FDA data analysis for AstraZeneca interview by Naveen Ahluwalia

```
library("devtools")
```

```
## Warning: package 'devtools' was built under R version 3.5.3
```

```
## Loading required package: usethis
```

```
## Warning: package 'usethis' was built under R version 3.5.3
```

```
devtools::install_github("ropenhealth/openfda")
```

```
## WARNING: Rtools is required to build R packages, but is not currently
installed.
##
## Please download and install Rtools custom from http://cran.r-
project.org/bin/windows/Rtools/.
```

```
## Skipping install of 'openfda' from a github remote, the SHA1 (ace7ef93)
has not changed since last install.
##   Use `force = TRUE` to force installation
```

```
library("openfda")
```

```
library(rjson)
```

```
## Warning: package 'rjson' was built under R version 3.5.2
```

```
json_file <- fda_query("/animalandveterinary/event.json") %>%
  fda_filter("reaction.veddra_term_name", "emesis")  %>%
  fda_limit(100) %>%
  fda_search() %>%
  fda_exec()
```

```
## Fetching:
https://api.fda.gov/animalandveterinary/event.json?search=reaction.veddra_ter
m_name:emesis&limit=100
```

```
#View(json_file)
#https://api.fda.gov/animalandveterinary/event.json?search=reaction.veddra_te
rm_name:"emesis"+AND+animal.species:"Cat"&limit=100
```

From the FDA website, I have downloaded 100 observations related to animal and veterinary event, all of which had a reaction of “emesis”

```
library(jsonlite)
```

```
## Warning: package 'jsonlite' was built under R version 3.5.3
```

```
##
## Attaching package: 'jsonlite'

## The following objects are masked from 'package:rjson':
##
##     fromJSON, toJSON

fdadata <- flatten(json_file)
#View(fdadata)
```

` Since the files are downloaded as nested arrays, the flatten code above is needed to work with individual features.

```
library(lubridate)

## Warning: package 'lubridate' was built under R version 3.5.3

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.5.3

## -- Attaching packages ---------------------------------------------
tidyverse 1.2.1 --

## v ggplot2 3.1.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.3.1
## v readr   1.3.1     v forcats 0.4.0

## Warning: package 'ggplot2' was built under R version 3.5.3

## Warning: package 'tibble' was built under R version 3.5.3

## Warning: package 'tidyr' was built under R version 3.5.3

## Warning: package 'readr' was built under R version 3.5.3

## Warning: package 'purrr' was built under R version 3.5.3

## Warning: package 'dplyr' was built under R version 3.5.3

## Warning: package 'forcats' was built under R version 3.5.3

## -- Conflicts ------------------------------------------------
tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
```

```
## x dplyr::filter()          masks stats::filter()
## x purrr::flatten()         masks jsonlite::flatten()
## x jsonlite::fromJSON()     masks rjson::fromJSON()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x jsonlite::toJSON()       masks rjson::toJSON()
## x lubridate::union()       masks base::union()

library(dplyr)



fdadata$treated_for_ae<-as.factor(fdadata$treated_for_ae)
fdadata= fdadata[!(is.na(fdadata$treated_for_ae) |
fdadata$treated_for_ae==""),]

fdadata= fdadata[!(is.na(fdadata$onset_date) | fdadata$onset_date==""), ]
fdadata$onset_date <- substr(fdadata$onset_date, 5, 6)
fdadata$onset_date<-as.factor(fdadata$onset_date)

fdadata$health_assessment_prior_to_exposure.condition<-
as.factor(fdadata$health_assessment_prior_to_exposure.condition)

fdadata$animal.gender<-as.factor(fdadata$animal.gender)
fdadata= fdadata[!(is.na(fdadata$animal.gender) | fdadata$animal.gender==""),
]

fdadata$animal.species<-as.factor(fdadata$animal.species)

fdadata$animal.reproductive_status<-
as.factor((fdadata$animal.reproductive_status))
fdadata= fdadata[!(is.na(fdadata$animal.reproductive_status) |
fdadata$animal.reproductive_status==""), ]

fdadata$animal.weight.min<-as.numeric(fdadata$animal.weight.min)
fdadata$animal.weight.min<-round(fdadata$animal.weight.min)
fdadata=mutate(fdadata, animalweight = ifelse(fdadata$animal.weight.min %in%
0:5, 5, ifelse(fdadata$animal.weight.min %in% 5:10,10,
ifelse(fdadata$animal.weight.min %in% 10:15, 15,
ifelse(fdadata$animal.weight.min %in% 15:20, 20,
ifelse(fdadata$animal.weight.min %in% 20:25, 25,
ifelse(fdadata$animal.weight.min %in% 25:30,
30,ifelse(fdadata$animal.weight.min %in% 30:35, 35, 40)))))))



#fdadata=mutate(fdadata,animalweight=ifelse(fdadata$animal.weight.min %in%
#0:20,20,40))

fdadata1<-fdadata[,c(1,10,25,26,27,29,38)]
```

```
fdadata2<-fdadata[,c(1,10,25,26,27,29,38)]
```

Various data preprocessing steps are done above to remove “NA” observations, to format dates, to convert ranges into numbers, and change the format to “factors” which is necessary to run some machine learning algorithms.

```
fdadata1=mutate(fdadata1, treated_for_ae = ifelse(treated_for_ae %in% "true",
"1", "0"))
fdadata1=mutate(fdadata1, health_assessment_prior_to_exposure.condition =
ifelse(health_assessment_prior_to_exposure.condition %in% "Good", "1", "0"))
fdadata1=mutate(fdadata1, animal.gender = ifelse(animal.gender %in% "Male",
"1", "0"))
fdadata1=mutate(fdadata1, animal.species = ifelse(animal.species %in% "Dog",
"1", "0"))
fdadata1=mutate(fdadata1, animal.reproductive_status =
ifelse(animal.reproductive_status %in% "Neutered", "1",
ifelse(animal.reproductive_status %in% "Intact", "2","0")))



fdadata1$treated_for_ae<-as.numeric(fdadata1$treated_for_ae)
fdadata1$onset_date<-as.numeric(fdadata1$onset_date)
fdadata1$health_assessment_prior_to_exposure.condition<-
as.numeric(fdadata1$health_assessment_prior_to_exposure.condition)
fdadata1$animal.gender<-as.numeric(fdadata1$animal.gender)
fdadata1$animal.species<-as.numeric(fdadata1$animal.species)
fdadata1$animal.reproductive_status<-
as.numeric(fdadata1$animal.reproductive_status)
fdadata1$animalweight<-as.numeric(fdadata1$animalweight)
```

Some more data preprocessing to change catergorical data to numerical data which makes it easier to run correlation matrix and PCA.

```
View(fdadata)
view(fdadata1)

library(corrplot)

## corrplot 0.85 loaded

#devtools::install_github("vsimko/corrplot")
x<-cor(fdadata1)
#corrplot(x, method="circle", mar=c(1,1,1,1))
corrplot(x, type = "upper", tl.pos = "td",
         method = "circle", tl.cex = 0.5, tl.col = 'black',
         order = "hclust", diag = FALSE)
```

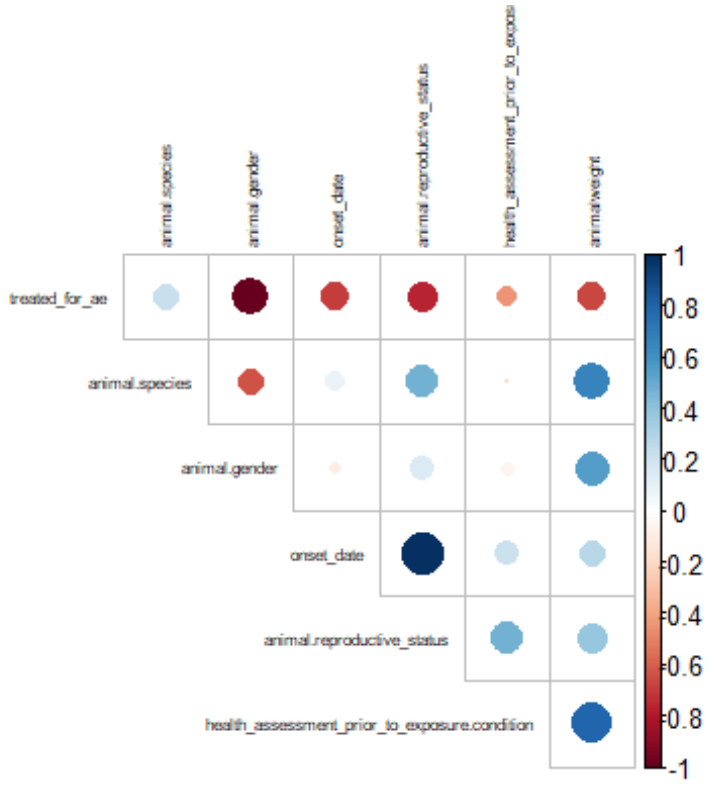


Figure above shows correlation between various features. As we can see onset-date is very positively correlated to animal-reproductive_status and similary treated_for_ae is negatively correlated to animal gender.

```
Cov_data <- cov(fdadata1)
Eigen_data <- eigen(Cov_data)
PCA_data <- princomp(fdadata1 ,cor="False")
Eigen_data$values
```

```
## [1] 129.54263139   5.09136181   0.29144668   0.24924382   0.15948321
## [6]   0.09640272   0.07745506
```

```
PCA_data$sdev^2
```

```
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5
## 127.94333965   5.02850549   0.28784857   0.24616674   0.15751429
##       Comp.6       Comp.7
##   0.09521256   0.07649882
```

```
PCA_data$loadings[,1:7]
```

```
##                                                    Comp.1        Comp.2
## treated_for_ae                                0.005046766  0.0251092220
## onset_date                                   -0.025524696 -0.9963150246
## health_assessment_prior_to_exposure.condition -0.007364130 -0.0107619379
## animal.gender                                -0.008818313  0.0005228848
## animal.species                               -0.006391280 -0.0063060129
## animal.reproductive_status                   -0.007393993 -0.0766871217
```

```
## animalweight                                        -0.999547651  0.0262512019
##                                                           Comp.3        Comp.4
## treated_for_ae                                       0.39414890  0.2222065664
## onset_date                                           0.06237399 -0.0511450229
## health_assessment_prior_to_exposure.condition       -0.06632710  0.0839046917
## animal.gender                                       -0.61323837 -0.6269933545
## animal.species                                       0.04990708  0.2341781022
## animal.reproductive_status                          -0.67652364  0.7021452546
## animalweight                                         0.01098148  0.0006499635
##                                                           Comp.5        Comp.6
## treated_for_ae                                       0.871014861  0.145185271
## onset_date                                           0.013951392 -0.001924262
## health_assessment_prior_to_exposure.condition       -0.047700690  0.815657983
## animal.gender                                        0.462124638 -0.044125379
## animal.species                                       0.120854504 -0.558265537
## animal.reproductive_status                           0.103424508  0.003234396
## animalweight                                        -0.001221871 -0.001292136
##                                                           Comp.7
## treated_for_ae                                       0.1219396892
## onset_date                                           0.0004357709
## health_assessment_prior_to_exposure.condition       -0.5664074219
## animal.gender                                       -0.1234251635
## animal.species                                      -0.7850624597
## animal.reproductive_status                           0.1807349691
## animalweight                                         0.0095492982
```

```
Eigen_data$vectors
```

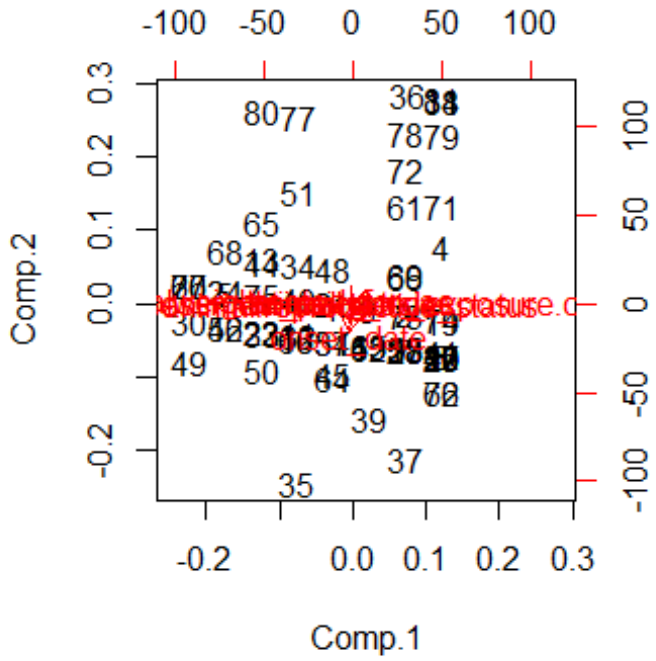
```
##              [,1]          [,2]        [,3]          [,4]         [,5]
## [1,]  0.005046766  0.0251092220  0.39414890 -0.2222065664  0.871014861
## [2,] -0.025524696 -0.9963150246  0.06237399  0.0511450229  0.013951392
## [3,] -0.007364130 -0.0107619379 -0.06632710 -0.0839046917 -0.047700690
## [4,] -0.008818313  0.0005228848 -0.61323837  0.6269933545  0.462124638
## [5,] -0.006391280 -0.0063060129  0.04990708 -0.2341781022  0.120854504
## [6,] -0.007393993 -0.0766871217 -0.67652364 -0.7021452546  0.103424508
## [7,] -0.999547651  0.0262512019  0.01098148 -0.0006499635 -0.001221871
##              [,6]          [,7]
## [1,]  0.145185271  0.1219396892
## [2,] -0.001924262  0.0004357709
## [3,]  0.815657983 -0.5664074219
## [4,] -0.044125379 -0.1234251635
## [5,] -0.558265537 -0.7850624597
## [6,]  0.003234396  0.1807349691
## [7,] -0.001292136  0.0095492982
```

```
summary(PCA_data)
```

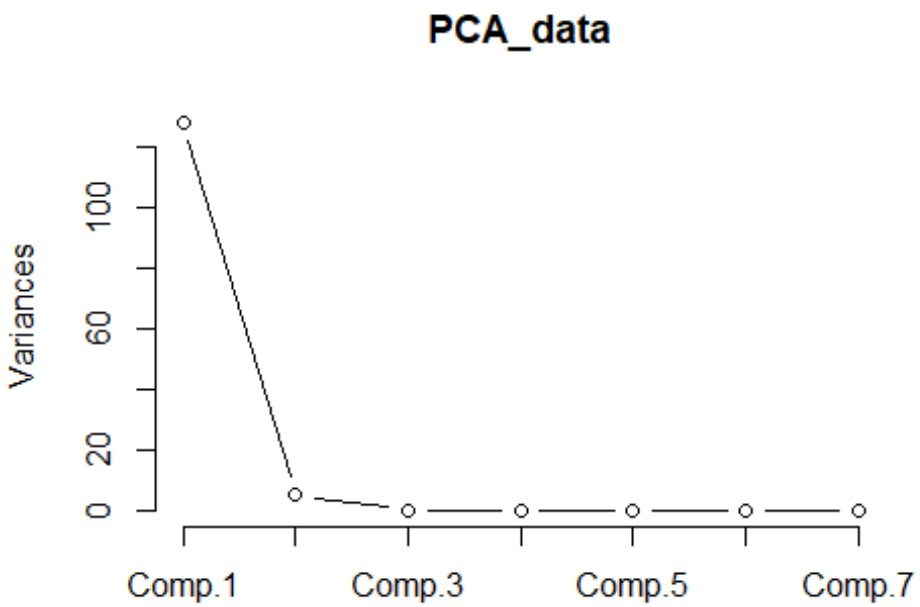
```
## Importance of components:
##                             Comp.1    Comp.2      Comp.3      Comp.4
## Standard deviation     11.3112042 2.2424329 0.536515209 0.496151930
## Proportion of Variance  0.9559776 0.0375724 0.002150771 0.001839329
```

```
## Cumulative Proportion   0.9559776 0.9935500 0.995700736 0.997540065
##                             Comp.5      Comp.6       Comp.7
## Standard deviation     0.396880694 0.308565328 0.2765842085
## Proportion of Variance 0.001176928 0.000711417 0.0005715902
## Cumulative Proportion  0.998716993 0.999428410 1.0000000000
```

```
biplot (PCA_data)
```




```
screeplot(PCA_data, type="lines")
```



PCA is a dimension reduction technique that not only provides better analyses by reducing the features but also enables a much better visual representation of the data. As we can see in the PCA results above, out of the 7 features, just 2 are able to account for nearly 100% of the variance in the data.

```
library(dplyr)
fdadata1 %>% filter(animal.species=="1") %>% group_by(animal.gender) %>%
summarise(animalweight=mean(animalweight))
```

```
## # A tibble: 2 x 2
##   animal.gender animalweight
##           <dbl>        <dbl>
## 1             0         15.8
## 2             1         21.0
```

The code above can run some stratification similarly to a pivot chart in excel.

```
set.seed(8)
train <- sample(1:nrow(fdadata2), nrow(fdadata2)/2)
test <- fdadata2[-train,]
x_test <- test[,-c(1)]

treated_for_ae.test <- fdadata2$treated_for_ae[-train]

library(randomForest)

## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(8)
bag.fda <- randomForest(treated_for_ae ~ .,
data = fdadata2,
subset = train,
mtry=6,
importance = TRUE)

print(bag.fda)
```

```
##
## Call:
##  randomForest(formula = treated_for_ae ~ ., data = fdadata2, mtry = 6,
importance = TRUE, subset = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 35%
## Confusion matrix:
##       false true class.error
## false    23    8   0.2580645
## true      6    3   0.6666667
```

```
bag.pred <- predict(bag.fda, newdata = test, type="class")

print(mean(bag.pred!=treated_for_ae.test))
```

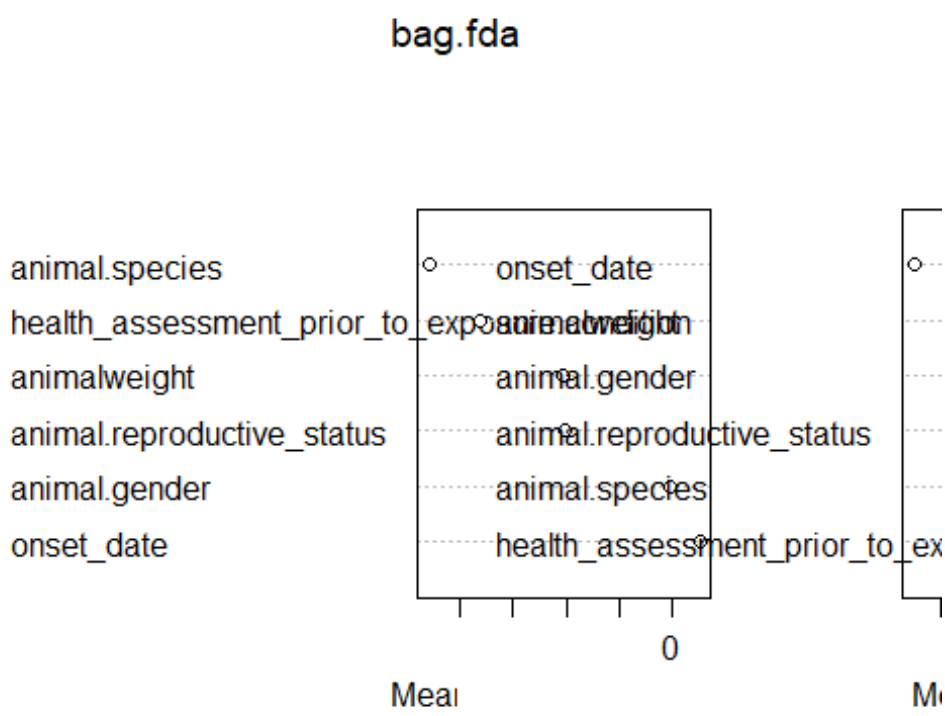
```
## [1] 0.4390244
```

Finally I have performed a random forest analysis. Because the FDA data is essentially a single class dataset i.e. all the observations are for animal that did have emesis, I ran RF on a level below the dataset where the dependent variable was “treated for ae”. The ideas was to see if the features that caused the animal to be treated for ae could perhaps have also been the determinant features in causing emesis.

```
importance(bag.fda)
```

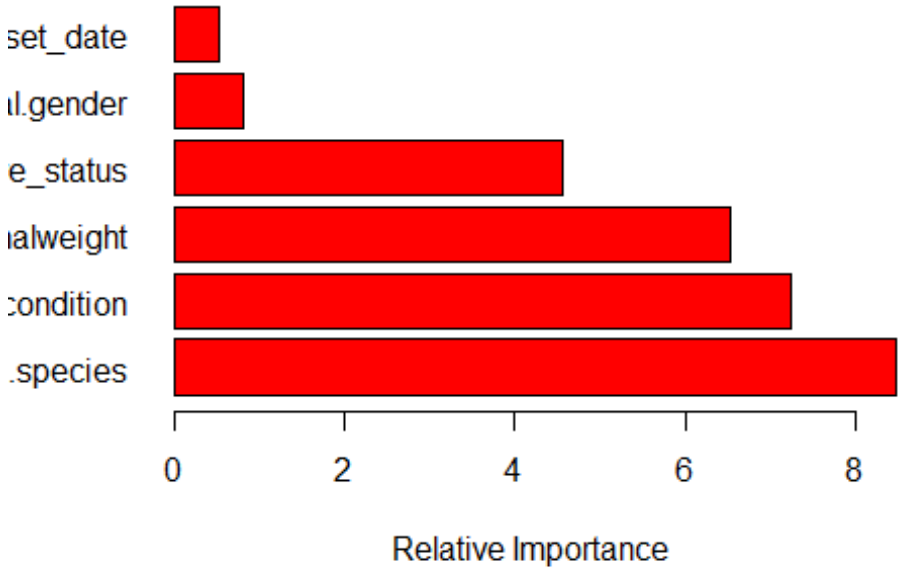
```
##                                                    false       true
## onset_date                                     0.5154172 -2.7900004
## health_assessment_prior_to_exposure.condition 7.2425750  1.6691101
## animal.gender                                  0.8008253 -0.7107609
## animal.species                                 8.4825608  4.1253104
## animal.reproductive_status                     4.5580753 -0.9687327
## animalweight                                   6.5287709 -3.4003820
##                                                MeanDecreaseAccuracy
## onset_date                                               -1.03744461
## health_assessment_prior_to_exposure.condition             7.23558270
## animal.gender                                             0.08126755
## animal.species                                            9.14933408
## animal.reproductive_status                                4.03766047
## animalweight                                              4.09328198
##                                                MeanDecreaseGini
## onset_date                                            5.5616050
## health_assessment_prior_to_exposure.condition         0.4932070
## animal.gender                                         1.3852781
## animal.species                                        0.9310356
## animal.reproductive_status                            1.1331466
## animalweight                                          4.1028277
```

```
varImpPlot(bag.fda)
```




```
barplot(sort(importance(bag.fda)[,1], decreasing = TRUE),
xlab = "Relative Importance",
horiz = TRUE,
```

```
col = "red",
las=1 #The las argument will allow rotation of 90 degrees for labels
)
```



Finally I ran a relative importance plot that yield the features most contributing toward the animal being treated for ae.