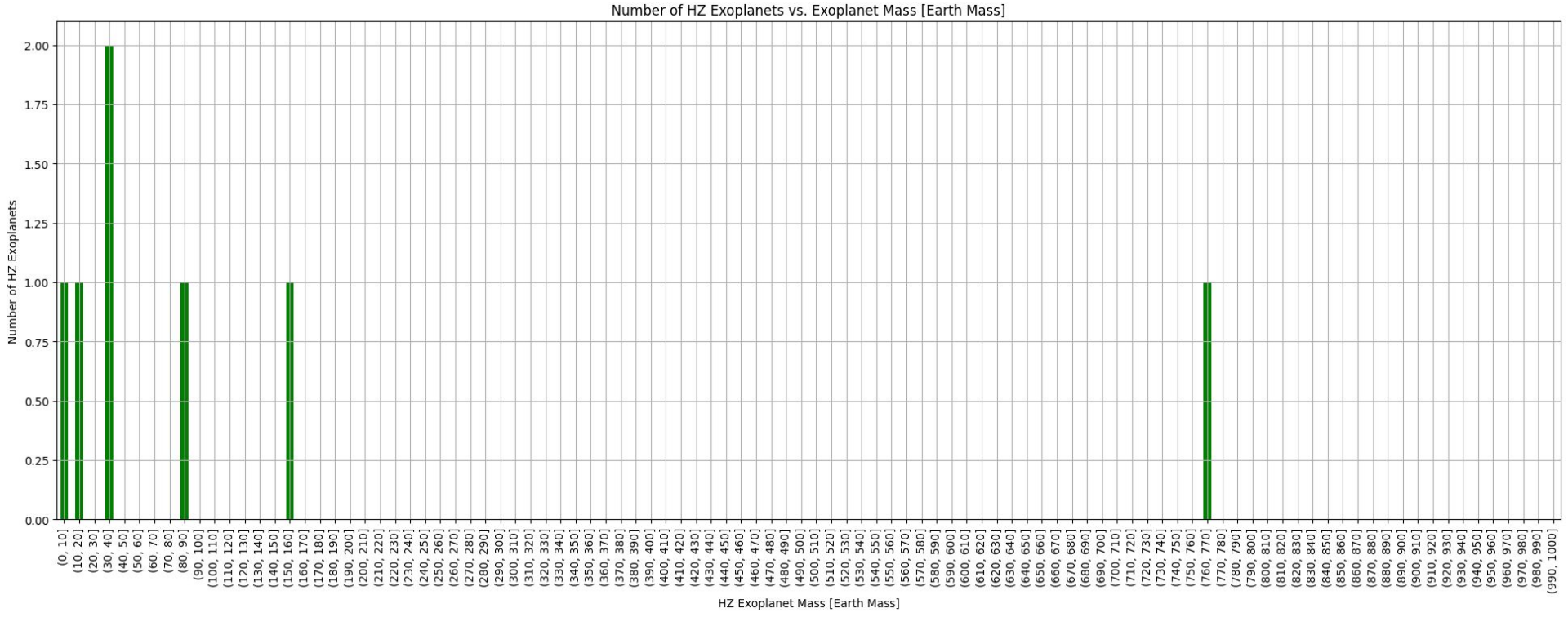
making spicier graphs of the important identified feat.

2. planet mass – number of exoplanets **(log scale)**



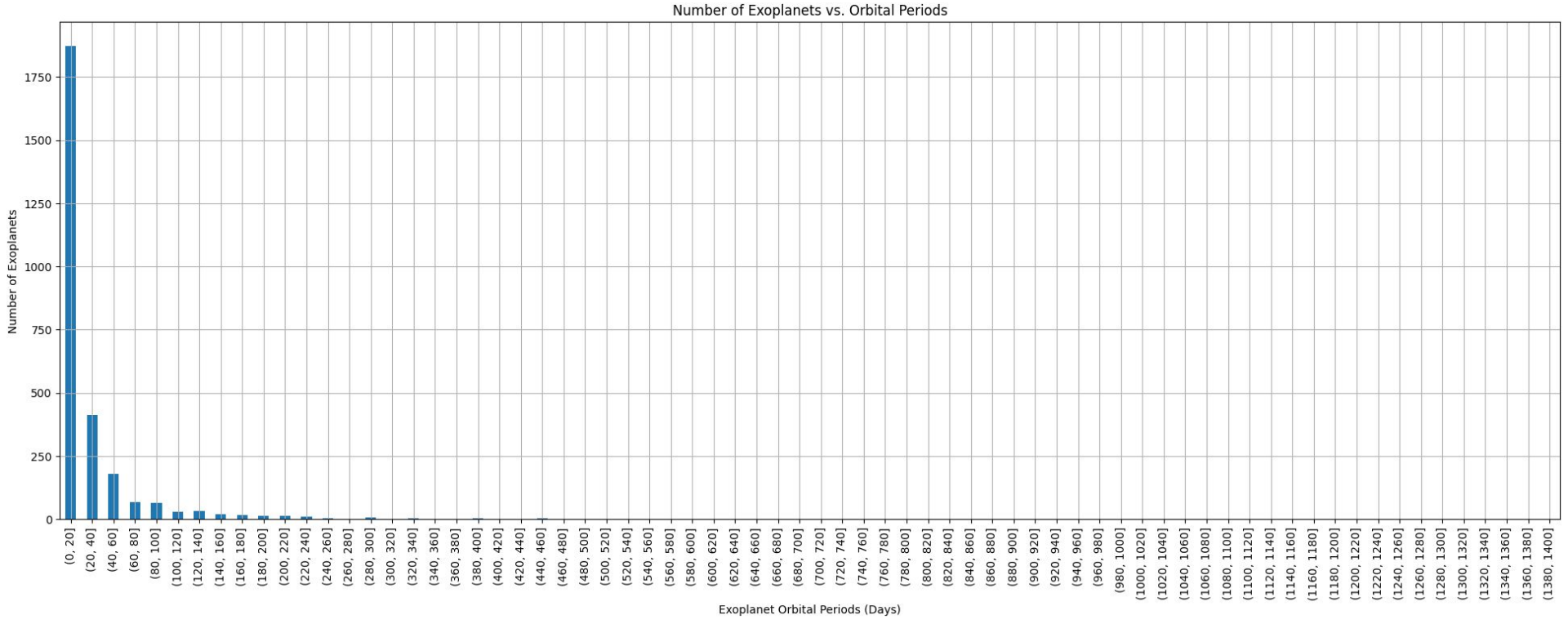
making spicier graphs of the important identified feat.

2. planet mass – number of **HZ** exoplanets



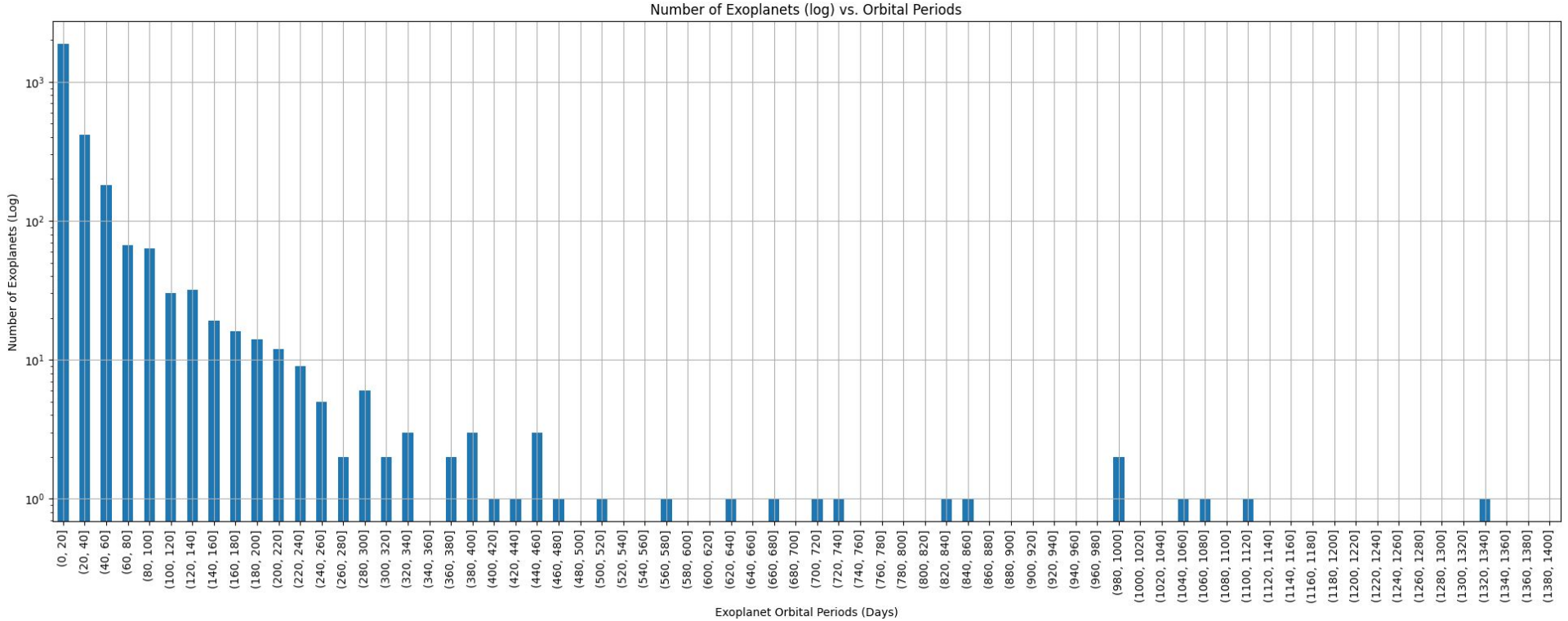
making spicier graphs of the important identified feat.

1. planet orbital period (days) – number of exoplanets



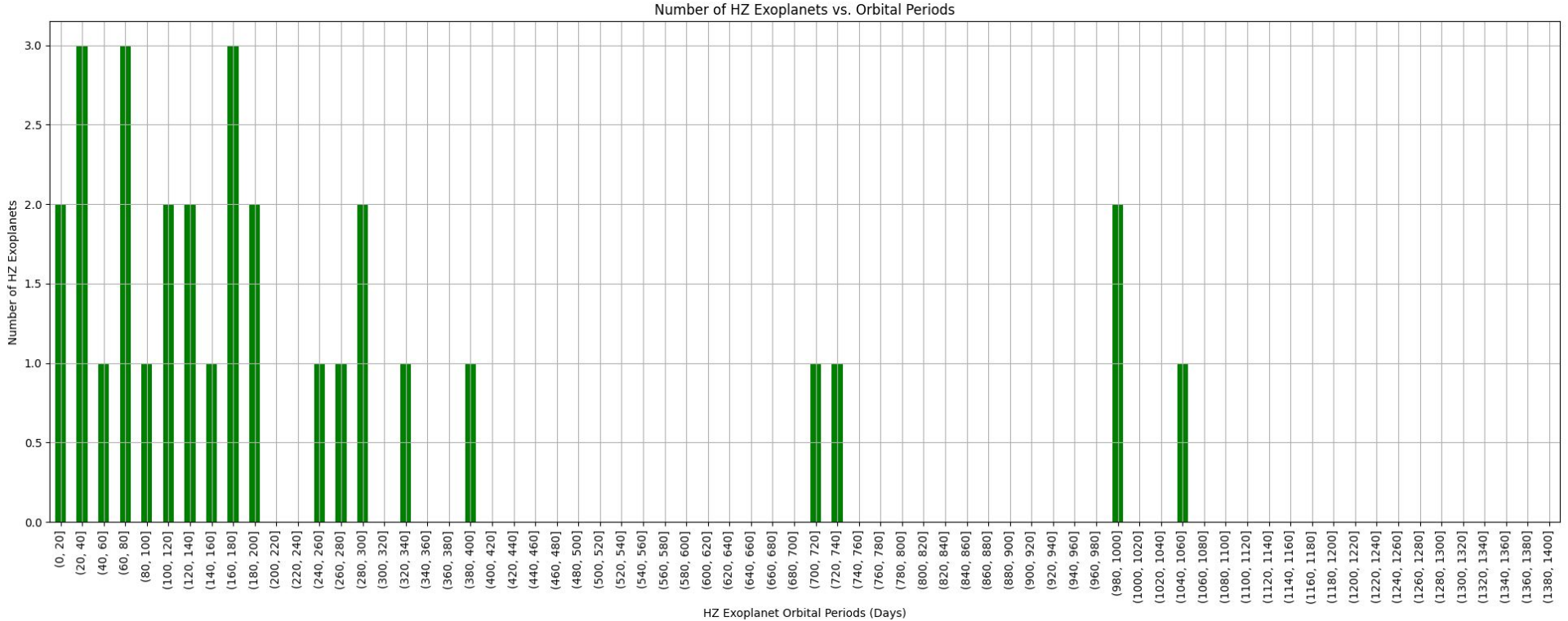
making spicier graphs of the important identified feat.

1. planet orbital period (days) – number of exoplanets **(log scale)**



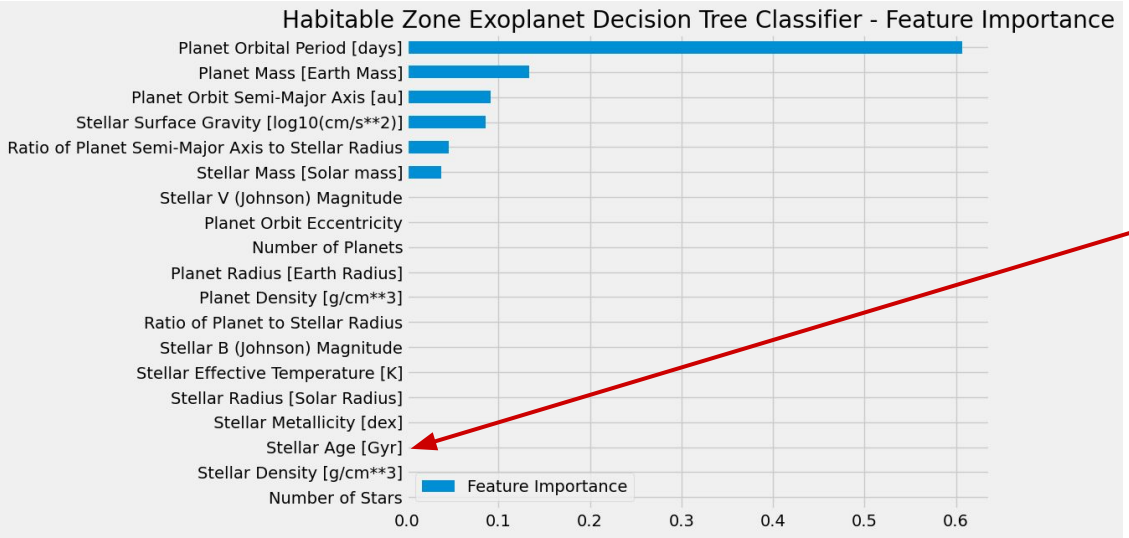
making spicier graphs of the important identified feat.

1. planet orbital period (days) – number of **HZ** exoplanets



something silly…

looks like stellar age didn't matter as much as i thought…



next steps

- fine-tune the classifier
- study the 6 graphs in more detail