Capstone #1 Final Report

Introduction

Question at hand

When we look in the night sky, we can see a number of twinkling lights. Most of them are stars, and some of them are planets in our solar system. The farthest object we can see in the sky with the unaided eye is, in fact, a galaxy! Andromeda, sitting 2.5 million light years away, is a spiral galaxy similar to our Milky Way.

Galaxies come in a variety of shapes and sizes, but they can generally be grouped into three categories: irregular, elliptical, and spiral. Spiral galaxies can appear to us as either spinning clockwise (CW) or counterclockwise (CCW). According to one of the major theories in astronomy, the universe is isotropic (i.e. the same in all directions). This means that the number of CW and CCW galaxies in the universe should be equal and evenly dispersed throughout the sky. But is there any way to distinguish the two types of galaxies, or do they all “look” the same?

Why is it important?

Given a variety of observable properties about a galaxy (such as size, brightness, and tilt angle), we can try to predict the rotation of the galaxy. This can prove to be useful for astronomers when conducting sky surveys (i.e. observing and cataloging a large number of objects in a particular region of the sky) so they would not have to manually classify each galaxy they come across. The rotation of a galaxy indicates which part of the galaxy is moving closer to the Earth and which is moving away; this can be helpful when calculating its Doppler shift, an important physical property for objects in space (stellar objects). Additionally, we can use this information to verify the isotropy of the universe. If there are more CW or CCW galaxies confined to a certain area of the sky, this could challenge the prevailing theory and should be further studied.

Data acquisition and cleanup

Data source

The data used for this project was acquired from this site. The data had already been collected and cleaned for the most part, so I expected my data wrangling to be minimal. The handedness of each entry was sourced from the Galaxy Zoo 2 project. This project allows anyone to view images of galaxies and classify their shape and structure. Because this classification is done by non-experts, only galaxies with a 90% or higher rate of agreement were included. The data was then “supercleaned” by a galaxy image analyzer and visually inspected, resulting in an error-free dataset. The photometric data (observables) of each galaxy was acquired from the findings of the Sloan Digital Sky Survey, a vast collection of data on millions of objects in the sky. Again, all of this had already been accomplished and compiled into one CSV file. I still, however, had to perform a few more steps until the data was clean and ready for analysis.

Loading the data

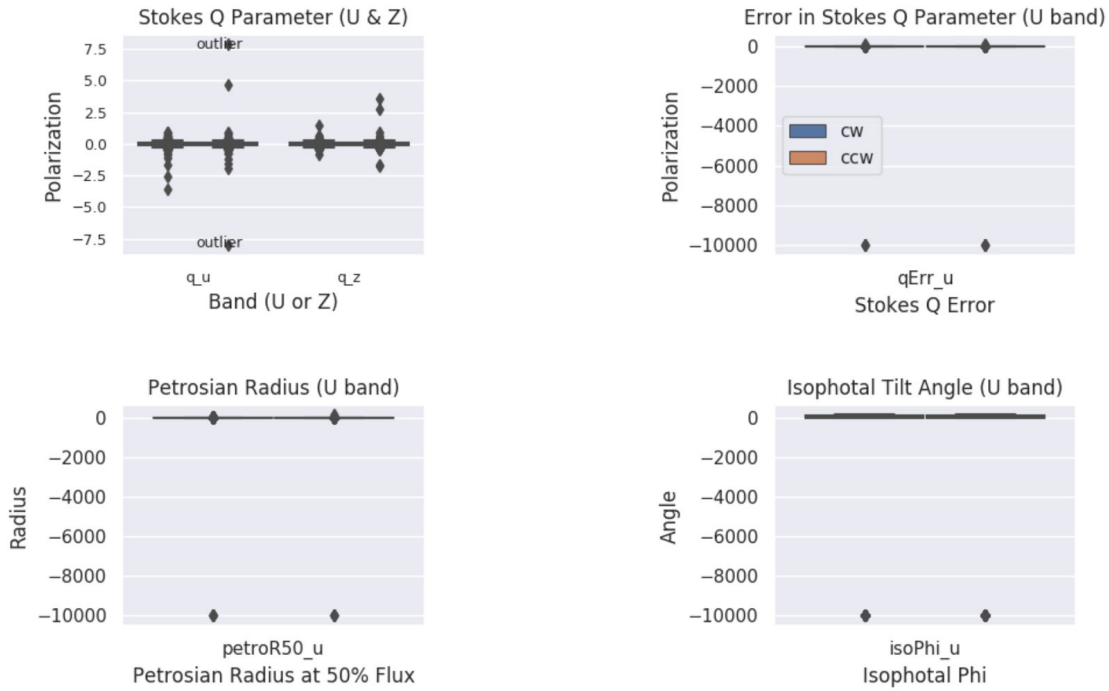
I used the pandas read_csv function and passed the URL of the CSV file. This saved me from having to download the full file, as the file size is too large to upload to Github. The dataset has 13,440 rows and 455 columns, so my next task was to organize the columns into groups in order to make preparation and analysis more manageable. I did this by inspecting the columns, grouping them into relevant categories, and saving the groupings as a CSV file. I then read in the file and used the groupings to create a multi-level version of the original dataframe. This way, I could access any subset of the dataframe I needed.

Then, I created a method which pulled out the subset to create an entirely new dataframe (preserving the rotation column), in case I needed to perform separate analyses on different groups. For example: the “coordinates” category, which indicates the location of the observed object in the sky, and the “stokes” category, which contains signal intensity measurements, will likely not benefit from the same statistical methods. It makes more sense to extract these categories into separate dataframes as needed. Once I was able to make the data more accessible and manageable, I moved on to the next step.

Screening for outliers

In order to verify if there were any missing values, I checked the dataframe to see if there were any null values in any of the columns. This includes None, np.nan, and empty strings. I also made sure that there were no “object” data types in the dataframe, save for the “rotation” column. As it turns out, there are no erroneous values in the entire dataframe. Thus, I was able to confirm that there were no missing entries, and no further cleaning procedures were necessary.

I created plots of various variables within each category. At first, I wanted to check the measurements of the Stokes parameters (which indicate polarization) to make sure they did not have any outliers. I plotted data for the Stokes Q parameter across two bands (U and Z) and separated them by rotation. Once I created the chart, I noticed two outliers in the Stokes Q parameter in the U band. Upon further inspection, I realized that these two entries had anomalies in many fields of the Stokes dataframe (one column even had a value of -9999). After plotting some more columns, I found that several entries had -9999 in one or more columns of the dataframe. This dataset, as I realized, uses -9999 as its null value. I decided to retain all of the entries and simply replace all instances of -9999 with NaN values. The number of NaN values in this dataset only represents 0.16% of the total data. This does not seem to be significant enough to affect our analysis. We have still retained all rows and columns. Fortunately, we did not lose much data in the data wrangling process.



Top left: plot of Stokes Q parameter in U and Z plots. Upon investigating the two outlier values indicated, I found entries of -9999.00 within the Stokes category (top right). The remaining two plots show that -9999.00 appears throughout the entire dataframe. All of these values were replaced by NaN.

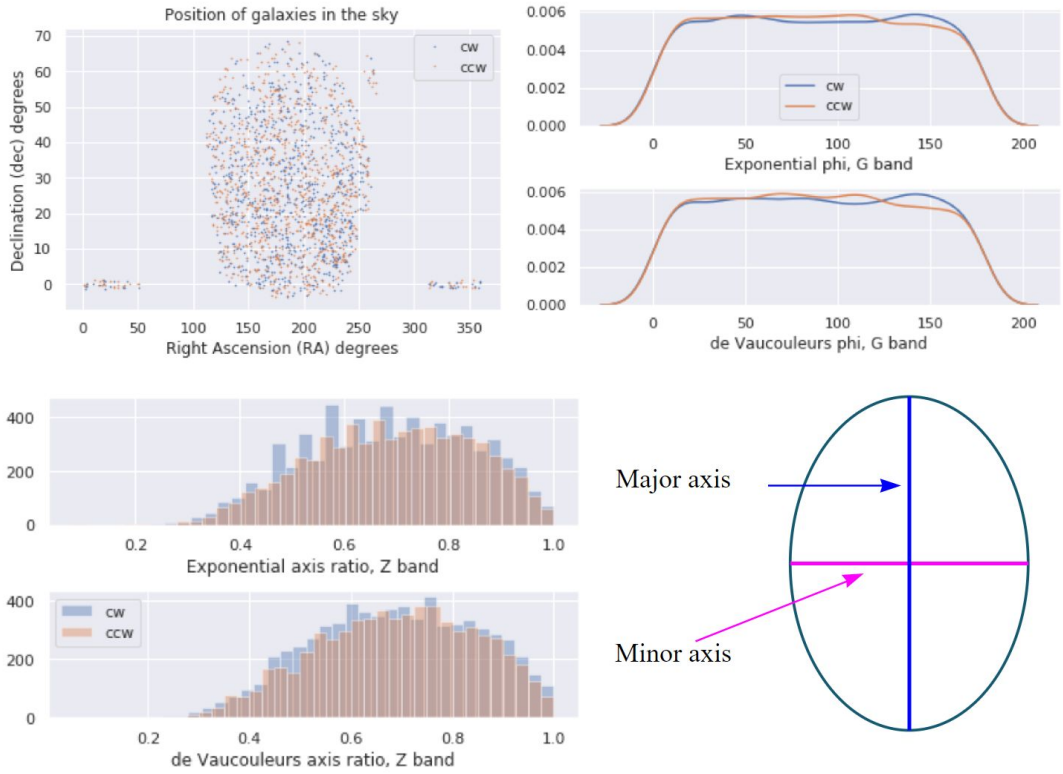
Exploratory and statistical data analysis

Visualizing our data

Now that our data is clean, we can create some plots to gain an understanding of what our data looks like. First, we can create a scatter plot of the coordinates of each galaxy, separated by rotation. Looking at the chart, we can see that the CW and CCW galaxies appear to be evenly dispersed (that is, no section has more blue/orange than another).

Similarly, we can examine the distribution of the tilt angle (phi) of CW and CCW galaxies. By inspecting the chart, we can see that CW galaxies tend to have a larger tilt angle than CCW galaxies.

Finally, we can take a look at the distribution of galaxy shapes. The ratio of major and minor axes in an ellipse determines its shape. The closer the ratio to 1, the more “circular” it is, and the further it is from 1, the more “oblong” it is. We can observe the axis ratio of CW and CCW galaxies. It seems that CW galaxies are more likely to have a smaller axis ratio (further away from 1), while CCW galaxies are more likely to have a larger axis ratio (closer to 1). This means that CW galaxies in this dataset are typically more "oblong" than CCW galaxies.



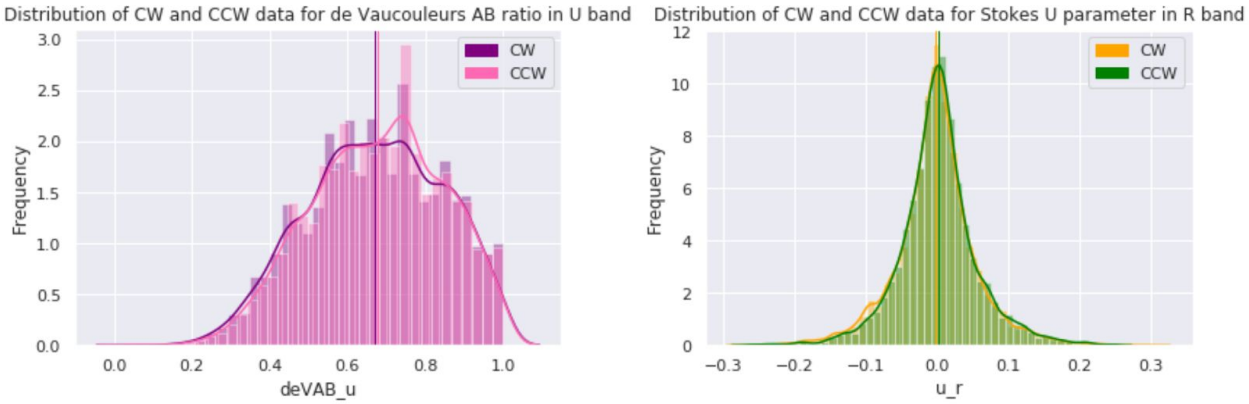
Top left: distribution of coordinates of CW and CCW galaxies. Top right: distribution of tilt angles of CW and CCW galaxies. Bottom left: distribution of axis ratios of CW and CCW galaxies. Bottom right: location of major and minor axes of an ellipse.

Statistical tests and analysis

After cleaning, inspecting, and tidying the data, we are ready for statistical analysis. Because the data has 454 features, and it is not feasible to examine each one, we need a way to isolate the ones with the most influence over the rotation of a galaxy. We identified the features that have a significant difference of means between CW and CCW galaxies since a difference in means can be indicative of varying distributions. In order to test for the difference in means of the two samples, we performed a two-tailed t-test for independent samples: two-tailed since we screened for any difference; t-test since we do not know the standard deviation of the population, which in this case is every single galaxy in the universe; and independent because the spectral data of one galaxy does not depend on the others. Because we will be using the results of the analysis to draw conclusions about the population, we are assuming that our sample is representative of the population. Our sample was taken from only one hemisphere of the sky, so we cannot truly be certain. Due to the isotropic nature of the universe, and for the sake of simplicity, we will consider our sample to be representative.

I iterated through all of the columns and performed a t-test on each one in order to determine if there is a difference in means between CW and CCW entries. My null hypothesis was that there

is no difference in means: $\mu_{CW} - \mu_{CCW} = 0$. My alternative hypothesis was that there is a significant difference in means. I set $\alpha = 0.05$ as it is a typical measure of statistical significance. Once the t-tests were complete, I added all features with a p-value of $p < \alpha$ to a list called "significant_cols." Out of 454 features (not including the "rotation" column), I was able to isolate 61 features where the difference in means between CW and CCW galaxies was statistically significant. This means that for these 61 features, we were able to reject the null hypothesis. For the others, we fail to reject the null hypothesis.



Distributions of CW and CCW entries in two of the 61 significant columns. The vertical lines indicate the means of each distribution. The difference in means appears to be small, but not zero.

Machine learning analysis

Preparation

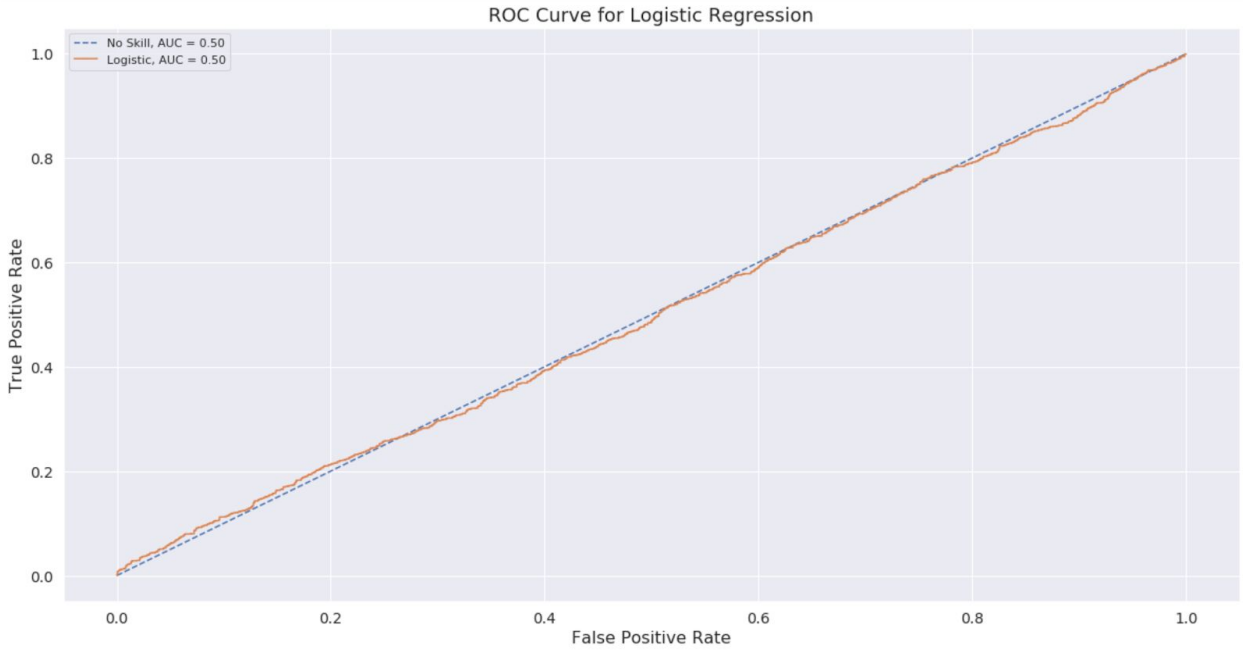
Now, we can build a machine learning model that will attempt to predict the handedness of a galaxy. The data is loaded and cleaned according to the data wrangling techniques established earlier. Rather than imputing any NaN values, I opted to drop all samples with NaN entries. We lost about 4% of our samples, but I do not think this enough to skew the results. Then, the data was split into training and testing sets (with a 75%/25% split). Below are the various models I tried and their results.

Logistic Regression model

For a baseline model, I chose to use logistic regression. Initially, I set all of the hyperparameters to their defaults. The parameter C, which inversely corresponds to the regularization strength, was initially set to 1. The model's performance was judged by its mean accuracy. This baseline model yielded a test score of 0.519, which is only slightly more effective than a coin toss. Clearly, we need to refine our approach.

I then used grid search cross validation with five folds on my logistic regression model in order to tune the regularization parameter C. The model suggested that the optimal value of C is

C=0.0001, which is far lower than the default value of C=1. The lower the regularization parameter, the harsher the model penalizes the size of its coefficients. The strict penalty forces the model to keep the features and parameters it deems most important and reduce the rest. Unfortunately, correcting the value of C resulted in the same score of 0.519 on the test data. After plotting the ROC curve for this model, we can see that the curve falls along the no skill line and has an area under the curve (AUC) of 0.50; again, this is akin to a coin toss.



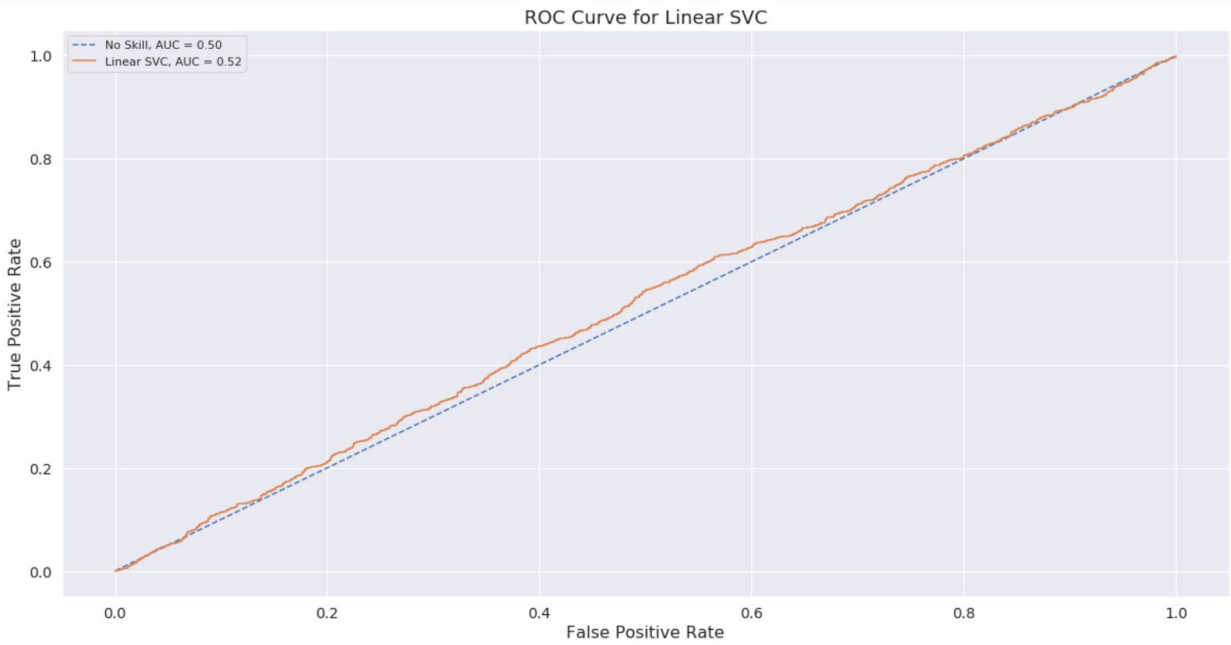
ROC curve for the logistic regression model. This model was cross-validated with five folds and the hyperparameter C (corresponding to the regularization strength) was tuned to C=0.0001. The AUC is 0.50, so a different approach is needed.

Support Vector Machine

Next, I tried using a support vector machine (SVM) classifier with a linear kernel. First, we needed to scale our data so that our features have the same mean, variance, and units. We arrived at a score of 0.523, which is slightly better than the logistic regression model. The ROC curve has moved away from the no skill line and the AUC is now 0.52. There is definitely some improvement with this model.

The main drawback of using an SVM is that they generally do not handle high dimensional data well. My feature space has 454 dimensions with 12,952 samples. Drawing a hyperplane that cleanly separates the two classes in 454-dimensional space is difficult for a support vector machine to do, especially with a linear kernel. I decided to tackle this issue by performing principal component analysis (PCA) with 100 components on the data with the hopes of reducing the dimensionality to accommodate the SVM. PCA works by transforming our feature space into a lower-dimensional space where the new, rotated axes are linear combinations of

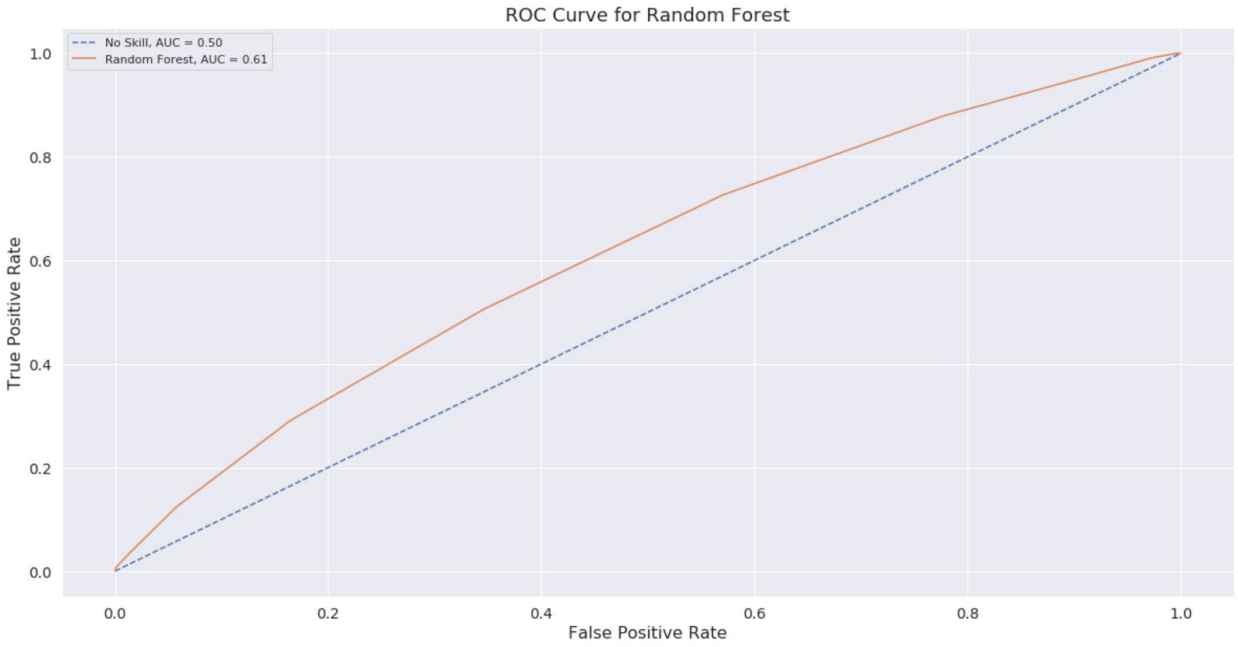
the original features. Unfortunately, this model yielded a test score of 0.515, so there was no improvement. Instead of continuing to tune the SVM's hyperparameters, I decided to try another model.



ROC curve for the support vector classifier with a linear kernel. These models generally cannot handle high dimensional data with many features, but even after reducing the dimensionality with PCA, the results did not improve. The AUC is 0.52, so a different approach is needed.

Random Forest

Next, I chose a random forest model. Due to their ability to handle high dimensional data and compute multiple decision trees in parallel, this seemed like a logical choice. Our test score was 0.577, which seemed to be an improvement, but the training score was 0.988. This is clear evidence that the low score was a result of overfitting. The ROC curve has moved further away from the no skill line and our AUC is 0.61. Generally these are indicators of a better model, but in this case, our model overfit. Random forests are already adept at combating overfitting, so the fact that this model overfit suggests that a different approach is needed.



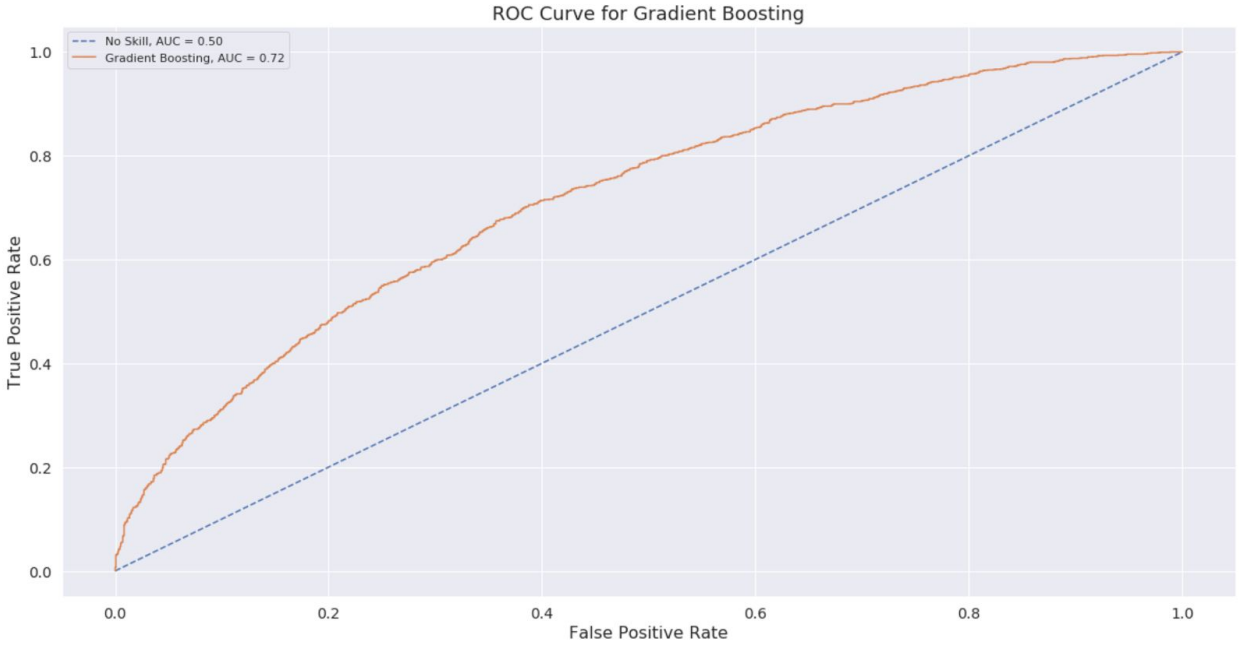
ROC curve for the random forest model. The testing score did improve slightly but the training score was extremely high, which indicates that this model overfit the data. The AUC is 0.61, so even though we are making progress, we cannot use an overfit model.

Gradient Boosting

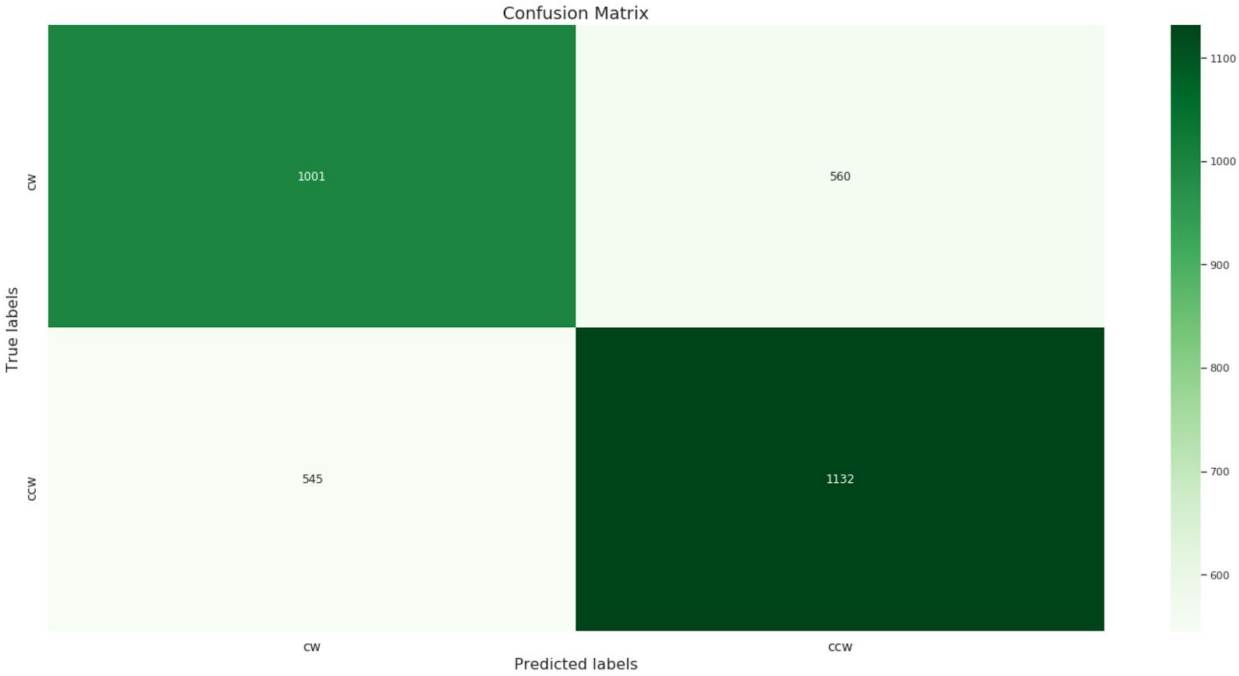
Finally, I used a gradient boosting model with 100 estimators and a learning rate of 0.1. This model is similar to a random forest, except that the decision trees are not computed all at once. The trees are added one by one, where each subsequent tree takes into account the prediction error from the trees before it. As a result, gradient boosting models are fairly resistant to overfitting. We arrive at a test score of 0.659, our highest score so far. Our training score of 0.726 does not show strong evidence of overfitting. The ROC curve is clearly separated from the no skill line, and our AUC for this model is 0.72.

Literature on this topic cited an accuracy score of 64%[1]. Because I achieved a score of 66%, I accepted this model and decided to analyze the results. Upon plotting the confusion matrix, we can see that there are a few more true negatives and false negatives than true positives and false positives. This may be due to the slight imbalance in the sizes of the positive and negative classes. The original dataset had an even split of positive and negative classes, but when we dropped the NaN entries from the data, this resulted in an uneven sample size. I do not believe this affected our analysis too much, however.

[1] L. Shamir, 2016: Asymmetry between galaxies with clockwise handedness and counterclockwise handedness



ROC curve for gradient boosting model. The AUC is 0.72, which is a significant improvement over the other models we tried. Our testing score is 2% higher than what the literature on this subject reported, so I accepted this model.



Confusion matrix for gradient boosting model. This is indicative of uneven positive and negative class sizes, but I doubt this affected our analysis too much.

Summary

Below is a summary table of the various models and their accuracy scores. As we can see, the gradient boosting model performed the best on the testing data.

Method	Training Score	Testing Score
Logistic Regression with default parameters	0.518	0.519
Logistic Regression with Grid Search CV	0.518	0.519
Support Vector Machine with linear kernel	0.559	0.523
Support Vector Machine after PCA	0.528	0.515
Random Forest	0.988	0.577
Gradient Boosting	0.726	0.659

Next, I examined the most important features as determined by the gradient boosting model. I found it interesting that the photometric data as fitted to the isophotal model (i.e. features that begin with “iso”) had the greatest influence over whether a galaxy is clockwise or counterclockwise. The top four features have to do with the tilt angle (phi); this suggests that CW and CCW galaxies are tilted at different angles as observed from Earth.

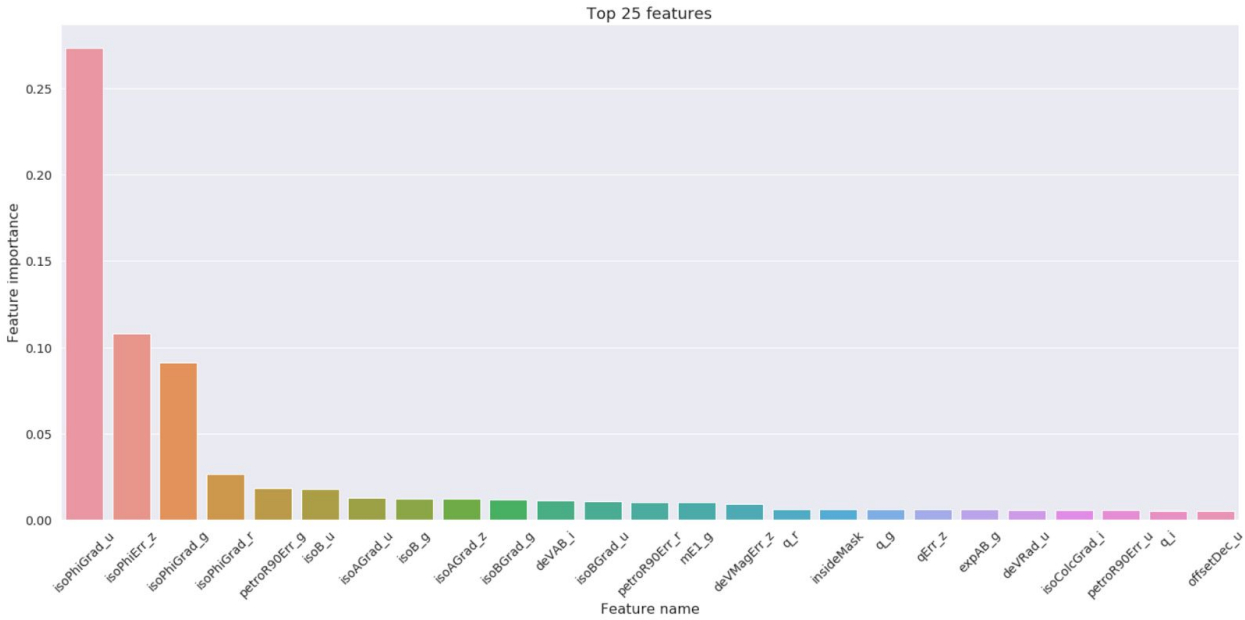


Chart of the 25 most important features as calculated by the gradient boosting model. Features from the isophotal category (beginning with “iso”) are especially prevalent in this list. Also, this list has more features in the U and G bands than in the R, I, and Z bands combined.

Conclusion

According to the analysis above, we can use our model to determine a galaxy’s handedness given its photometric data with a mean accuracy of 66%. While this model is not quite ready for

scientists to use, I am confident that one can eventually achieve a higher score. We could tune the hyperparameters of the gradient boosting model by performing a grid search over various values for the number of estimators and learning rate. On the other hand, we could try a different approach altogether and use a clustering algorithm such as K-means to see if we can divide the samples into two distinct classes.

I believe that scientists will eventually use algorithms such as this one (albeit a much more enhanced version) to automate sky surveys. Surveys are tedious to perform and organize, but they are absolutely essential to astronomical research. Since obtaining telescope time is competitive and expensive, it is extremely helpful if someone else has already conducted thorough and detailed observations for you. The increasing number of major advancements in astrophysics requires an increasing level of sophistication in handling the large amounts of data available to us. The sky will never run out of data; it is up to us to determine how to make the most of it.