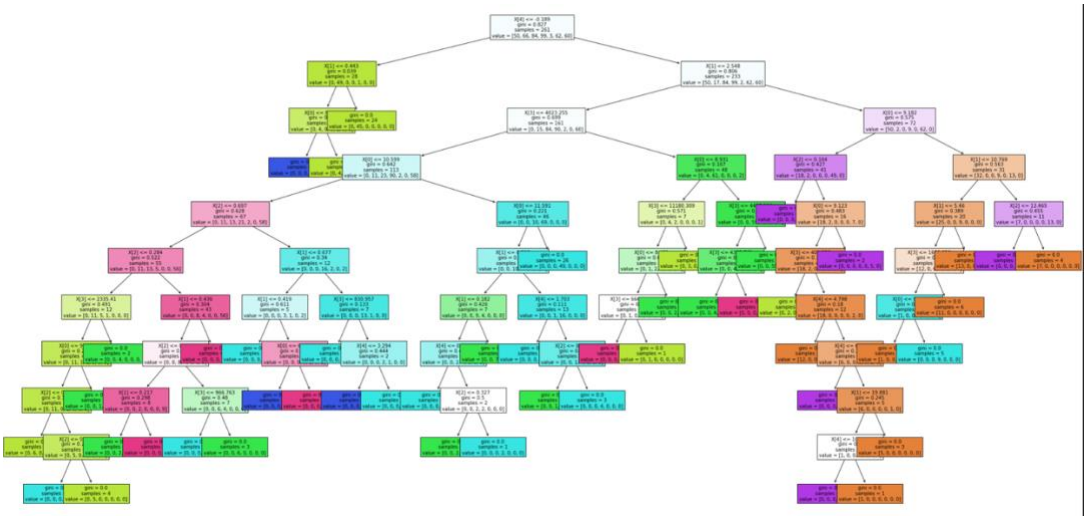
Jadyn Waggoner

1. Project Overview
    - Project Title: A Comprehensive Catalog of Classical Variable Stars in TESS
    - GitHub Repository Link: https://github.com/jadynwaggoner/cmse492_project
    - Brief project description: The goal of this project is to accurately identify and catalog classical variable stars using machine learning algorithms. Data from individual stars collected during NASA's Transiting Exoplanet Survey Satellite (TESS) primary mission were visually cleaned and sorted using plotted light curves, then used to train various sci-kit-learn machine learning algorithms. The categories that the stars get sorted into are RR Lyraes, classical Cepheids, γ Doradus, δ Scutis, contact eclipsing binaries, detached eclipsing binaries, non-variable stars, and miscellaneous. The miscellaneous category identifies special cases that don't fit any known classical variable types, which can lead to compelling discoveries. Utilizing the results from this project, stars from the TESS extended missions will be roughly categorized as the data becomes available. Having these categorized datasets will be useful for any scientist looking to study a specific type of variable star without being required to visually sift through millions of targets
2. Project Setup
    - Description of repository structure and Explanation of key files and directories: There are 5 main folders in the repository structure: data, notebooks, reports, results, and src. The data folder contains subfolders titled raw and processed, and the raw folder will contain the data that I directly download from the TESS full frame image portal and from VizieR. The processed folder will contain the data after it has been sorted into the known variable star types and manually cleaned. The notebooks folder contains the subfolder exploratory which will contain all my exploring and testing python notebooks, and the subfolder final for my final cleaned and commented code. The reports folder will contain my interim and final written reports for the project. The results folder will contain the final figures and models for my project once I create them. The src folder will contain the source code for the data, features, models, and visualizations.
    - List of dependencies and setup instructions: For this assignment most of my time was spent in the writing and testing phase so I do not have a full easy to use notebook written up yet, but I am continuing to work on making it much more user friendly. For now I have created a notebook that contains all the functions I created and comments with how to use them.
3. Completed Tasks
    - List of tasks completed from the homework: Download, sort, and clean data, train sci-kit-learn classifiers to test which one works best, write all the previous code into functions, test to see what combination of columns worked the best for the classifiers
    - Justification for each task's relevance to your project: I was able to download, sort, and manually clean the data from multiple TESS sectors to make it suitable for machine learning. I was also able to create many functions to make the process of cleaning the data and testing machine learning classifiers more easily. I

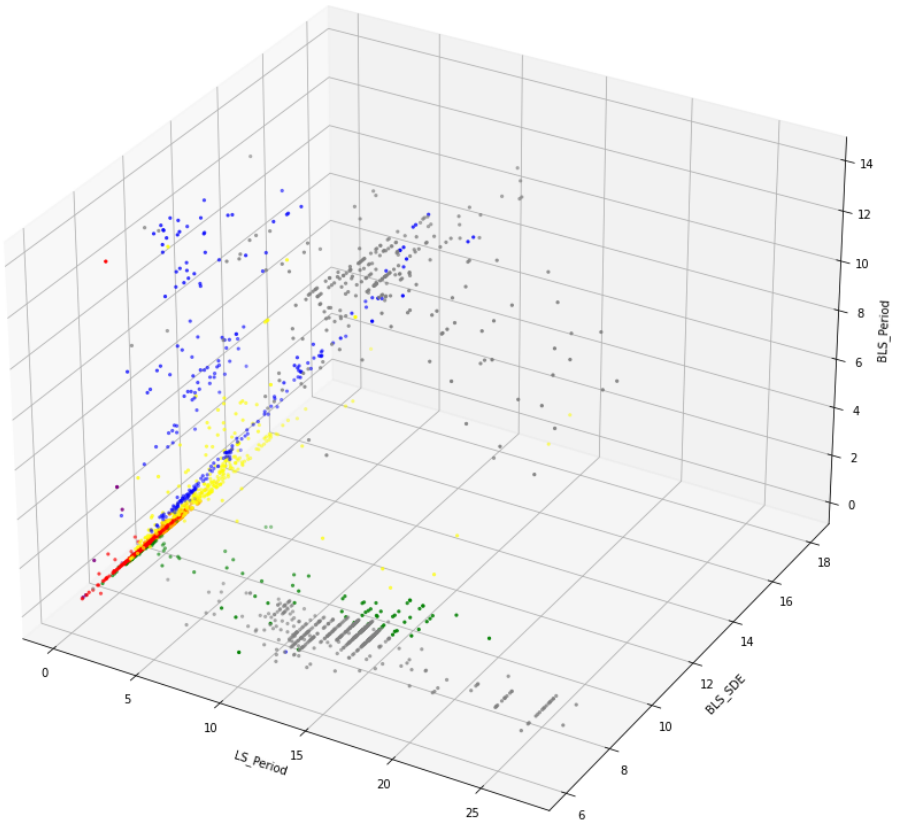
tested various sci-kit-learn classifiers to see which one had the highest accuracy score and would work the best for predicting the variable star type.

- Brief description of the process for each task: I downloaded all the variables and RA and Dec from the TESS FFI portal, and then used the RA and Dec in VizieR to download the labels for each specific star. To sort the data, I used loops to find the indices that contained each specific star types and separated them all out into their own specific lists. To clean each list to ensure that they're all the correct types (the data was very messy), I created a GUI using tkinter that displays a lightcurve for each specific index and two buttons 'yes' and 'no', if you click 'yes' the lightcurve will be saved into a folder and the data will be saved into a csv file, and if you click 'no' that index will be deleted since it is not the correct star type that it was labeled as. Then I tested 'KNeighbors', 'MLP', 'RandomForest', and 'KMeans' classifiers using sklearn, and wrote a function that tested which variables work the best for each of them by seeing what combination leads to the highest accuracy score (or silhouette score for kmeans).

4. Initial Analysis and Findings
   - Summary of key findings from your initial analysis: My key findings are that the classifier with the highest accuracy score (about .95) is the Random Forest classidier, and that the best combination of features is 'BLS_Period', 'LS_Period', 'BLS_SDE', and 'LS_SNR'.
   - Include and discuss relevant data visualizations, plots, or other visual elements: Here is an example of one of the trees created when using the random forest classifier-

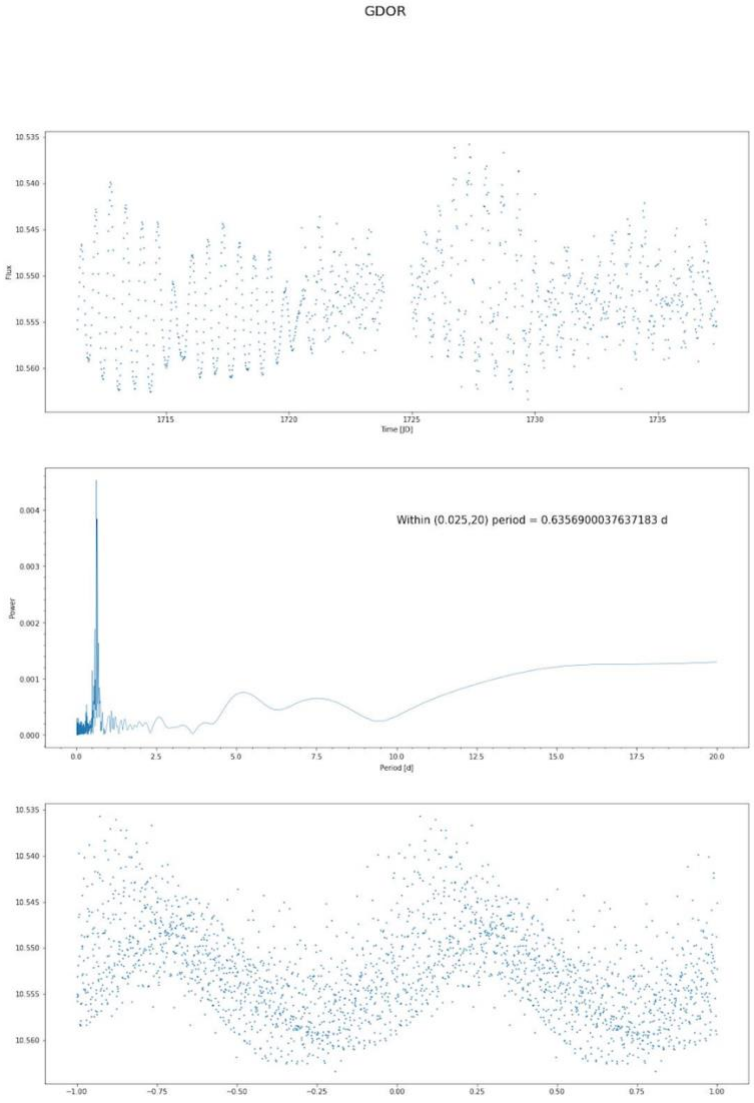


And here is a plot where each axis is one of the best performing features, and each color is a different variable star type to show how the types can be separated. I will work on making this plot more readable with a legend and a title, this was just for testing purposes.



5. Proposed Approach
    - Description of your proposed machine learning approach: After lots of testing, I decided that the Random Forest classifier is the best machine learning classifier to complete my project with. I will train a random forest classifier with the data from as many TESS Sectors as I can in order to ensure that in the future, new data will be able to be sorted from any sector using the trained classifier.
    - Justification for chosen methods: After testing many other sklearn clsasifiers, I found that the random forest classifier had the highest accuracy score after being trained and tested with the TESS data.
6. Preliminary Results (if any)
    - Description of any initial experiments or model testing: I experimented with different sklearn machine learning classifiers, and found that the random forest classifier was the best option for this project.
    - Preliminary findings or insights: With the random forest classifier, I was able to train and test the classifier with a 0.95 accuracy score using data from TESS sector one.
    - Interpretation of early results: This is a surprisingly good result, and I wonder if there is any overfitting happening. I will have to check this and test with data from other TESS sectors.
    - Include plots, tables, or other visualizations of your results:

Here is an example light curve of a correctly predicted γ Doradus variable star using the random forest classifier.



7. Challenges and Solutions
    - Description of significant challenges encountered (both coding and non-coding): While trying to use the pre-sorted labels for all the stars from VisieR, I realized that many stars were either mislabeled or were so messy that it would be impossible to tell what the star type is. I realized that I would have to go through and manually sort through anything I was going to use for machine learning, which was a significant challenge and cost me a lot of time.
    - Explanation of the solutions implemented: I created a GUI using tkinter which allowed me to more easily and quickly sort through all of the stars I was going to use and be able to save them into their corresponding labelled files.
    - Include relevant code snippets illustrating key challenges and solutions: Here is a code snippet from the function I created with the tkinter GUI that was my solution for being able to easily and quickly sort through the messy data.

```
window = tk.Tk()  #Creating a GUI using tkinter
window.title(f"Is this a {dataset['Type'][i]}? Index {i}")
window.geometry("500x1000")

fig, (ax1,ax2,ax3) = plt.subplots(3,1,figsize=(18,25))

ax2.text(0.5, 0.8, f'Within (0.025,20) period = {pg.period_at_max_power}', transform=ax2.transAxes, fontsize=15)

ax1.invert_yaxis()
ax3.invert_yaxis()
ax1.set_ylabel('Magnitude')
ax3.set_xlabel('Phase')
ax3.set_ylabel('Magnitude')

plots(ax1,ax2,ax3)
plt.suptitle(f"{dataset['Type'][i]}",fontsize = 30)

canvas = FigureCanvasTkAgg(fig, master=window)
canvas.draw()

canvas_widget = canvas.get_tk_widget()
canvas_widget.pack(side=tk.TOP, fill=tk.BOTH, expand=1)

def yes_command():  #Defining a button that will append "yes" to the list if the button is pressed
    cleaned.append('yes')
    if not os.path.exists(f"{foldername}/{seccam}_{dataset['Type'][i]}_images"):
        os.makedirs(f"{foldername}/{seccam}_{dataset['Type'][i]}_images")
    plt.savefig(f"{foldername}/{seccam}_{dataset['Type'][i]}_images/{TICID}_{SECTOR}_{CAMERA}_{CCD}.jpg")
    plt.close(fig)
    window.update_idletasks()
    window.destroy()

def no_command(): #Defining a button that will append "no" to the list if the button is pressed
    cleaned.append('no')
    plt.close(fig)
    window.destroy()
```

- Reflections on lessons learned from overcoming these challenges

8. Next Steps
    - Outline of upcoming tasks and milestones: For upcoming tasks, I will be expanding my code to be able to use it on more TESS sectors, which will make the classifier more accurate across the entirety of the TESS data.
    - Any adjustments to the original project plan: This was a part of the original plan, so there will not be any adjustments made so far.
    - Timeline for the next phase of the project: I believe it will take me a few weeks to be able to implement the code and sort through the new TESS sector data.
9. Conclusion
    - Summary of current project status: I was able to download, sort, and manually clean the data from multiple TESS sectors, and I was able to create many functions to make the process of cleaning the data and testing machine learning classifiers more easily. I also tested various sci-kit-learn classifiers to see which one had the highest accuracy score and would work the best for predicting the variable star type, which turned out to be the Random Forest classifier.
    - Reflection on progress and lessons learned: I did a ton of testing and I feel like I made a lot of progress on being able to predict what a variable star type is based on the features given. The main lesson I learned is to not trust any data to be cleaned, and to always double check it and clean it myself.
    - Outlook for project completion: Since I am getting a high accuracy score with multiple TESS sectors, and the light curves that are outputted after testing them are looking accurate, I believe that the outlook for completing my project looks good and that I will be able to finish it on time.