Final Project

CS 750/850 Machine Learning

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Warning: package 'e1071' was built under R version 3.6.3
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

Asteroidz

The problem we are investigating is the best method of machine learning to predict if an asteroid is potentially hazardous based on the standard observations recorded when documenting asteroids. An asteroid can be either potentially hazardous or not potentially hazardous so we will investigate popular classification methods. The data is sourced from the NASA Jet Propulsion Lab's small body database. We are using ~10,000 records for our dataset with ~2,400 classified as potentially hazardous. This is not the total dataset that NASA provides, but is a subset that we determined would be large enough to yield accurate results while still running in an acceptable amount of time.

Evaluation

The metric we will be judging different methods by comparing the true positive rate of each method. The reasoning for this is where we allow the error, we would rather overfit asteroids to classify a non-hazardous asteroid as hazardous rather than misclassify a potentially hazardous asteroid.

Data scrubbing, conforming and organization

Used data migrated to

```r
data <- read.csv('asteroids.csv')
```

```r
set.seed(1)
dataf = data[ , -which(names(data) %in% c("n_del_obs_used","n_dop_obs_used", "full_name"))]
pha.y = dataf %>% filter(pha == "Y")
pha.n = dataf %>% filter(pha == "N")

pha.n = na.omit(pha.n)
pha.n = sample_frac(pha.n, 0.01)
pha.y = na.omit(pha.y)

pha = rbind(pha.y, pha.n)
pha$pha = as.numeric(pha$pha)-2
pha$condition_code = as.numeric(pha$condition_code)
write.csv(pha, "./pha.csv")
```

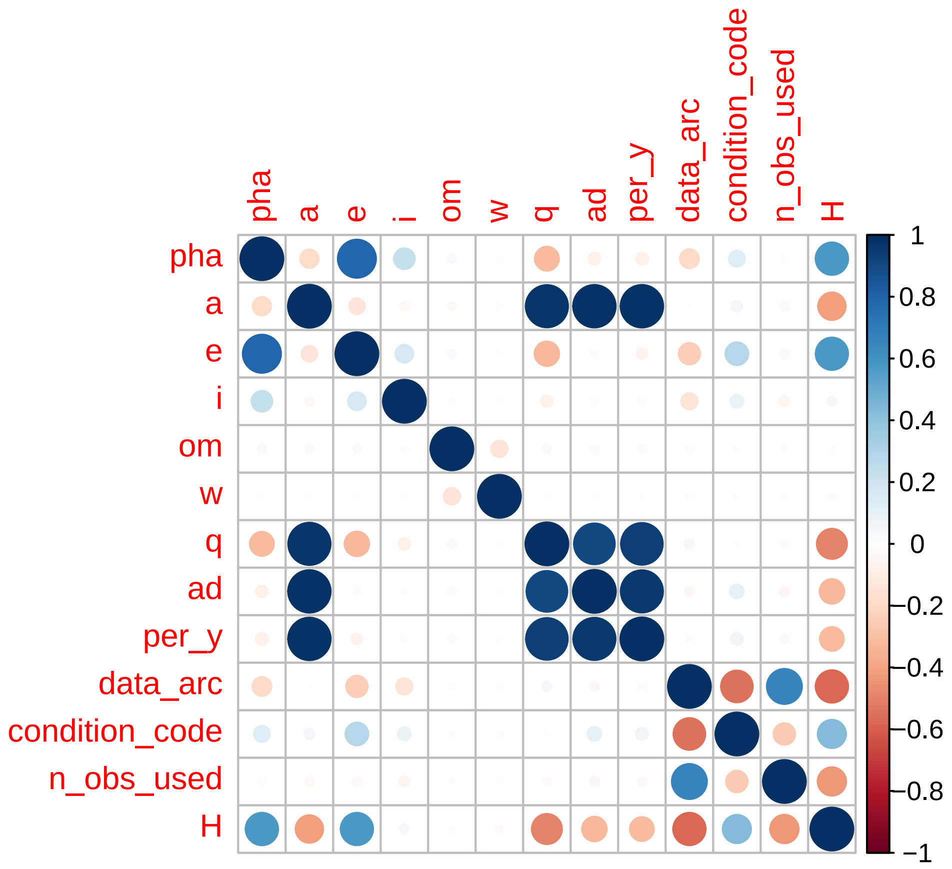
Training and Test Set Generation

```r
pha.test = sample_frac(pha, 0.25)
pha.train = anti_join(pha, pha.test)
```

```
## Joining, by = c("pha", "a", "e", "i", "om", "w", "q", "ad", "per_y", "data_arc", "condition_code", "
```

```r
pha.test = pha.test[, 1:13]
pha.train = pha.train[, 1:13]

correlation = cor(pha.train)
corrplot(correlation)
```



Logistic Regression

```
fit = glm(pha ~ ., data=pha.train, family="binomial")
```

```
## Warning: glm.fit: algorithm did not converge

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

```
pred = predict(fit, newdata=pha.test, type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
mean(round(pred)==pha.test$pha)
```

```
## [1] 0.9805928
```

```
table(round(pred), pha.test$pha, dnn=c('LogisticPredicted', 'True'))
```

```
##                    True
## LogisticPredicted    0    1
##                 0 2273   35
##                 1   20  506
```

Support Vector Machines

```
sv_classifier = svm(pha ~ ., data = pha.train, kernel = "linear", cost = 10, scale = TRUE)
plot(sv_classifier, pha.train)
```

```
plot(sv_classifier, pha.train, fill = TRUE)
```

Quadratic Discriminant Analysis

```
# head(simple_pha.test)
# summary(pha)
# summary(simple_pha.train)

# fit = qda(pha ~ ., data=pha.train)
# pred = predict(fit, newdata=pha.test, type="response")$class
# mean(pred==pha.test$pha)
# table(pred, pha.test$pha, dnn=c('QDAPredicted', 'True'))
```

Linear Discriminant Analysis

```
# head(simple_pha.test)
# summary(pha)
# summary(simple_pha.train)

fit = lda(pha ~ ., data=pha.train)
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

```
pred = predict(fit, newdata=pha.test, type="response")$class
mean(pred==pha.test$pha)
```

```
## [1] 0.9756528
```

```
table(pred, pha.test$pha, dnn=c('LDAPredicted', 'True'))
```

```
##             True
## LDAPredicted    0    1
##            0 2242   18
##            1   51  523
```