NASA ASTEROID CLASSIFICATION PROJECT

Student: Alison Jing March

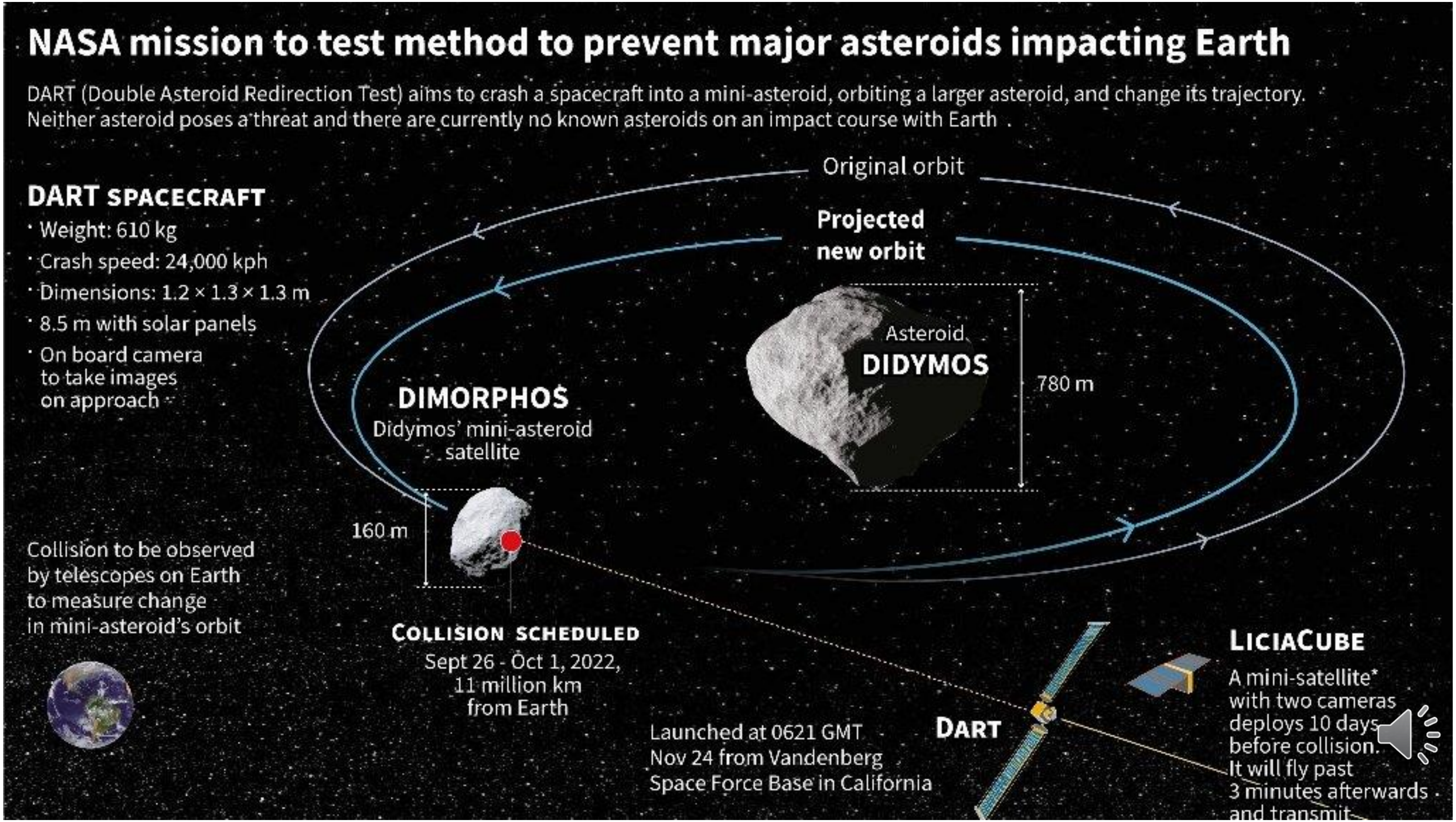
University of Colorado, Boulder

AGENDA

I. Abstract
II. Introduction
III. Related Work
IV. Proposed Work + Project Timeline
V. Evaluation
VI. Discussion
VII. Conclusion

I. Abstract

- The project is about NASA Asteroids classification using its NeoWs(Near Earth Object Web Service) Restful web service for near-Earth Asteroid information. This dataset is directly available on Kaggle(https://www.kaggle.com/datasets/shrutimehta/nasa-asteroids-classification). **The primary goal of analyzing this project is to identify whether an asteroid is potentially hazardous and to classify hazardous and non-hazardous ones.** In addition, we are studying the features that are responsible for identifying the hazardous asteroid, and most importantly summarizing the Data Science methodology and process to accomplish these goals.
- **This project is important because NASA hasn't ruled out the risk of an asteroid collision hitting Earth soon.** It's imperative for NASA to develop a rapid and accurate data monitoring and analysis system such as NEO Observations Program(https://www.nasa.gov/planetarydefense/neoo) for future collision prevention.



Source: https://phys.org/news/2022-09-nasa-deflect-asteroid-key-planetary.html

II. Introduction

NASA ASTEROID CLASSIFICATION

The raw dataset consists of 4,687 rows(data instances) and 40 columns(features). The features are listed below:

1. ***Neo Reference ID***: reference ID assigned to an asteroid.
2. ***Name:*** the name assigned to an asteroid
3. ***Absolute Magnitude***: refers to the magnitude of an asteroid. An absolute magnitude is a visual magnitude an observer would record if the asteroid were placed 1 Astronomical Unit(au) away, and 1 au from the Sun and at a zero-phase angle.
4. ***Est Dia in KM (min)***: refers to the estimated minimum diameter of the asteroid in kilometers(KM)
5. ***Est Dia in KM (max)***: refers to the estimated maximum diameter of the asteroid in kilometers(KM)
6. ***Est Dia in M(min)***: refers to the estimated minimum diameter of the asteroid in meters(M)
7. ***Est Dia in M(max)***: refers to the estimated maximum diameter of the asteroid in meters(M)
8. ***Est Dia in Miles(min)***: refers to the estimated minimum diameter of the asteroid in miles.
9. ***Est Dia in Miles(max)***: refers to the estimated maximum diameter of the asteroid in miles.
10. ***Est Dia in Feet(min):*** refers to the estimated minimum diameter of the asteroid in feet.
11. ***Est Dia in Feet(max)***:
12. ***Close Approach Date***: refers to the actual date the asteroid is approaching Earth
13. ***Epoch Date Close Approach***: refers to an arbitrary fixed instant of time/date used as a chronological reference approaching date of the asteroid to Earth.
14. ***Relative Velocity km per sec:*** is the relative velocity of the asteroid in kilometers per second
15. ***Relative Velocity km per hour***: is the relative velocity of the asteroid in kilometers per hour
16. ***Miles per hour:*** relative velocity of the asteroid in miles per hour
17. ***Miss Dist.(Astronomical)***: the miss distance between an asteroid to Earth in the Astronomical unit(Au)
18. ***Miss Dist.(lunar)***: the miss distance between an asteroid to Earth in lunar distance unit
19. ***Miss Dist.(kilometers)***: the miss distance between an asteroid to Earth in kilometers
20. ***Miss Dist.(miles):*** the miss distance between an asteroid to Earth in miles
21. ***Orbiting Body:*** the planet the asteroid is revolving around is Earth.
22. ***Orbit ID***: the ID # assigned to an asteroid that is revolving around planet Earth.
23. ***Orbit Determination Date:*** the date and time a brand-new asteroid is verified and tracked using the Orbit Determination process.
24. ***Orbit Uncertainty***: a number on a scale from 0-9 to label the orbit uncertainty of an asteroid.
25. ***Minimum Orbit Intersection:*** the distance between the closest points of the orbit of the asteroid and the Earth's orbit.
26. ***Jupiter Tisserand Invariant***: this feature denotes the Tisserand's parameter for the asteroid. Tisserand's parameter is a value calculated from several orbital elements (semi-major axis, orbital
27. ***Epoch Osculation***: refers to an asteroid's orbiting osculating time in Epoch time. Judging from the number of digits in decimal format, this could be in Julian Day with fractional day format.
28. ***Eccentricity***: refers to the value of eccentricity of the asteroid's orbit. It measures "how circular" an orbit is. A perfect circle has an eccentricity of 0 and an ellipse has an eccentricity approaching but never reaches 1. Eccentricity does not have units.
29. ***Semi-Major Axis:*** refers to the semi-major axis of the asteroid's elliptical realm. The semi-major axis is measured in astronomical units. One AU is equal to Earth's average distance from the Sun.
30. ***Inclination:*** measures the angle between the plane of an asteroid's orbit and the plane of Earth's orbit around the Sun.
31. ***Asc Node Longitude:*** refers to an Asteroid's ascending node longitude, this is an angle in the ecliptic plane between the inertial-frame x-axis and the line through the ascending node.
32. ***Orbital Period:*** the value of the orbital period of an asteroid, aka the time taken by the asteroid to make one full revolution around its orbiting body.
33. ***Perihelion Distance***: refers to the value of the Perihelion distance of the asteroid. Perihelion means the object's closest distance to the Sun measured in AU.
34. ***Perihelion Arg:*** refers to the argument of the perihelion of the asteroid. It is an angle in the orbit plane between the ascending node and the perihelion point.
35. ***Aphelion Dist:*** refers to the value of the Aphelion distance of the asteroid. Aphelion is the object's farthest distance from the Sun measure in AU.
36. ***Perihelion Time:*** the time it takes for the asteroid to reach the closest distance to the sun.
37. ***Mean Anomaly***: refers to the fraction of an Asteroid's elliptical orbit's period that has elapsed since the orbiting body passed periapsis; it is expressed as an angle that can be used in calculating the position of the asteroid.
38. ***Mean Motion:*** refers to the angular speed required for an asteroid to complete one orbit,
39. ***Equinox:*** all data entries use J2000 which refers to the Julian date 2451545.0 in the concerned time system. In the Gregorian calendar, this is 2000/1/1 12:00:00
40. ***Hazardous:*** denotes whether the asteroid is hazardous or not.

In [20]:
```
1 nasa.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4687 entries, 0 to 4686
Data columns (total 40 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Neo Reference ID              4687 non-null   int64
 1   Name                          4687 non-null   int64
 2   Absolute Magnitude            4687 non-null   float64
 3   Est Dia in KM(min)            4687 non-null   float64
 4   Est Dia in KM(max)            4687 non-null   float64
 5   Est Dia in M(min)             4687 non-null   float64
 6   Est Dia in M(max)             4687 non-null   float64
 7   Est Dia in Miles(min)         4687 non-null   float64
 8   Est Dia in Miles(max)         4687 non-null   float64
 9   Est Dia in Feet(min)          4687 non-null   float64
 10  Est Dia in Feet(max)          4687 non-null   float64
 11  Close Approach Date           4687 non-null   object
 12  Epoch Date Close Approach     4687 non-null   int64
 13  Relative Velocity km per sec  4687 non-null   float64
 14  Relative Velocity km per hr   4687 non-null   float64
 15  Miles per hour                4687 non-null   float64
 16  Miss Dist.(Astronomical)      4687 non-null   float64
 17  Miss Dist.(lunar)             4687 non-null   float64
 18  Miss Dist.(kilometers)        4687 non-null   float64
 19  Miss Dist.(miles)             4687 non-null   float64
 20  Orbiting Body                 4687 non-null   object
 21  Orbit ID                      4687 non-null   int64
 22  Orbit Determination Date      4687 non-null   object
 23  Orbit Uncertainity            4687 non-null   int64
 24  Minimum Orbit Intersection    4687 non-null   float64
 25  Jupiter Tisserand Invariant   4687 non-null   float64
 26  Epoch Osculation              4687 non-null   float64
 27  Eccentricity                  4687 non-null   float64
 28  Semi Major Axis               4687 non-null   float64
 29  Inclination                   4687 non-null   float64
 30  Asc Node Longitude            4687 non-null   float64
 31  Orbital Period                4687 non-null   float64
 32  Perihelion Distance           4687 non-null   float64
 33  Perihelion Arg                4687 non-null   float64
 34  Aphelion Dist                 4687 non-null   float64
 35  Perihelion Time               4687 non-null   float64
 36  Mean Anomaly                  4687 non-null   float64
 37  Mean Motion                   4687 non-null   float64
 38  Equinox                       4687 non-null   object
 39  Hazardous                     4687 non-null   bool
dtypes: bool(1), float64(30), int64(5), object(4)
memory usage: 1.4+ MB
```

III. Related Work

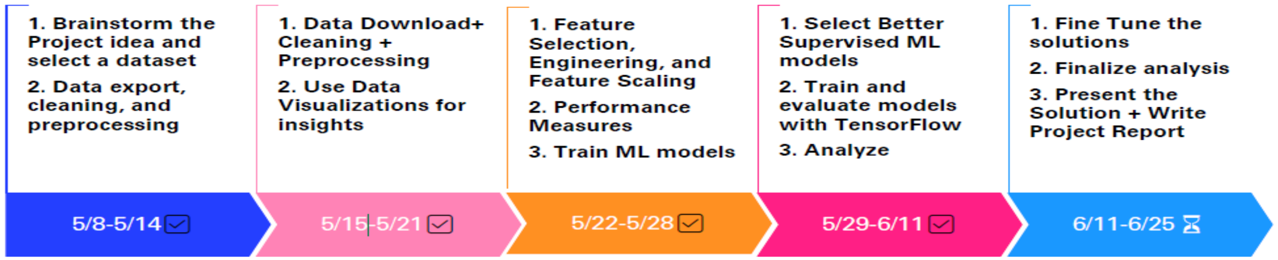
The following steps have been implemented in Jupyter notebook:

1. Data Cleaning + Preprocessing
   - Check data format + missing values
   - Data Visualization to gain insights
   - Use Heatmap to identify features correlations + outliers(see slide # 8)
   - Use the ***OneHotEncoder*** method to transform categorical data features(see highlighted features)

IV. Proposed Work + Project Timeline

Project Checklist:

1. Frame the problem and look at the big picture ☑
2. Get the data ☑
3. Explore the data to gain insights ☑
4. Prepare the data to better expose the underlying data patterns to Machine Learning algorithms ☑
5. Explore many different models and shortlist the best ones. ☑
6. Fine-tune the models and combine them into a great solution
7. Explore solutions using Tensorflow* ☑
8. Present the solutions ☑
9. Launch, monitor, and maintain the system ☑



References:

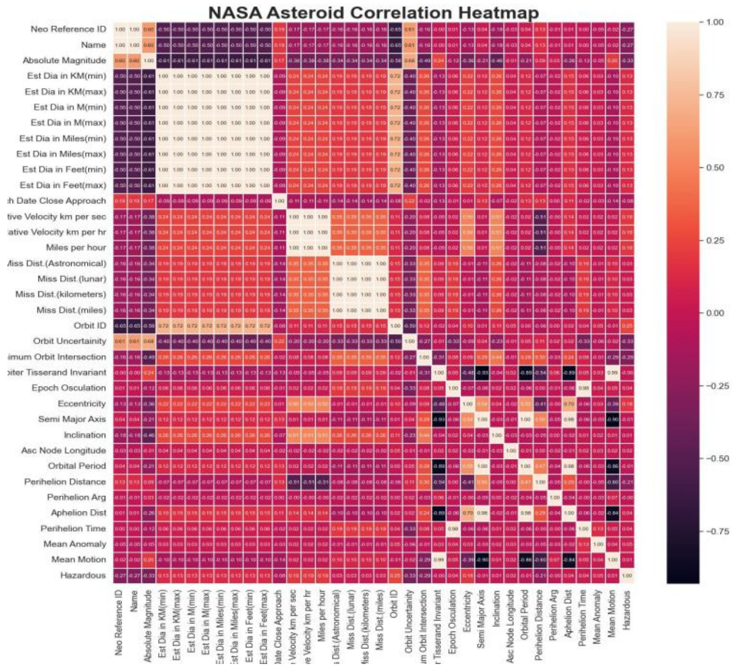
Aurelien Geron. 2019. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd. ed.). O'Reilly Media, Inc.

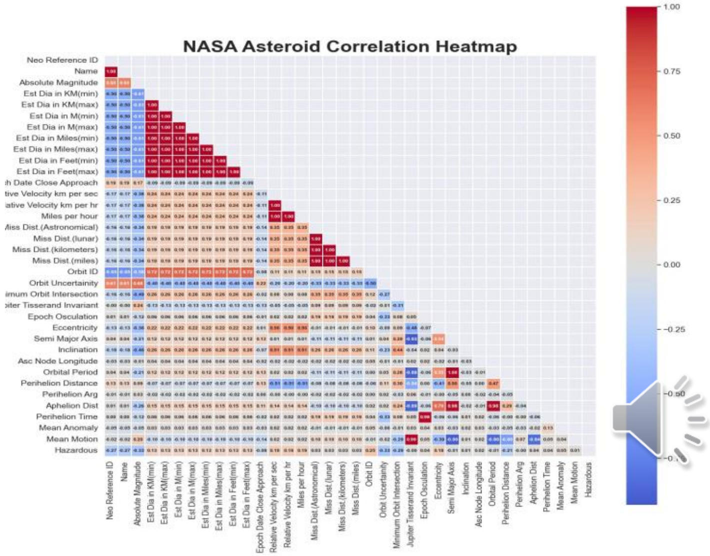
IV. Proposed Work

1. Data Cleaning + Preprocessing ☑
   - Check data format + missing values
   - Data Visualization to gain insights
   - Use Heatmap to identify features correlations + outliers
   - Use the ***OneHotEncoder*** method to transform categorical data features

2. Dimensionality Reduction ☑
   - Feature selection
     - Drop features with correlation > 90% to prevent multicollinearity
     - Drop Features like ‘Orbiting Body’ and ‘Equinox’
   - Feature Scaling
     - Use **Min-max scaling(normalization) method**: Scikit-Learn **MinMaxScaler** transformer
   - Check and Resolve the Imbalance

3. Supervised Machine Learning Models ☑
4. Fine-Tune the System ☑
5. Present the Solution + Write Project Report ☑
6. Launch the Solution for Production ⌛
   - Deploy Final Code + Documentation on Github ☑
   - Deploy model with Heroku using Flask ⌛





IV. Related Work

Supervised Learning ML Models:

1. Logistic Regression
2. Decision Trees
3. Support Vector Machines
4. Random Forest
5. XGBoosting
6. KNN

…

V. Evaluations

1. Cross-Validation
   - K-fold cross-validation with three folds(tentatively)
2. Confusion Matrix
   - Precision: TP/(TP+FP)
   - Recall: TP/(TP+FN)
   - Specificity: TN/(TN+FP)
   - F1 score: TP/(TP+ (FN+FP)/2)
3. Precision/Recall Trade-off
4. The ROC Curve
5. Calibration Plots

IV. Related Work

Data Preprocessing + Feature Engineering

1. Verified there is no missing data
2. Data transformation + normalization
3. Reduced data features from 40 to 19(18 predictor and 1 response features)
4. Used SMOTE+ENN method to resolve data imbalance(from 83.89%:16.11% → 53%:47%).

5. Feature Importance ranking using XGBoost

Completed training Supervised Learning ML Models:

1. Logistic Regression
2. Decision Trees
3. Support Vector Machines
4. Random Forest
5. XGBoosting
6. K-Nearest Neighbor

IV. Related Work Examples

Data Cleaning + Dimension Reduction

```
# Check current data features in new encoded NASA dataset:
encoded_nasa.columns
```

```
Index(['Neo Reference ID', 'Name', 'Absolute Magnitude', 'Est Dia in KM(min)',
       'Est Dia in KM(max)', 'Est Dia in M(min)', 'Est Dia in M(max)',
       'Est Dia in Miles(min)', 'Est Dia in Miles(max)',
       'Est Dia in Feet(min)', 'Est Dia in Feet(max)', 'Close Approach Date',
       'Epoch Date Close Approach', 'Relative Velocity km per sec',
       'Relative Velocity km per hr', 'Miles per hour',
       'Miss Dist.(Astronomical)', 'Miss Dist.(lunar)',
       'Miss Dist.(kilometers)', 'Miss Dist.(miles)', 'Orbit ID',
       'Orbit Determination Date', 'Orbit Uncertainity',
       'Minimum Orbit Intersection', 'Jupiter Tisserand Invariant',
       'Epoch Osculation', 'Eccentricity', 'Semi Major Axis', 'Inclination',
       'Asc Node Longitude', 'Orbital Period', 'Perihelion Distance',
       'Perihelion Arg', 'Aphelion Dist', 'Perihelion Time', 'Mean Anomaly',
       'Mean Motion', 'Hazardous', 'Orbiting Body_Earth', 'Equinox_J2000'],
      dtype='object')
```

```
len(encoded_nasa.columns) # retain the equal number of data features as the original dataset
```

```
40
```

```
cleaned_features.columns
```

```
Index(['Name', 'Absolute Magnitude', 'Epoch Date Close Approach', 'Orbit ID',
       'Orbit Uncertainity', 'Minimum Orbit Intersection',
       'Jupiter Tisserand Invariant', 'Eccentricity', 'Inclination',
       'Asc Node Longitude', 'Perihelion Distance', 'Perihelion Arg',
       'Mean Anomaly', 'Hazardous'],
      dtype='object')
```

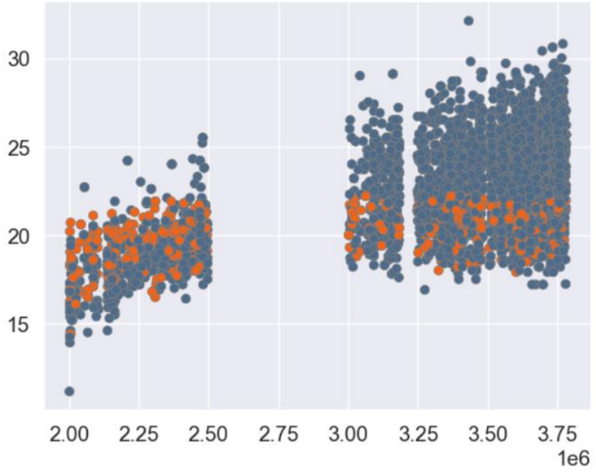
```
len(cleaned_features.columns)
```

```
14
```

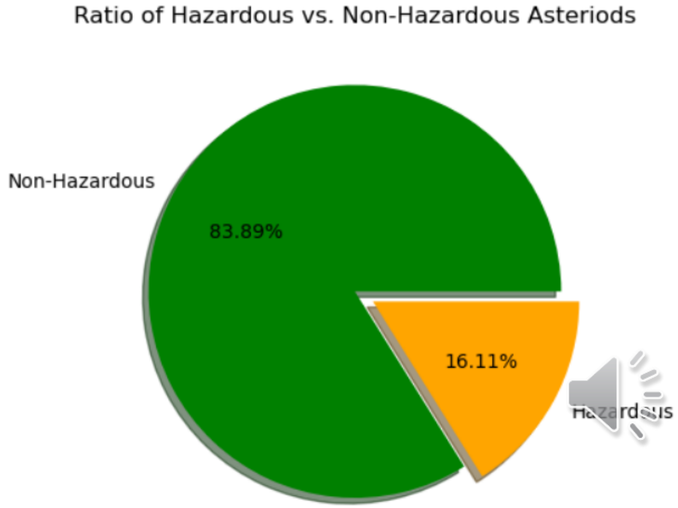
```
False    3932
True      755
Name: Hazardous, dtype: int64
```

Dataset Balancing

```
Before:
Class=1, Count=755 (16.11%)
Class=0, Count=3932 (83.89%)
```

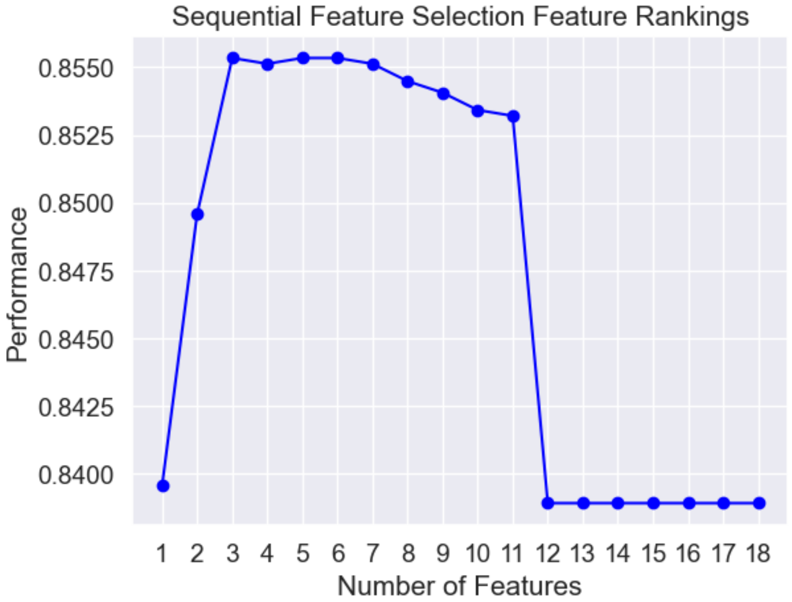



```
After:
Class=0, Count=1857 (53.00%)
Class=1, Count=1647 (47.00%)

Text(0.5, 1.0, 'SMOTE+EEN')
```



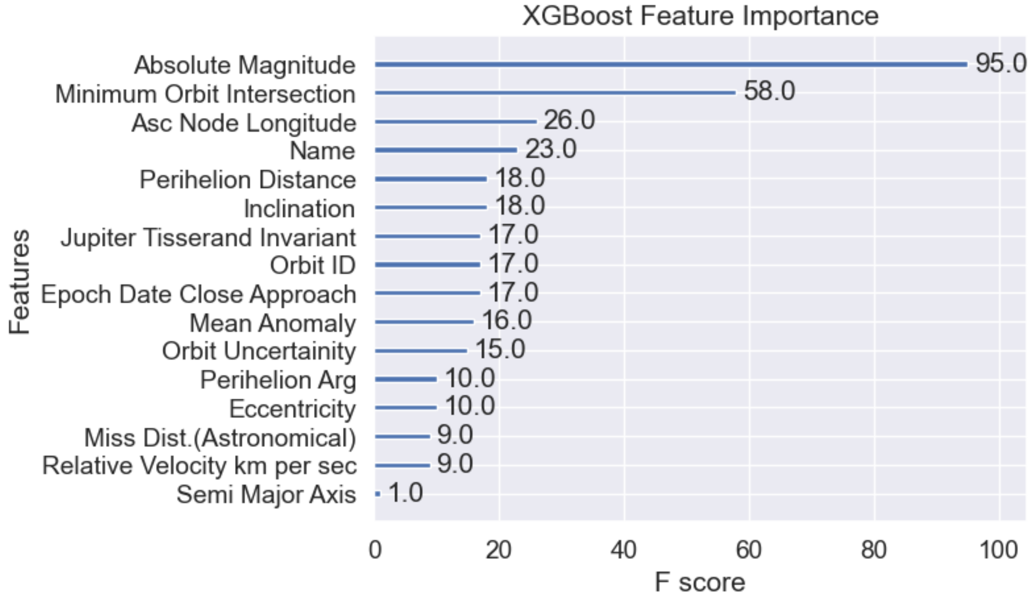
IV. Related Work Examples Continued…

(Backend) Sequential Feature Selection Algorithm




```
features = nasa[['Name', 'Absolute Magnitude', 'Est Dia in M(min)',
       'Epoch Date Close Approach', 'Relative Velocity km per sec',
       'Miss Dist.(Astronomical)', 'Orbit ID', 'Orbit Uncertainity',
       'Minimum Orbit Intersection', 'Jupiter Tisserand Invariant',
       'Epoch Osculation', 'Eccentricity', 'Semi Major Axis', 'Inclination',
       'Asc Node Longitude', 'Perihelion Distance', 'Perihelion Arg',
       'Mean Anomaly']]

target= nasa['Hazardous']
```

XGBoost Feature Importance



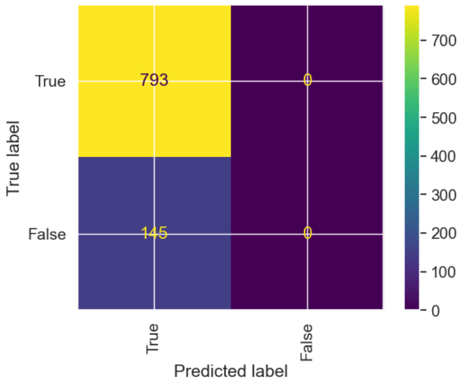

```
Accuracy: 100.00%
Thresh=0.000, n=18, Accuracy: 100.00%
Thresh=0.000, n=18, Accuracy: 100.00%
Thresh=0.001, n=16, Accuracy: 100.00%
Thresh=0.003, n=15, Accuracy: 100.00%
Thresh=0.004, n=14, Accuracy: 100.00%
Thresh=0.004, n=13, Accuracy: 100.00%
Thresh=0.004, n=12, Accuracy: 100.00%
Thresh=0.005, n=11, Accuracy: 100.00%
Thresh=0.005, n=10, Accuracy: 100.00%
Thresh=0.006, n=9, Accuracy: 100.00%
Thresh=0.007, n=8, Accuracy: 100.00%
Thresh=0.008, n=7, Accuracy: 100.00%
Thresh=0.008, n=6, Accuracy: 99.89%
Thresh=0.008, n=5, Accuracy: 100.00%
Thresh=0.013, n=4, Accuracy: 100.00%
Thresh=0.018, n=3, Accuracy: 100.00%
Thresh=0.254, n=2, Accuracy: 100.00%
Thresh=0.651, n=1, Accuracy: 81.56%
```

V. Proposed Work

1. Logistic Regression Model

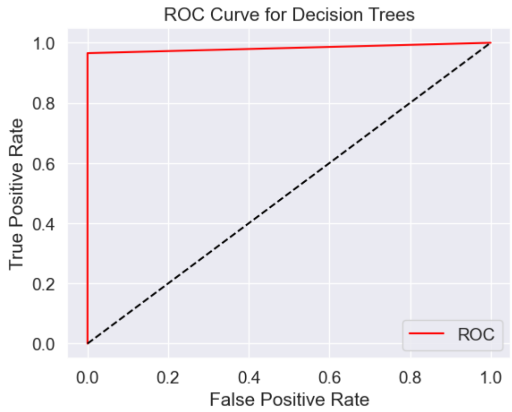
Accuracy: 84.54157783%

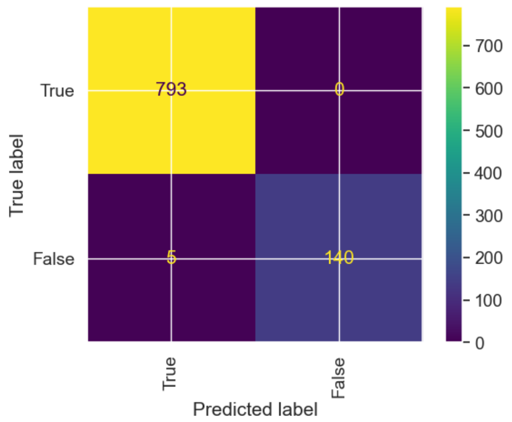
	precision	recall	f1-score	support
False	0.85	1.00	0.92	793
True	0.00	0.00	0.00	145
accuracy			0.85	938
macro avg	0.42	0.50	0.46	938
weighted avg	0.71	0.85	0.77	938

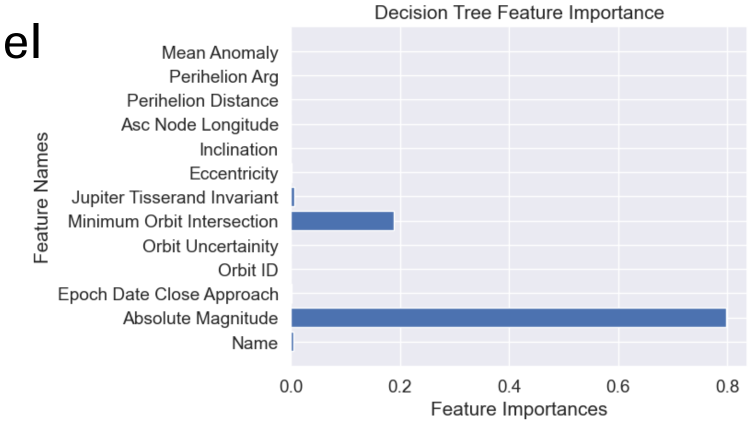


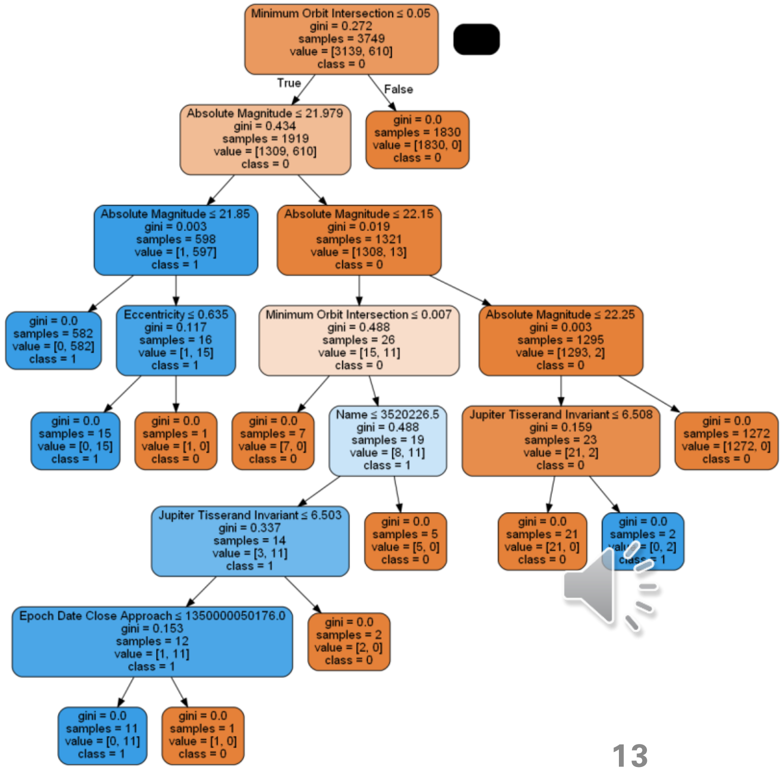
2. Decision Tree Model

Accuracy: 99.46695096%





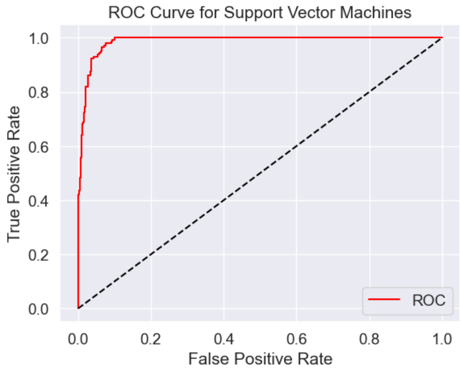


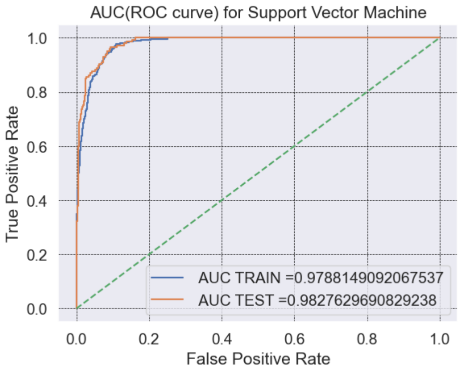


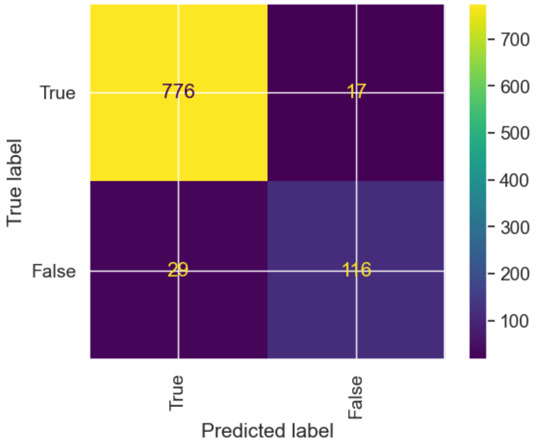
V. Proposed Work continued..

3. Support Vector Machines Model

`Accuracy: 95.09594883%`

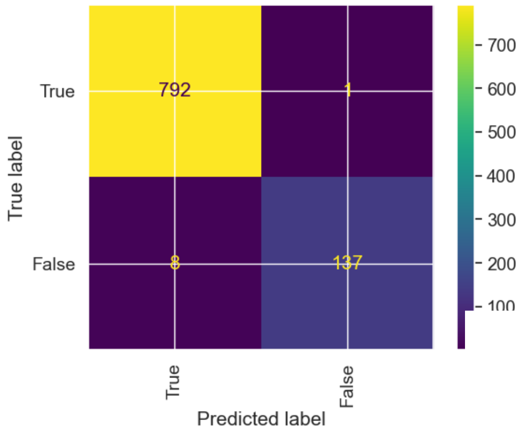


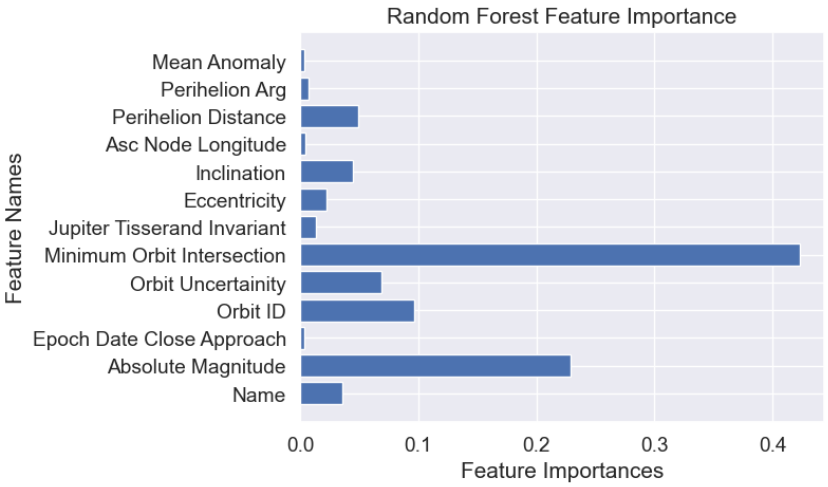


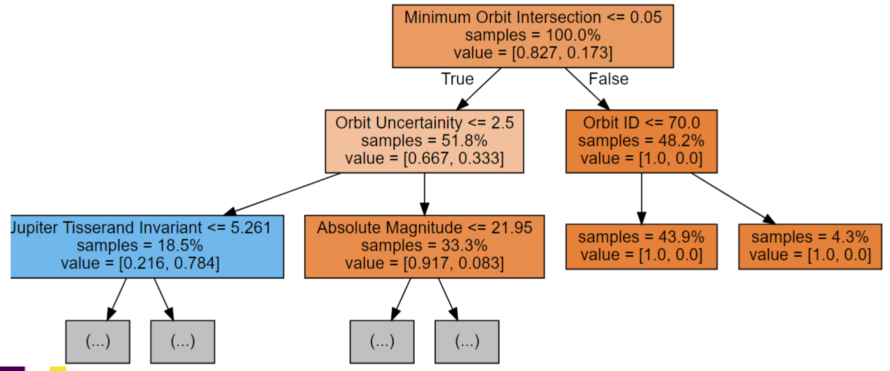


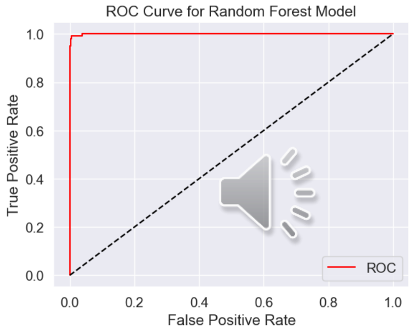
4. Random Forest Model

`Accuracy: 99.04051173%`



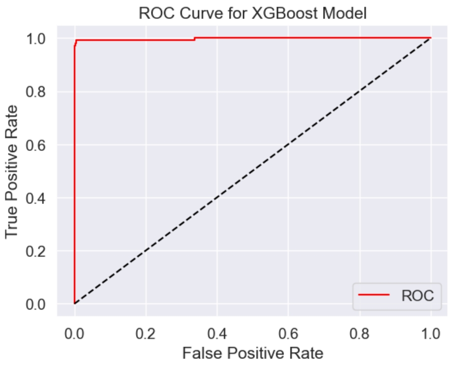


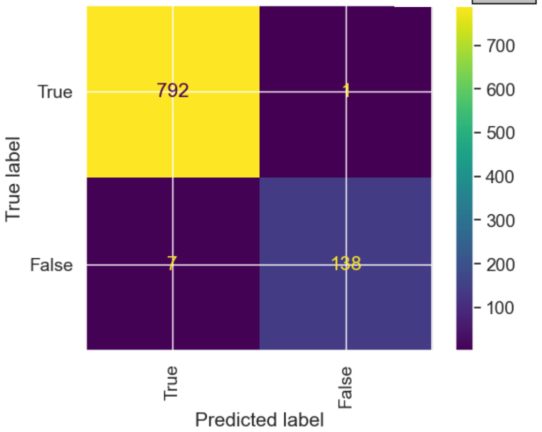




5. XGBoost Model

`Accuracy: 99.14712154%`

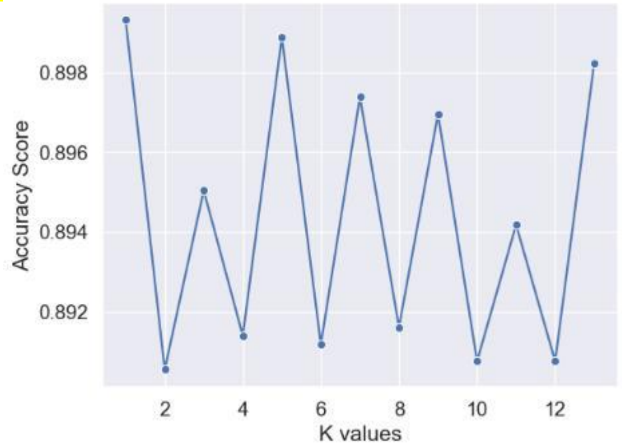


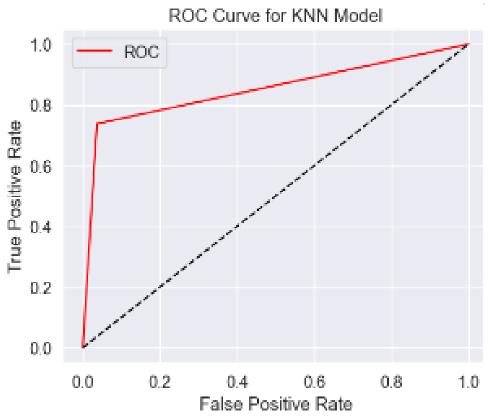


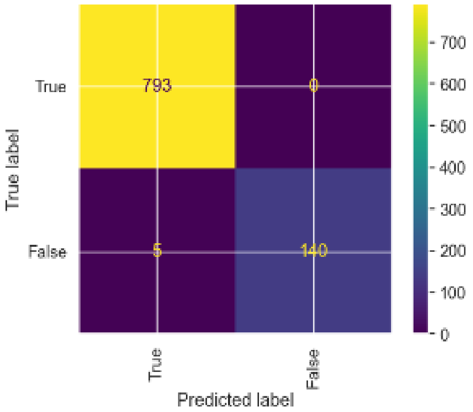
V. Proposed Work Continued…

6. KNN Model

Cross-validation to find the best k value







Accuracy: 92.85714286%

7. TensorFlow Sequential Neural Network

```python
# Create a neural network model using tensorflow

# of features in cleaned_features2 = 14 --> 14 neurons input layer
# 9 neurons in layer 2: hidden layer
# 1 neuron as the last layer --> output layer

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(128, input_dim = len(X_train[0,:]), activation = 'relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1792

 dense_1 (Dense)             (None, 128)               16512

 dense_2 (Dense)             (None, 1)                 129

=================================================================
Total params: 18,433
Trainable params: 18,433
Non-trainable params: 0
```

```python
# Check if there are any cycles in the Sequential Model
model.compile(loss = 'binary_crossentropy', optimizer ='rmsprop', metrics =['accuracy'])
```

Train the model

- We feed X_train into the model and the model calculates errors using y_train
- In one epoch the model scans through the entire rows in the X_train
- Updating the number of epochs usually increases the accuracy of the model
- To observe the accuracy on the testing dataset during the training, add validation_data = (X_test, y_test)

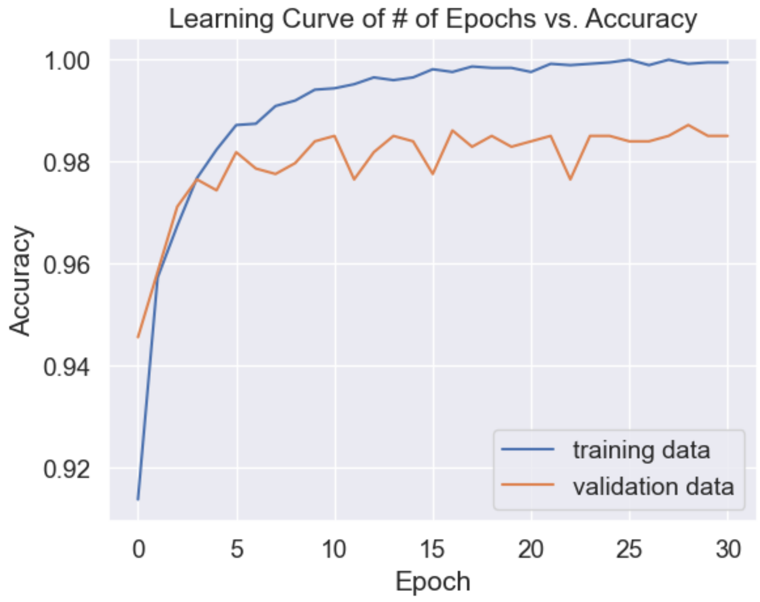
```python
from keras.callbacks import EarlyStopping, ModelCheckpoint
callback_a = ModelCheckpoint(filepath ='my_best_model.gif', monitor ='val_loss', save_best_only=True, save_weights_only=True
callback_b = EarlyStopping(monitor ='val_loss', mode ='min', patience=20, verbose=1)
```

```python
print(history.params)
```

```
{'verbose': 1, 'epochs': 256, 'steps': 375}
```

V. Proposed Work Continued…

6. TensorFlow Sequential Neural Network Continued



Is 'accuracy sufficient to evaluate our model

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

accuracy = accuracy_score(y_test, prediction.round())
precision = precision_score(y_test, prediction.round())
recall = recall_score(y_test, prediction.round())
f1_score = f1_score(y_test, prediction.round())

print("Accuracy: %.2f%%" % (accuracy * 100.0))
print("Precision: %.2f%%" % (precision *100.0))
print("Recall: %.2f%%" % (recall * 100.0))
print("F1 Score: %.2f%%" % (f1_score * 100.0))
```

```
Accuracy: 98.51%
Precision: 95.80%
Recall: 94.48%
F1 Score: 95.14%
```

```
print(prediction[0:10].round())
```

```
4034    False
3607    False
3933    False
1736     True
1533    False
2899    False
1892    False
3375    False
2048    False
4331    False
Name: Hazardous, dtype: bool
```

```
[[0.]
 [0.]
 [0.]
 [1.]
 [0.]
 [0.]
 [0.]
 [0.]
 [0.]
 [0.]]
```

Evaluate the model on the training data

```
scores = model.evaluate(X_train, y_train)
print(model.metrics_names)
print(scores)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

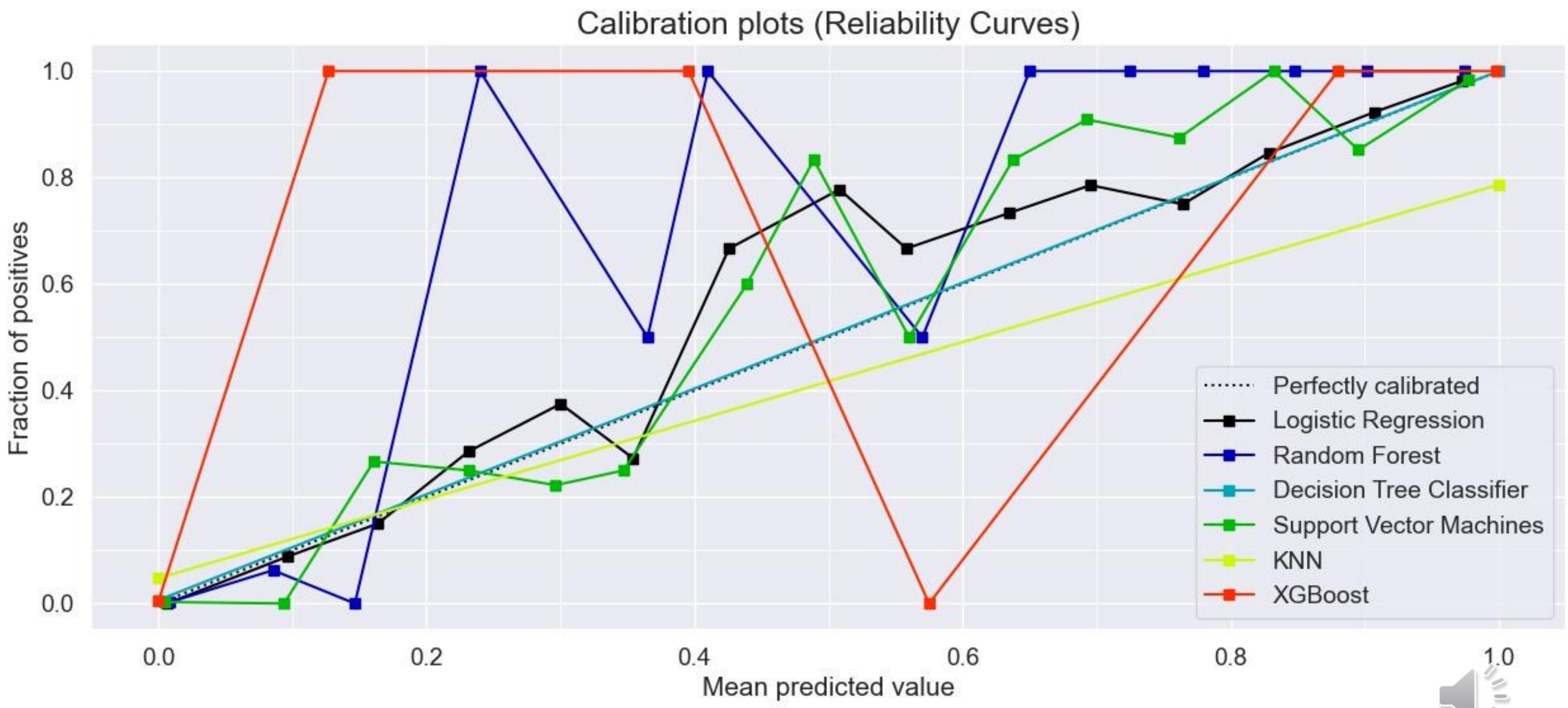
```
118/118 [==============================] - 0s 2ms/step - loss: 0.0123 - accuracy: 0.9947
['loss', 'accuracy']
[0.012318885885179043, 0.994665265083313]

accuracy: 99.47%
```

Evaluate the model on the testing data

```
scores = model.evaluate(X_test, y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

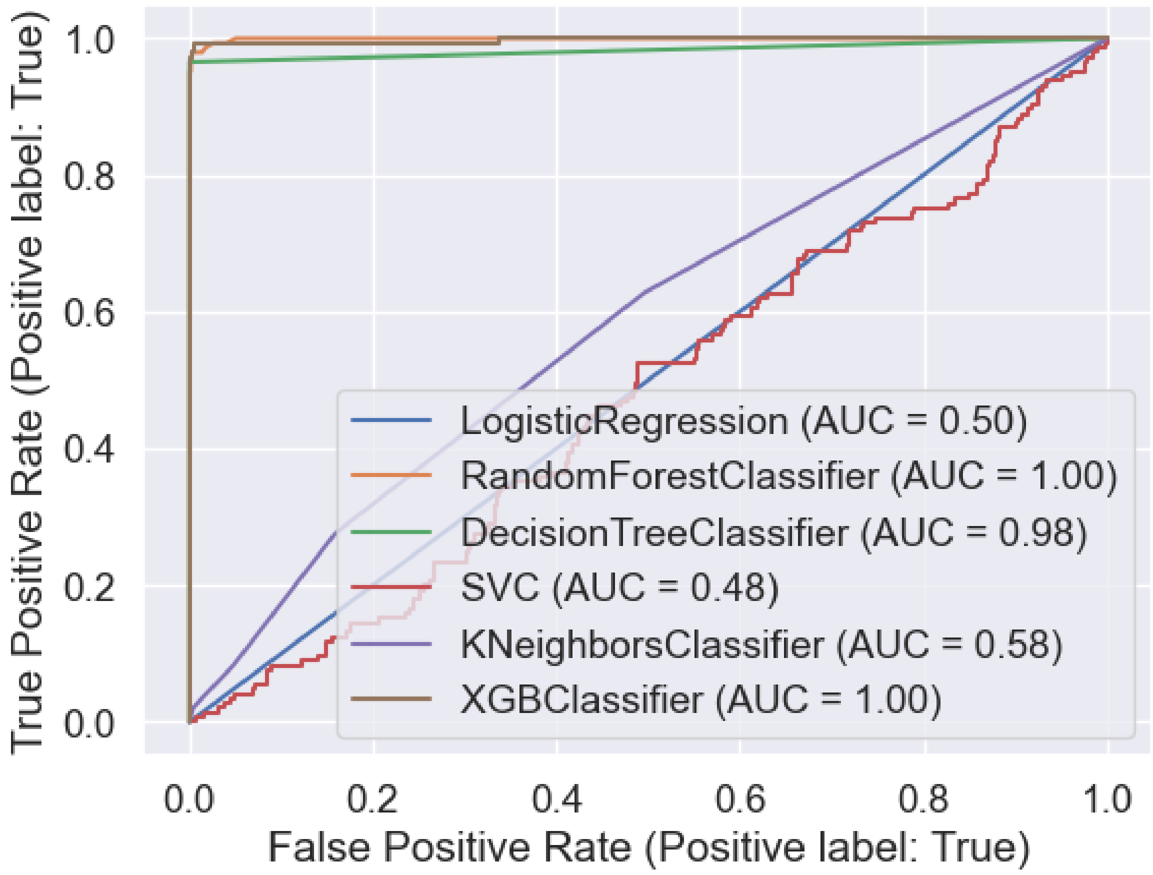
```
30/30 [==============================] - 0s 1ms/step - loss: 0.0511 - accuracy: 0.9851

accuracy: 98.51%
```

VI. Evaluation



VI. Discussion

Machine Learning Models	Accuracy	ROC AUC Score
1. Logistic Regression	84.54% before SMOTEEN, 53.5% After	54.96%
2. Random Forest	99.04%	99.96%
3. Decision Tree	99.47%	98.28%
4. Support Vector Machine	95.10%	98.69%
5. KNN	92.86%	83.72%
6. XGBoost	99.15%	99.76%



VII. Conclusion

We have determined that three machine learning models: Random Forest, XGBoost, and Decision Tree along with Neural Network be more effective models to classify asteroids. Coincidentally all three models have a Features Importance function to help us identify the most relevant feature variables- particularly using XGBClassifer feature importance, with just the top two features ***Absolute Magnitude*** and **Minimum Orbit Intersection**, we can predict 100% accuracy.

Random Forest and XGBoost stand out as the winners as both their accuracy score and ROC-AUC reached close to 100%, however, the Decision Tree model is the most perfectly calibrated model which suggests it might be a more reliable model than the other two. Finally, the Sequential Neural Network trained and tested using TensorFlow also shows similar results as the previously mentioned three models, effectively concluding our project analysis.

THANK YOU