Classification

Dillon Carter

02/18

The data for this notebook was provided from here (https://www.kaggle.com/datasets/sakhawat18/asteroid-dataset).

Logistic Classification Algorithm Logisitic classification is about taking a set of predictors and figuring out a qualitative value from those set of predictors. Using the log odds of the predictors set to a logistic curve, we give each set of predictors a location on the curve. Then, the curve is split and the values that lay to one side are classified as one qualitative value and the other side as another qualitative value.

```
if(!require('tidyverse')){
  install.packages('tidyverse')
}
```

```
## Loading required package: tidyverse
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.1     ✓ purrr   1.0.1
## ✓ tibble  3.1.8     ✓ dplyr   1.1.0
## ✓ tidyr   1.3.0     ✓ stringr 1.5.0
## ✓ readr   2.1.4     ✓ forcats 1.0.0
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
```

```
if(!require('e1071')){
  install.packages('e1071')
}
```

```
## Loading required package: e1071
```

```
if(!require('caret')){
  install.packages('caret')
}
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library('caret')
library('tidyverse')
library('e1071')
data <- read.csv("data/asteroid_dataset.csv", header=TRUE)
set.seed(02222001)
str(data)
```

```
## 'data.frame':    958524 obs. of  45 variables:
##  $ id            : chr  "a0000001" "a0000002" "a0000003" "a0000004" ...
##  $ spkid         : int  2000001 2000002 2000003 2000004 2000005 2000006 2000007 2000008 20
00009 2000010 ...
##  $ full_name     : chr  "     1 Ceres" "     2 Pallas" "     3 Juno" "     4 Vesta" ...
##  $ pdes          : chr  "1" "2" "3" "4" ...
##  $ name          : chr  "Ceres" "Pallas" "Juno" "Vesta" ...
##  $ prefix        : chr  "" "" "" "" ...
##  $ neo           : chr  "N" "N" "N" "N" ...
##  $ pha           : chr  "N" "N" "N" "N" ...
##  $ H             : num  3.4 4.2 5.33 3 6.9 5.8 5.6 6.5 6.3 5.5 ...
##  $ diameter      : num  939 545 247 525 107 ...
##  $ albedo        : num  0.09 0.101 0.214 0.423 0.274 ...
##  $ diameter_sigma: num  0.2 18 10.59 0.2 3.14 ...
##  $ orbit_id      : chr  "JPL 47" "JPL 37" "JPL 112" "JPL 35" ...
##  $ epoch         : num  2458601 2459001 2459001 2458601 2459001 ...
##  $ epoch_mjd     : int  58600 59000 59000 58600 59000 59000 59000 59000 59000 59000 ...
##  $ epoch_cal     : num  20190427 20200531 20200531 20190427 20200531 ...
##  $ equinox       : chr  "J2000" "J2000" "J2000" "J2000" ...
##  $ e             : num  0.076 0.23 0.2569 0.0887 0.1909 ...
##  $ a             : num  2.77 2.77 2.67 2.36 2.57 ...
##  $ q             : num  2.56 2.14 1.98 2.15 2.08 ...
##  $ i             : num  10.59 34.83 12.99 7.14 5.37 ...
##  $ om            : num  80.3 173 169.9 103.8 141.6 ...
##  $ w             : num  73.6 310.2 248.1 150.7 358.6 ...
##  $ ma            : num  77.4 145 125.4 95.9 17.8 ...
##  $ ad            : num  2.98 3.41 3.35 2.57 3.07 ...
##  $ n             : num  0.214 0.213 0.226 0.272 0.239 ...
##  $ tp            : num  2458239 2458321 2458446 2458248 2458926 ...
##  $ tp_cal        : num  20180430 20180721 20181123 20180509 20200317 ...
##  $ per           : num  1683 1687 1592 1325 1508 ...
##  $ per_y         : num  4.61 4.62 4.36 3.63 4.13 ...
##  $ moid          : num  1.59 1.23 1.03 1.14 1.1 ...
##  $ moid_ld       : num  621 480 403 443 426 ...
##  $ sigma_e       : num  4.82e-12 3.19e-08 3.05e-08 2.33e-10 2.37e-08 ...
##  $ sigma_a       : num  1.03e-11 4.03e-09 3.47e-09 1.51e-09 3.97e-09 ...
##  $ sigma_q       : num  1.96e-11 8.83e-08 8.14e-08 1.93e-09 6.09e-08 ...
##  $ sigma_i       : num  4.61e-09 3.47e-06 3.22e-06 2.17e-07 2.74e-06 ...
##  $ sigma_om      : num  6.17e-08 6.27e-06 1.66e-05 3.88e-07 2.89e-05 ...
##  $ sigma_w       : num  6.62e-08 9.13e-06 1.77e-05 1.79e-07 2.98e-05 ...
##  $ sigma_ma      : num  7.82e-09 8.86e-06 8.11e-06 1.21e-06 8.30e-06 ...
##  $ sigma_ad      : num  1.11e-11 4.96e-09 4.36e-09 1.65e-09 4.73e-09 ...
##  $ sigma_n       : num  1.20e-12 4.65e-10 4.41e-10 2.61e-10 5.52e-10 ...
##  $ sigma_tp      : num  3.78e-08 4.08e-05 3.53e-05 4.10e-06 3.47e-05 ...
##  $ sigma_per     : num  9.42e-09 3.68e-06 3.11e-06 1.27e-06 3.49e-06 ...
##  $ class         : chr  "MBA" "MBA" "MBA" "MBA" ...
##  $ rms           : num  0.433 0.359 0.338 0.4 0.522 ...
```

This model will be used to classify whether an asteroid is a Potentially Hazardous Asteroid or not. To do this, the predictors that matter are the NEO flag, the diameter, eccentricity, and the inclination. The NEO flag is a factor that is either “Y” or “N”. The diameter is a numeric list, e (eccentricinty) is a numeric list, inclination (i) is a numeric list.

```
extraneous <- c("")
data <- data[-c(which(data$neo %in% extraneous))]
data <- na.omit(data)
```

```
data <- data[,c(7,8,10,18,21)]
data$neo <- factor(data$neo)
data$pha <- factor(data$pha)
data$eccentricity <- data$e
data$inclination <- data$i
data <- data[,c(1,2,3,6,7)]
str(data)
```

```
## 'data.frame':    131142 obs. of  5 variables:
##  $ neo         : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 1 1 ...
##  $ pha         : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 1 1 ...
##  $ diameter    : num  939 545 247 525 107 ...
##  $ eccentricity: num  0.076 0.23 0.2569 0.0887 0.1909 ...
##  $ inclination : num  10.59 34.83 12.99 7.14 5.37 ...
```

```
#Separate training and test data
temp <- sample(1:nrow(data), 0.8*nrow(data), replace=FALSE)
train <- data[temp,]
test <- data[-temp,]

attach(train)
```

With all the data cleaning and restructuring done, let’s examine the data. First looking at the summaries of diameter, inclination, and eccentricity.

```
d_sum <- summary(diameter)
e_sum <- summary(eccentricity)
i_sum <- summary(inclination)

print("Diameter Summary")
```

```
## [1] "Diameter Summary"
```

```
d_sum
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.008   2.854   4.063   5.583   5.866 900.000
```

```
print("Eccentricity Summary")
```

```
## [1] "Eccentricity Summary"
```

```
e_sum
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
## 0.0003095 0.0903514 0.1393085 0.1455507 0.1911261 0.9837890
```

```
print("Inclination Summary")
```

```
## [1] "Inclination Summary"
```

```
i_sum
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
##   0.02206   4.96113   9.12670  10.06273  13.39833 158.55655
```

```
sd(eccentricity)
```

```
## [1] 0.07637316
```

```
sd(inclination)
```

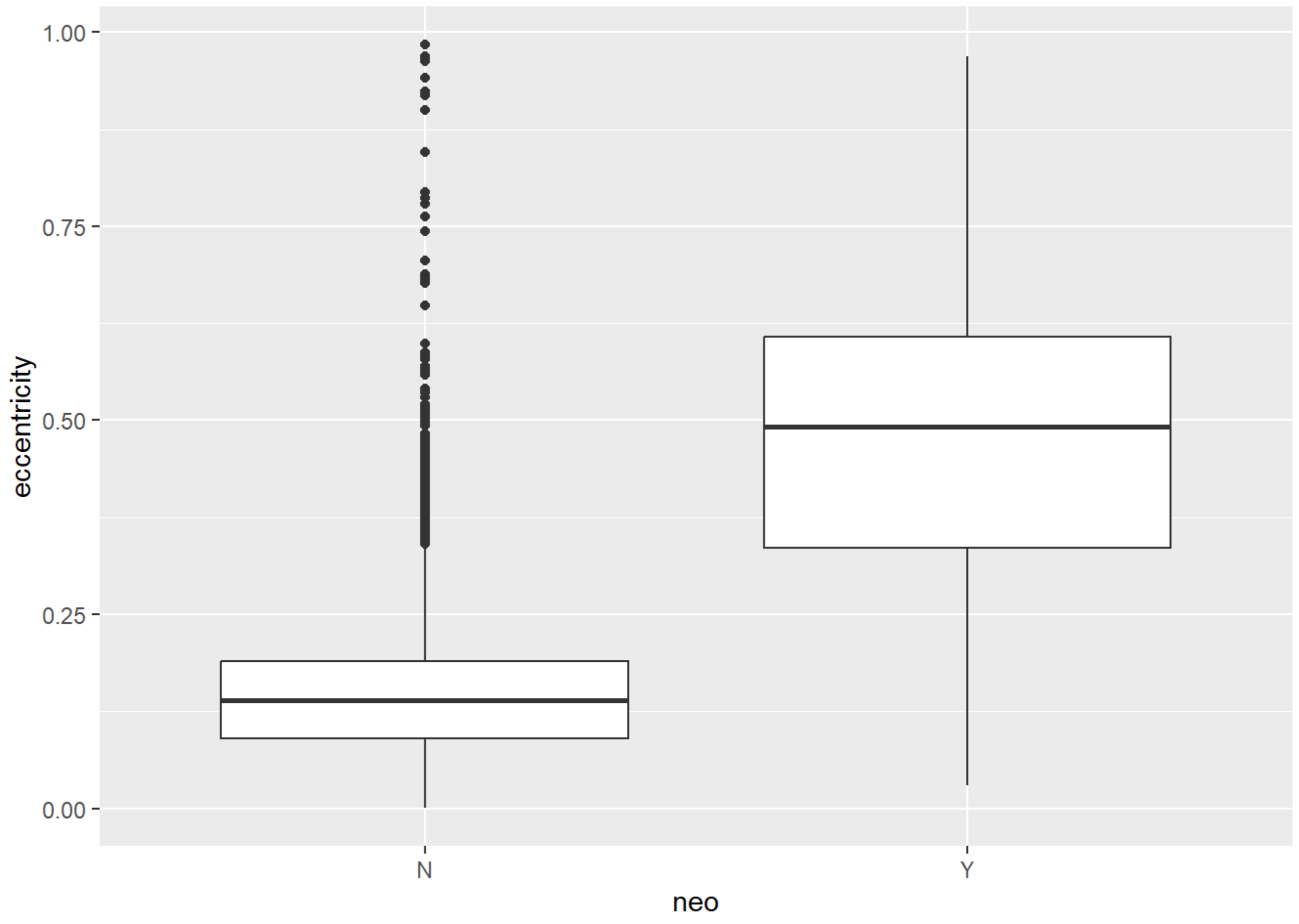
```
## [1] 6.676265
```

Looking at this, a few things are apparent. First, the diameter of most objects is quite small, less than 10 km. There are how,ever larger ones up to a greatest size of 900km. For reference, the Earth has a radius of 6400 km. The eccentricities of the asteroids, which is the deviation of an object from a circular object falls roughly around .10 to .20 with some objects up to .98. The smaller and closer to 0, the closer the object is to a circular orbit and less likely to colide with other objects in other orbits. The closer to 1 the closer to a parabolic trajectory. If it is 1 or greater, than gravity is unable to keep the object contained and will gain enough velocity to exit the gravity well. Finally, inclination describes the tilt at which an object orbits the sun. It describes the angle between it and a reference of Earth’s orbit. Most objects have pretty acute inclinations. This makes sense as most of the solar system orbits on a roughly equal plane. But there are some outliers with much alrger inclinations. The key to determining the likelihood of an object colliding with Earth will be based off how ecentric the orbit is and its inclination mostly. At least, that is my hypothesis. So let’s go about testing it.

```
cor(inclination, eccentricity)
```

```
## [1] 0.1601188
```

```
ggplot(train, aes(x=neo, y=eccentricity)) +
  geom_boxplot(notch=FALSE)
```



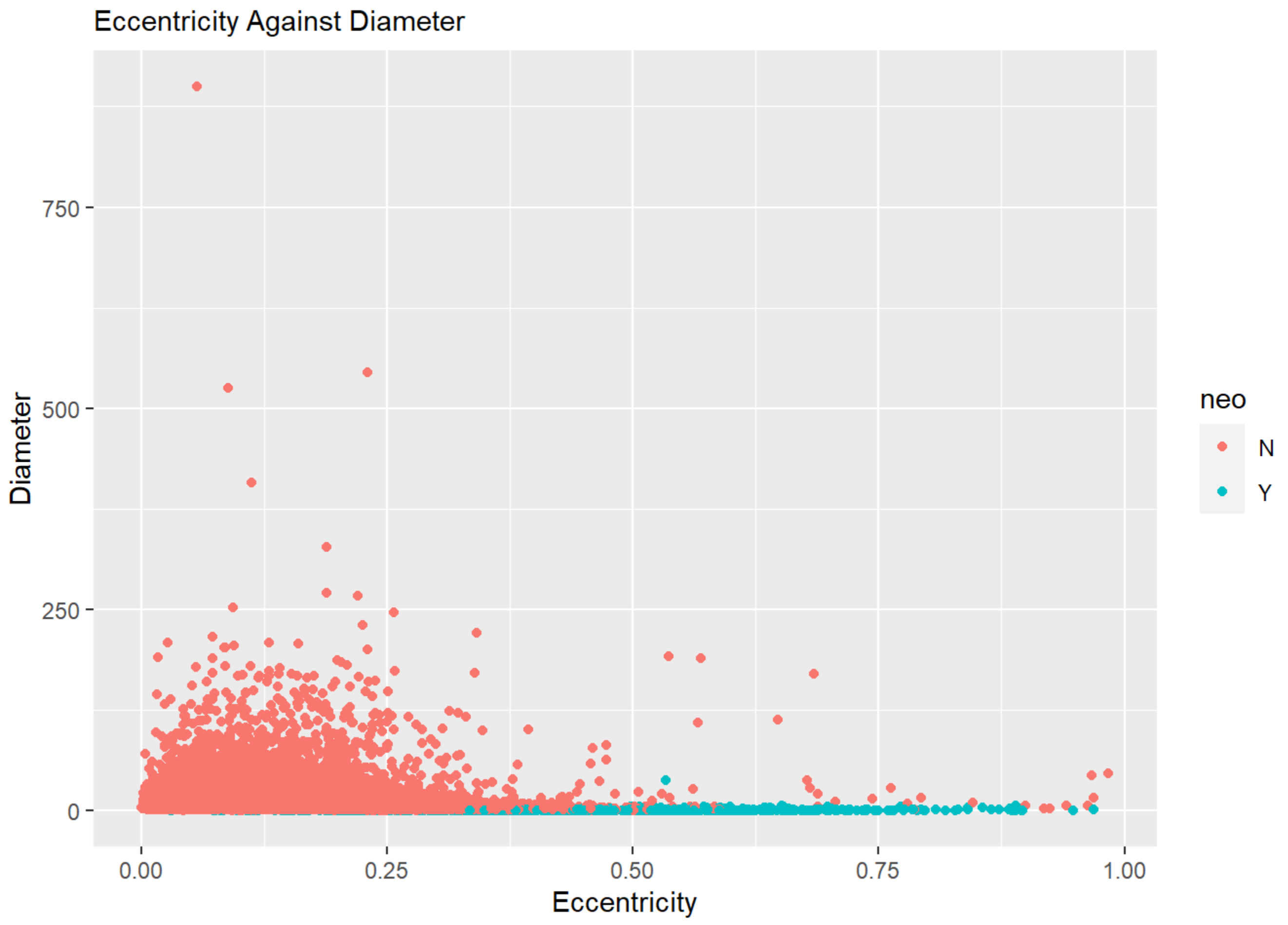
```
length(which(neo=="Y"))
```

```
## [1] 586
```

Looking at the correlation between inclination and eccentricity, there is some there. But, for the most part, objects that orbit at different angles to Earth do not necessarily indicate a more ovular orbit. The distance to Earth is somewhat useful in determining eccentricity. The data set is only 586 objects which is minuscule compared to the number of objects outside of near-Earth orbit. So when looking at the box and whisker plot, the difference between min and max is almost nothing. There is a difference between the means and quartiles however. Objects that are closer to Earth are more likely to have more eccentric orbits than objects further from Earth. This does make sense. As Earth is closer to the sun, objects that have more eccentric orbits are going to come closer to Earth than those without eccentric orbits.

As I think that more eccentric objects are more dangerous, maybe it would be beneficial to take a look at the diameter of those objects compared to the eccentricity as well. After all, a tiny object that could intersect with Earth doesn't pose as much of a risk as some of those larger objects.

```
gg <- ggplot(train, aes(x=eccentricity, y=diameter)) +
  geom_point(aes(col=neo)) +
  labs(subtitle="Eccentricity Against Diameter", y = "Diameter", x="Eccentricity")
plot(gg)
```



This seems to say that as the asteroids get larger, the eccentricity tends to get smaller. So likely, the larger objects have more stable orbits.

#Building Logistic Regression Model#

```
glm1 <- glm(pha~., data=train, family="binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm1)
```

```
##
## Call:
## glm(formula = pha ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.070   0.000   0.000   0.000   2.345
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -24.600233 661.007249  -0.037 0.970313
## neoY          24.564798 661.007269   0.037 0.970355
## diameter      -0.111517   0.099475  -1.121 0.262261
## eccentricity  -0.965113   0.564743  -1.709 0.087462 .
## inclination   -0.032862   0.008736  -3.762 0.000169 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2080.14  on 104912  degrees of freedom
## Residual deviance:  611.39  on 104908  degrees of freedom
## AIC: 621.39
##
## Number of Fisher Scoring iterations: 24
```

The model seems to be an okay model for predicting a threat to Earth. Most of the predictors are not great for the model with the exception of inclination. Even with that, there is a large drop between the Null deviance and the residual deviance. Given that the lack of fit drops by including more of the predictors, it can be safe to assume that more predictors gives a greater certainty that the object will fit the log odds. The AIC isn't that low which tells that there is quite a bit of deviance between different tuples. It might not be the most accurate of models in that case. Now to compare with the Naive Bayes.

```
nb1 <- naiveBayes(pha~., data=train)
nb1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##           N           Y
## 0.998703688 0.001296312
##
## Conditional probabilities:
##    neo
## Y             N           Y
##   N 0.995705164 0.004294836
##   Y 0.000000000 1.000000000
##
##    diameter
## Y        [,1]     [,2]
##   N 5.5890179 9.026784
##   Y 0.8369485 0.925778
##
##    eccentricity
## Y        [,1]       [,2]
##   N 0.1451519 0.07533105
##   Y 0.4528172 0.18258894
##
##    inclination
## Y       [,1]      [,2]
##   N 10.05529  6.661429
##   Y 15.79577 12.858040
```

The Naive Bayes algorithm takes the assumption that the variables are all independent from one another. It takes this assumption to find the probability an event occurs, the posterior, given its likelihood and prior over the marginal. The model above is finding that after going through each asteroid, the probability of it being a PHA is either yes or no. Given all that, the model thinks that the probability of any object being a hazard is .001296. The strongest indicator seems to be NEO as it has the least number of false positives and true negatives. Inclination seems somewhat untrustworthy as does the rate at which diameter gives false positives.

#Predicting and Evaluating#

```
probs <- predict(glm1, newdata=test, type="response")
p1 <- predict(nb1, newdata=test, type="class")
pred <- ifelse(probs>0.2, "Y", "N")
acc <- mean(pred==test$pha)
print(paste("accuracy = ", acc))
```

```
## [1] "accuracy =  0.997140569598536"
```

```
table(pred, test$pha)
```

```
##
## pred     N     Y
##    N 26118     9
##    Y    66    36
```

```
table(p1, test$pha)
```

```
##
## p1      N     Y
##   N 25953     2
##   Y   231    43
```

```
confusionMatrix(as.factor(pred), reference=test$pha)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     N     Y
##          N 26118     9
##          Y    66    36
##
##                Accuracy : 0.9971
##                  95% CI : (0.9964, 0.9978)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.4886
##
##  Mcnemar's Test P-Value : 1.004e-10
##
##             Sensitivity : 0.9975
##             Specificity : 0.8000
##          Pos Pred Value : 0.9997
##          Neg Pred Value : 0.3529
##              Prevalence : 0.9983
##          Detection Rate : 0.9958
##    Detection Prevalence : 0.9961
##       Balanced Accuracy : 0.8987
##
##        'Positive' Class : N
##
```

```
confusionMatrix(as.factor(p1), reference=test$pha)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     N     Y
##          N 25953     2
##          Y   231    43
##
##                Accuracy : 0.9911
##                  95% CI : (0.9899, 0.9922)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2674
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9912
##             Specificity : 0.9556
##          Pos Pred Value : 0.9999
##          Neg Pred Value : 0.1569
##              Prevalence : 0.9983
##          Detection Rate : 0.9895
##    Detection Prevalence : 0.9896
##       Balanced Accuracy : 0.9734
##
##        'Positive' Class : N
##
```

The logistic regression required a low probability to test for if an asteroid was a threat. Going greater than even 50% on dividing the logistic curve caused it to falsly label every asteroid as a non-threat. Set at a .2, it missed 9 instances of should be PHA and 66 instance of shouldn't. Compared to the Naive Bayes, which missed 2 true negatives and 231 false positives. The Naive Bayes is a bit more cautious of a model, in this circumstance, as it has a lower accuracy but will flag objects as potential hazards more often. The logistic regression is more likely to flag a true positive than naive bayes as the sensitivity is higher but is also less likely to correctly flag a true negative with a lower specificity rate. When each is adjusted for Kappa, the Bayes algorithm is much less likely to correctly flag than the logistic regression. The reason for why is that Bayes likely had a lower threshold for considering an asteroid as a threat. It flagged many more objects as threats than logistic regression did. So, by chance, when one did happen to be a threat, it did correctly flag it. But, the borderline threats that are not classified as such put it off.

#Strengths, Weaknesses, Benefits, and Drawbacks of the Algorithms# Naive Bayes is a great method for looking at smaller datasets. It isn't an efficient algorithm so it takes time to run but can be extremely accurate when accounting for those smaller datasets. Once the datasets start to grow in size to sizes like this one, it becomes far less accurate. Its assumption that every predictor is independent can backfire as larger datasets can highlight some correlations between them. Logistic regression is better for these types of datasets. Because it makes use of gradient descent to create the logistic curve, it can be inefficient to run it for smaller datasets. But larger ones will take much more time to analyze each predictor in Bayes than through gradient descent. In terms of the metrics used to evaluate the models, accuracy can be the most generally helpful but also the least insightful. It tells just the general rate at which the model correctly attributed the right classification to the data. It can help tell at just a glance whether a model is good or not but will not help when a deeper knowledge of how the model is failing is needed. The confusion matrix, specificity and sensitivity will help there as they describe what a model classified as wrong or right correctly. They can help tune a model that might not care about the rate of true positives when trying to tune the false positives out of the system. For example, in a case of testing for a disease, it would be far better to try to tune the system so that it will never incorrectly identify someone with the disease as without it but it is far less important to classify someone without as with it.