5/6/22

REPORT: CS 4375 FINAL PROJECT

1. DETAILS

Problem Type: Classification

Goal: Classifying whether an asteroid is hazardous to life on Earth using features such as diameter, closest approach distance, velocity, orbit eccentricity, etc.

Notebook Link:
https://colab.research.google.com/drive/1pUFG54gJt2b6jcYQ0b5QO2gUGBvRzww1?usp=sharing

2. MOTIVATION

I chose this problem because of my interest in the astronomy domain. Ever since I can remember, I have always had an interest in planets, galaxies, and outer space in general. So, I researched on Kaggle to find a classification problem related to astronomy. I chose classification because of the multiplicity of models I had learnt about in class as opposed to other types of models like regression. Therefore, with classification, I can apply a lot of machine learning models in order to better understand the nature of the problem and also to better understand more models in the process. I eventually landed on an Asteroid Threat dataset suited for binary classification.

3. APPROACH

To get started right away, just tap any placeholder text (such as this) and start typing.

A. Data Preparation

i. Libraries: Pandas, NumPy, Seaborn, Matplotlib, Sklearn, Keras, Corner, and Tabulate

ii. The dataset, found on Kaggle, was originally collected by NASA. The dataset contains 16 columns and 15635 rows. The label is either true or false, true meaning an asteroid is hazardous.

iii. Import Data (CSV → Pandas DataFrame)

iv. Check dimensions

v. Remove NULL values

B. Data Analysis

i. Label counts (to determine class imbalance).
ii. Correlation matrix (Seaborn Heatmap) and corner plot (Corner) to visualize relationships between features.
iii. Basic statistical quantities such as mean, median, standard deviation and skew to better understand spread of the data.

C. Data Preprocessing

i. Splitting data into training and testing datasets using Sklearn's train_test_split
ii. Normalizing data using Sklearn's StandardScaler

D. Models

i. K-Nearest Neighbors (with/without normalization)
ii. Decision Tree
iii. Naïve Bayes
iv. Logistic Regression
v. Neural Network

E. Metrics

i. Accuracy
ii. Precision
iii. Recall

4. RESULTS

A. Data Preparation

i. First, columns such as Object Name and Classification had were dropped because they were unnecessary. Next, there was only one row which had a NULL value in it, so that row was dropped.

B. Data Analysis

i. In this dataset, there is a class imbalance. Only 11% of the labels were true. One consequence of this is that accuracy will not be a good metric, thus other metrics are required to gauge the performance of different models.
ii. The correlation heatmap and corner plot showed very few features having a direct correlation with the target label (Hazardous). These included Minimum Orbit Intersection Distance (AU), Asteroid Magnitude, and Perihelion Distance (AU). Only these features will be used to train the models in the next steps in order to save time, computing power, and reduce potential noise.
iii. Basic statistical quantities such as mean, median, standard deviation and skew showed that these measurements were far apart from one another (since units are different). The standard deviation showed that Asteroid Magnitude had greater difference in values compared to the other features

C. Data Preprocessing

i. For splitting data into training and testing sets, I chose 80% train and 20% test. I think this is a good split since it uses most of the data to train, but still leaves a significant portion to test with.

ii. Sklearn's StandardScaler normalizes the data between 0 and 1, which eliminates all the different scales and ranges which the data had before. But, as we'll see later on, this did not have huge of an impact as I expected.

D. Models

i. K-Nearest Neighbors (with/without normalization)

1. I ran KNN algorithm for K values 1-200 and found that the best trade-off between train accuracy and test accuracy was about K= 50-75. I also found out that KNN without normalization worked slightly better than KNN with normalization. In the end, with K=60, accuracy was around 0.89, precision was around 0.5, and recall was around 0.006. Although accuracy was high, the model did not actually perform that well as seen by the confusion matrix. In the matrix, the true positive rate (where the actual value and predicted value was true/threat) is very low. Instead, the KNN algorithm classified 338/340 hazardous asteroids as non-hazardous, thus leading to a high false negative value, which is very bad. This is most likely because hazardous and non-hazardous asteroids are very similar to each other, as seen by the pairplot in data analysis. Therefore, measuring Euclidean distance through KNN algorithm is not the best way to classify hazardous asteroids. Another reason that KNN is not the optimal solution is the time and space it takes to run this algorithm. Since distances between all data points must be calculated and stored, this algorithm runs for a long time on multi-dimensional data like this. In fact, Google Colab takes over 5-7 minutes just to run this algorithm.

ii. Decision Tree

1. This algorithm produced the most interesting results. With a depth limit of 2, the decision tree was able to get 100% accuracy. And with the random forest classifier, it was able to get 100% accuracy, precision, and recall. And it only used 2 features to be able to narrow down to true/false or hazardous/non-hazardous asteroid. How is this possible? The answer to this question is still being explored, but one likely possibility is that this data has some sort of inherent relations within it that resulted in this behavior. Either it was simplified in some way by NASA or it just had some of the most correlated features possible.

iii. Naïve Bayes

1. This model was very simple but effective. Accuracy was around 96%, precision was around 86%, and recall was around 75%. Looking at the

confusion matrix, one can also see that its true positive and true negative rate is very high. The number of misclassifications with regard to the class imbalance is surprisingly low. Overall, Naïve Bayes is one of, if not the best performing model for this asteroids dataset.

iv. Logistic Regression

1. I implemented logistic regression with and without normalization. But unlike KNN algorithm, normalization actually hurt the performance of this algorithm. Precision and recall both dropped to 0 in my testing for normalized data, while unnormalized data produced 79% and 55% for precision and recall, respectively. I also plotted the ROC curve (True Positive Rate vs False Positive Rate) for this model, and the area under the curve was equal to 0.975. This value is really good because it shows that this binary classifier was largely successful in distinguishing between the two classes.

v. Neural Network

1. My neural network (consisting of 3 8-node hidden layers) produced promising results, but the results it gives have lot of variability. The accuracy stays around 90%, but the precision and recall vary from 40% all the way to 80%. This is because each training process has a lot of computations, and each matrix computation can lead to different weights and biases being produced.

E. Metrics

i. Accuracy- As seen in models like K-Nearest Neighbor, accuracy is not good metric for imbalanced classes. This is because the high accuracies produced by most of the models given above are due to the fact that many instances were false (no threat), so the model randomly choosing one class can lead to a higher accuracy since most of the true labels are actually false (no threat).

ii. Precision- This metric evaluates how many of the true predictions are actually true in the dataset. This is a better metric because it focuses on the imbalanced class (true/threat). Therefore, it has a narrower criterion for successful models, making it a better metric to follow for imbalanced data and binary classification.

iii. Recall- This metric evaluates how many of the true positives were identified correctly by the models. Correctly identifying the hazardous asteroids is very important and useful for making these models applicable in the real world, and recall is great way to find which models are good on imbalanced datasets.

iv. Bias- Almost all the models which were normalized led to some form of oversimplification of the data. This resulted in a model with high bias and the classification of all asteroids as false (not a threat). This likely happened because the normalization eliminated all the important information hidden in the original distribution.

v. Variance- One model with high variance is the neural network model because it had a lot of variability in the training process. Sometimes it generalized well, but other times, the neural network failed to capture the essence of the training data and produced results which have high variance and high bias.

5. LESSONS

A. Data is not as widely available as you think

i. The process of finding a problem on which to do a project was a lot of work. It took me several hours to find a domain which interested me and a dataset which addressed the problem I was looking for.

B. Data is not clean

i. There are always missing values, confusing features, and hidden errors which can only be found out through data manipulation, analysis and testing.

C. Some results don't make sense

i. I'm still not sure why decision trees returned 100% accuracy. This will forever poke my curiosity. I will continue to figure out why this is the case.

D. Accuracy is not always a good metric

i. I was surprised when all my models returned 85%+ accuracy. Only when I learnt about other metrics in class and found out that imbalanced data can result to a misleading accuracy, I realized some of my models were not actually good.

E. Don't underestimate simple models

i. The simplest models like Logistic Regression and Naïve Bayes led to the best results on my dataset. Sometimes overcomplicating a problem by throwing a fully connected neural network with 50 hidden layers is not always the best idea. I, in fact, couldn't wait to implement neural network because I thought it would give me the best results, but after seeing how it did, I went back and implemented Logistic Regression and Naïve Bayes, only to realize that they performed better and more consistently.

6. RESOURCES

1. *https://www.kaggle.com/shrushtijoshi/asteroid-impacts*
2. *https://www.kaggle.com/nasa/asteroid-impacts*
3. *https://pandas.pydata.org/docs/index.html*
4. *https://seaborn.pydata.org/tutorial.html*
5. *https://matplotlib.org/stable/api/index*
6. *https://www.javatpoint.com/classification-algorithm-in-machine-learning*