Nikhil  Reddy Pasunoor

1. **Exploratory Data Analysis:** The challenge asked us to identify which type of plants can be grown in different geographical locations on mars. After exploring and plotting the Plant_Type feature we came to know that there are 7 different types Plant_Types. This becomes a multi class classification problem. When we plot the count of Different types of plant types we observe that Ascomoid and Assassin vine contribute to most of the samples in the train set. The rest of the plant types are very low compared to these two with the lowest being Dark tree which has only 2184 samples. So, this is an imbalanced data set. The 1 and 3 neighborhood types are around 200K each while the rest 2 and 4 are around 60k combined. The similar is observed with respect to turfs we can clearly see that some turfs are occurring more with the turf 29 being occurred the most times. When I have plotted the missing values in both train and test, I have observed that there were missing values in only shadow_in_midday column. There were around 20 percent of that column having missing values. While exploring the data I wanted to explore how the plant types were spread across various neighborhoods and various turfs. I have observed that the dark tree was only present in neighborhood 4 and kelpie and basidirond were present only in neighborhood 3 and 4. The train data had most samples of Assasin vine and Ascomoid so its not surprising that these two are seen in most of the neighborhoods and turfs while the rest of the plant types are missing from some neighborhoods and some turfs. The box plots show that there are some outliers in Shadow_In_Midday and Shadow_In_Morning and also, the most of the features appear to have skew. There does not seem to be a relationship on any two numerical input features of the data set.
2. **Data Preprocessing:** There was only one feature which was Shadow_In_Midday which had null values in both train and test set. After observing this column statistics, I think it's safe to replace the missing values with mean. I know this might be a bit inefficient and there exist some better practices like using knn imputation or multiple imputations but to simply avoid the computational complexity and substituting with the mean seems a reasonable approach, I have proceeded with the mean imputation. The features Shadow_In_Midday and Shadow_In_Morning have some clear outliers unlike other features, so I have eliminated those outliers and this only reduced 16 samples from the train data set.
3. **Feature Engineering:** Before feature engineering I wanted to think about the various algorithms that I might use and how the data would fit those algorithms. I know that the neighborhood type and turf are categorical variables and the rest are numerical features. For tree based algorithms I don't have to encode these categorical features as they seem kind of like label encoded so this might be ok and tree based algorithms don't need numerical features to be scaled. So I made two copies of the train data set one which is left as it is and the other where the numerical features are normalized and the categorical features are one hot encoded. I know the neighborhood type can be one hot encoded because they were comprised of only 4 types so it would add only 4 more columns to our data set, but turf had 40 different types so I was a bit skeptical to one hot encode this

feature because this would add 40 more columns to our data set. I thought that this will bring the total number of features to 54 which might not be very large. So considering this I went ahead with my approach. I didn't think about feature selection as there were initially only 12 features which is not large.

4. **Data Preparation for Predictive Modeling:** I had initially observed that the data set had class imbalance problem. So, to tackle this I wanted to use smote approach to increase the minority class samples. This was taking some very long time and I couldn't get the data set. When I thought about it the minority class were really low so this might increase the data set size substantially. I took another approach now where I did under sampling. I know that this is not preferable in most cases as data is lost but I did this to do quick experiments with different algorithm's and find best hyper parameters.I took only 2184 samples from the data set from each class because dark tree had only 2184 samples so I could and thus I have ended up with a data set with approximately 15000 samples. I did this for both the versions of the data where on version was one hot encoded and normalized and the other wasn't. I could have used PCA to reduce the dimension, but I didn't think it was necessary.

   Before diving into the model's, I wanted to make clear that by now I have made four versions of the data set. In the beginning of the variable if it is small x or y then it is the original data and if It is capitol X or Y it is sub sampled and balanced version of the data. Also, at the end of the data variable if it has 1 then it is not one hot encoded or normalized but if it has 2 then it is one hot encoded and normalized. Please keep in mind I did create 4 copies of data, but this was necessary to do quick experiments and ideally, I should have created just 2 copies.

   Ex-1: x_train1 is original data that is not subsampled and not one hot encoded or normalized.

   Ex-2: X_train2 is subsampled and one hot encoded and standardized.

5. **Classification Model Predictions:**

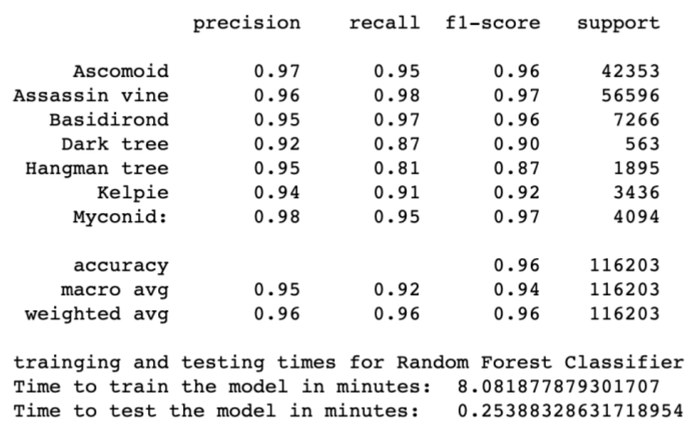
   At first, I wanted to use kNN, Random Forest and neural network. I first tried to run KNN on the entire data set and this was not terminating even after some time. After this initial experiment I have thought for quite some time and eliminated knn and neural network as these might take lot of time, as lot of computation was involved in these algorithms. I narrowed my three approaches to SVM , XGBoost and Random Forest.

   I did run Random Forest classifier, SVM and XGBoost on the under sampled version to find out the best hyper parameters. After I had found out the best hyper parameters I built my final classifiers and trained them on entire original version of the data set. Please note that I have used one hot encoded and normalized version of data set for svm and original copy of data set for tree based algorithms.

   Based on the results and taking into consideration the training time I have chosen Random Forest Classifier which had an F1 score of 0.96.While XGBoost had a bit slighter good results compared to Random Forest it had very high training time compared to Random Forest Classifier

   As the algorithms take longer to run I am enclosing the screenshots of my results.

Results of Random Forest Classifier:



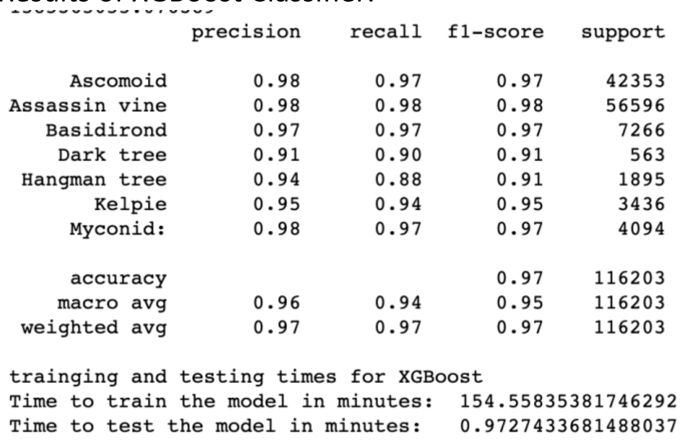
```
               precision    recall  f1-score   support

     Ascomoid       0.97      0.95      0.96     42353
Assassin vine       0.96      0.98      0.97     56596
   Basidirond       0.95      0.97      0.96      7266
    Dark tree       0.92      0.87      0.90       563
 Hangman tree       0.95      0.81      0.87      1895
       Kelpie       0.94      0.91      0.92      3436
     Myconid:       0.98      0.95      0.97      4094

     accuracy                           0.96    116203
    macro avg       0.95      0.92      0.94    116203
 weighted avg       0.96      0.96      0.96    116203

trainging and testing times for Random Forest Classifier
Time to train the model in minutes:  8.081877879301707
Time to test the model in minutes:   0.25388328631718954
```

Results of XGBoost Classifier:



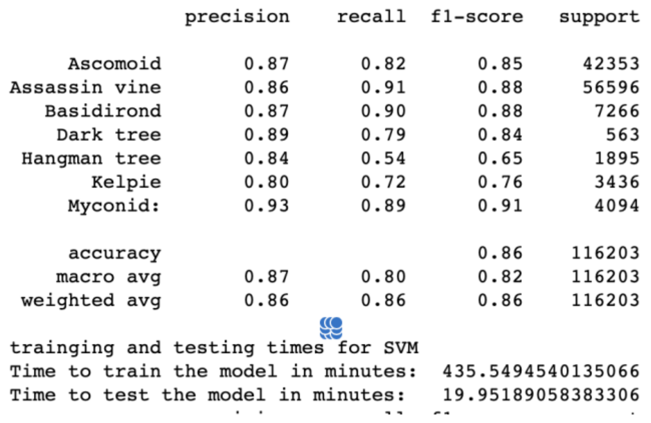
```
               precision    recall  f1-score   support

     Ascomoid       0.98      0.97      0.97     42353
Assassin vine       0.98      0.98      0.98     56596
   Basidirond       0.97      0.97      0.97      7266
    Dark tree       0.91      0.90      0.91       563
 Hangman tree       0.94      0.88      0.91      1895
       Kelpie       0.95      0.94      0.95      3436
     Myconid:       0.98      0.97      0.97      4094

     accuracy                           0.97    116203
    macro avg       0.96      0.94      0.95    116203
 weighted avg       0.97      0.97      0.97    116203

trainging and testing times for XGBoost
Time to train the model in minutes:  154.55835381746292
Time to test the model in minutes:   0.9727433681488037
```

Results of SVM Classifier:



```
               precision    recall  f1-score   support

     Ascomoid       0.87      0.82      0.85     42353
Assassin vine       0.86      0.91      0.88     56596
   Basidirond       0.87      0.90      0.88      7266
    Dark tree       0.89      0.79      0.84       563
 Hangman tree       0.84      0.54      0.65      1895
       Kelpie       0.80      0.72      0.76      3436
     Myconid:       0.93      0.89      0.91      4094

     accuracy                           0.86    116203
    macro avg       0.87      0.80      0.82    116203
 weighted avg       0.86      0.86      0.86    116203

trainging and testing times for SVM
Time to train the model in minutes:  435.5494540135066
Time to test the model in minutes:   19.95189058383306
```

The rest of the sections have been explained in the jupyter Notebook.