HealthcareAI[95]

Parinitha Kompala

9/8/2021

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
remove(list=ls())
library(foreign)
library(randomForest)
```

```
## randomForest 4.6-14

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

```
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-2
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(MASS)
library(boot)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::combine()  masks randomForest::combine()
## x tidyr::expand()   masks Matrix::expand()
## x dplyr::filter()   masks stats::filter()
## x dplyr::lag()      masks stats::lag()
## x ggplot2::margin() masks randomForest::margin()
## x tidyr::pack()     masks Matrix::pack()
## x dplyr::select()   masks MASS::select()
## x tidyr::unpack()   masks Matrix::unpack()
```

```
library(readxl)
#install RColorBrewer - install.packages("RColorBrewer")
library(RColorBrewer)
#install e1071 - install.packages("e1071")
library(e1071)
library(healthcareai)
```

```
## healthcareai version 2.5.0
## Please visit https://docs.healthcare.ai for full documentation and vignettes. Join the community at 
```

```
##
## Attaching package: 'healthcareai'
```

```
## The following object is masked from 'package:e1071':
##
##     impute
```

```
library(DescTools)
```

```
##
## Attaching package: 'DescTools'
```

```
## The following object is masked from 'package:healthcareai':
##
##     Mode
```

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
##
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:DescTools':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(broom)
```

r loading data

```
set.seed(539847)

setwd("/Users/parinithakompala/Desktop")
folder <- getwd()

#Load in data
df.set1 <- read_excel("/Users/parinithakompala/Library/Containers/com.microsoft.Excel/Data/Downloads/Adu
head(df.set1)
```

```
## # A tibble: 6 x 260
##   readmit30d_yn_state  dead obleed  ocva oleginf  otia   orf opneu oafib2 return
##                 <dbl> <dbl>  <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>
## 1                   1     0      0     0       0     0     0     0      1      0
## 2                   1     0      0     0       0     0     0     0      1      0
## 3                   0     0      0     0       0     0     0     0      0      0
## 4                   0     0      0     0       0     0     0     0      0      0
## 5                   0     0      0     0       0     0     0     0      1      0
## 6                   0     0      0     0       0     0     0     0      0      0
## # ... with 250 more variables: akin <dbl>, lm2cat <dbl>, rfcr_2 <dbl>,
## #   iabppre_2 <dbl>, atfibyn_2 <dbl>, hyperyn_2 <dbl>, ef50_neg <dbl>,
## #   ef50_neg_d <dbl>, priormi_21 <dbl>, priormi_22 <dbl>, priormi_23 <dbl>,
## #   priormi_24 <dbl>, age_d <dbl>, ua_nmi7 <dbl>, creatpre_2 <dbl>, rf_1 <dbl>,
## #   rf_2 <dbl>, rf_3 <dbl>, priority_21 <dbl>, priority_22 <dbl>, dm3cat <dbl>,
## #   dm3cat_21 <dbl>, dm3cat_22 <dbl>, dm3cat_21_2 <dbl>, dm3cat_22_2 <dbl>,
## #   pci_ta <dbl>, efvalue_r <dbl>, ef50_neg_r <dbl>, ef50_neg_d_r <dbl>, ...
```

```
names(df.set1) <- tolower(names(df.set1))
names(df.set1)
```

```
##   [1] "readmit30d_yn_state" "dead"                 "obleed"
##   [4] "ocva"                "oleginf"              "otia"
##   [7] "orf"                 "opneu"                "oafib2"
##  [10] "return"              "akin"                 "lm2cat"
##  [13] "rfcr_2"              "iabppre_2"            "atfibyn_2"
##  [16] "hyperyn_2"           "ef50_neg"             "ef50_neg_d"
##  [19] "priormi_21"          "priormi_22"           "priormi_23"
##  [22] "priormi_24"          "age_d"                "ua_nmi7"
##  [25] "creatpre_2"          "rf_1"                 "rf_2"
##  [28] "rf_3"                "priority_21"          "priority_22"
##  [31] "dm3cat"              "dm3cat_21"            "dm3cat_22"
##  [34] "dm3cat_21_2"         "dm3cat_22_2"          "pci_ta"
##  [37] "efvalue_r"           "ef50_neg_r"           "ef50_neg_d_r"
##  [40] "atfibyn_2_r"         "priormi_r"            "priormi_21_r"
##  [43] "priormi_22_r"        "priormi_23_r"         "priormi_24_r"
##  [46] "age_r"               "age_d_r"              "ua_r"
##  [49] "ua_nmi7_r"           "chf_r"                "chf2cat_r"
##  [52] "rf_r"                "creatpre_2_r"         "rf_1_r"
##  [55] "rf_2_r"              "rf_3_r"               "priority_r"
##  [58] "priority_21_r"       "priority_22_r"        "sex_r"
##  [61] "prcabg_r"            "copd_r"               "anydm_r"
##  [64] "dmtx_r"              "dm3cat_r"             "dm3cat_21_r"
##  [67] "dm3cat_22_r"         "iabppre_r"            "anyvad_r"
##  [70] "vad_r"               "hyper_r"              "hyperyn_2_r"
##  [73] "prptca6_r"           "pci_ta_r"             "lm3cat_r"
##  [76] "lm2cat_r"            "aortic_insuff"        "aortic_insuff_r"
##  [79] "aortic_sten"         "aortic_sten_r"       "chf_nyha_iv"        
##  [82] "chf_nyha_ltiv"       "chf_nyha_iv_r"       "chf_nyha_ltiv_r"    
##  [85] "smoker_r"            "cvd"                 "cvd_r"              
##  [88] "htcm_d"              "htcm_r"              "htcm_d_r"           
##  [91] "mitral_insuff"       "mitral_insuff_r"     "carotid_sten"       
##  [94] "carotid_sten_r"      "pvd"                 "pvd_r"              
##  [97] "tricusp_insuff"      "tricusp_insuff_r"    "bmi_squared"        
## [100] "bmi_r"               "bmi_squared_r"       "novsl_r"            
## [103] "readmit_1y_yn_state" "anyakin"             "creatcat"           
## [106] "lm50"                "anymssd"             "lowoutput"          
## [109] "emerg"               "urg"                 "elec"               
## [112] "bmicat"              "bmi1"                "bmi2"               
## [115] "bmi3"                "bmi4"                "bmi5"               
## [118] "bmi6"                "lvedpm"              "anemiapre"          
## [121] "iabpintra"           "lof1"                "ptimenumban"        
## [124] "ctimenumban"         "cardtimenumban"      "cardblood"          
## [127] "cardcold"            "hotshot"             "aoxcon"             
## [130] "ultrafilyn"          "cabg"                "valve"              
## [133] "cabgvalve"           "gfr60pre"            "male"               
## [136] "notcoldcard"         "fluidprel"           "ptime120"           
## [139] "heptotl"             "heptot5"             "tcys0"              
## [142] "cyspre3cat"          "i_cyspre3cat1"       "i_cyspre3cat2"      
## [145] "i_cyspre3cat3"       "logcys0"             "tcys1"              
## [148] "cyspost3cat"         "i_cyspost3cat1"      "i_cyspost3cat2"     
## [151] "i_cyspost3cat3"      "logcys1"             "cysdiff"            
## [154] "tcysdiff"            "cysdiff3cat"         "i_cysdiff3cat1"     
## [157] "i_cysdiff3cat2"      "i_cysdiff3cat3"      "logcysdiff"         
## [160] "til10_0"             "il10pre3cat"         "i_il10pre3cat1"     
## [163] "i_il10pre3cat2"      "i_il10pre3cat3"      "logil100"           
## [166] "til10_1"             "il10post3cat"        "i_il10post3cat1"    
## [169] "i_il10post3cat2"     "i_il10post3cat3"     "logil101"           
## [172] "il10diff"            "til10diff"           "il10diff3cat"       
## [175] "i_il10diff3cat1"     "i_il10diff3cat2"     "i_il10diff3cat3"    
## [178] "logil10diff"         "til6_0"              "il6pre3cat"         
## [181] "i_il6pre3cat1"       "i_il6pre3cat2"       "i_il6pre3cat3"      
## [184] "logil60"             "til6_1"              "il6post3cat"        
## [187] "i_il6post3cat1"      "i_il6post3cat2"      "i_il6post3cat3"     
## [190] "logil61"             "il6diff"             "til6diff"           
## [193] "il6diff3cat"         "i_il6diff3cat1"      "i_il6diff3cat2"     
## [196] "i_il6diff3cat3"      "logil6diff"          "gal30_adj"          
## [199] "tgal3_0"             "gal3pre3cat"         "i_gal3pre3cat1"     
## [202] "i_gal3pre3cat2"      "i_gal3pre3cat3"      "loggal30"           
## [205] "gal31_adj"           "tgal3_1"             "gal3post3cat"       
## [208] "i_gal3post3cat1"     "i_gal3post3cat2"     "i_gal3post3cat3"    
## [211] "loggal31"            "gal3diff"            "tgal3diff"          
## [214] "gal3diff3cat"        "i_gal3diff3cat1"     "i_gal3diff3cat2"    
## [217] "i_gal3diff3cat3"     "loggal3diff"         "ntpro0_adj"         
## [220] "tntbnp_0"            "ntbnppre3cat"        "i_ntbnppre3cat1"    
## [223] "i_ntbnppre3cat2"     "i_ntbnppre3cat3"     "logntp0"            
## [226] "ntpro1_adj"          "tntbnp_1"            "ntbnppost3cat"      
## [229] "i_ntbnppost3cat1"    "i_ntbnppost3cat2"    "i_ntbnppost3cat3"   
## [232] "logntp1"             "ntbnpdiff"           "tntbnpdiff"         
## [235] "ntbnpdiff3cat"       "i_ntbnpdiff3cat1"    "i_ntbnpdiff3cat2"   
## [238] "i_ntbnpdiff3cat3"    "logntbnpdiff"        "st20_adj"           
## [241] "tst2_0"               "st2pre3cat"           "i_st2pre3cat1"
## [244] "i_st2pre3cat2"        "i_st2pre3cat3"        "logst20"
## [247] "st21_adj"             "tst2_1"               "st2post3cat"
## [250] "i_st2post3cat1"       "i_st2post3cat2"       "i_st2post3cat3"
## [253] "logst21"              "st2diff"              "tst2diff"
## [256] "st2diff3cat"          "i_st2diff3cat1"       "i_st2diff3cat2"
## [259] "i_st2diff3cat3"       "logst2diff"
```

removing NA data

```
df.set1.2 <- na.omit(df.set1)
df.set1.2
```

```
## # A tibble: 25 x 260
##    readmit30d_yn_state  dead obleed  ocva oleginf  otia   orf opneu oafib2 return
##                  <dbl> <dbl>  <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>
##  1                   0     0      0     0       0     0     0     0      0      0
##  2                   0     0      0     0       0     0     0     0      0      0
##  3                   0     0      0     0       0     0     0     0      0      0
##  4                   0     0      0     0       0     0     0     0      0      0
##  5                   0     0      0     0       0     0     0     0      0      0
##  6                   0     0      0     0       0     0     0     0      0      0
##  7                   0     0      0     0       0     0     0     0      0      0
##  8                   0     0      0     0       0     0     0     0      0      0
##  9                   0     0      0     0       0     0     0     0      0      0
## 10                   0     0      0     0       0     0     0     0      0      0
## # ... with 15 more rows, and 250 more variables: akin <dbl>, lm2cat <dbl>,
## #   rfcr_2 <dbl>, iabppre_2 <dbl>, atfibyn_2 <dbl>, hyperyn_2 <dbl>,
## #   ef50_neg <dbl>, ef50_neg_d <dbl>, priormi_21 <dbl>, priormi_22 <dbl>,
## #   priormi_23 <dbl>, priormi_24 <dbl>, age_d <dbl>, ua_nmi7 <dbl>,
## #   creatpre_2 <dbl>, rf_1 <dbl>, rf_2 <dbl>, rf_3 <dbl>, priority_21 <dbl>,
## #   priority_22 <dbl>, dm3cat <dbl>, dm3cat_21 <dbl>, dm3cat_22 <dbl>,
## #   dm3cat_21_2 <dbl>, dm3cat_22_2 <dbl>, pci_ta <dbl>, efvalue_r <dbl>, ...
```

```
str(df.set1.2)
```

```
## tibble [25 x 260] (S3: tbl_df/tbl/data.frame)
##  $ readmit30d_yn_state: num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ dead               : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ obleed             : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ ocva               : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ oleginf            : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ otia               : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ orf                : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ opneu              : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ oafib2             : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ return             : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ akin               : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ lm2cat             : num [1:25] 0 0 0 0 0 1 0 0 0 1 ...
##  $ rfcr_2             : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ iabppre_2          : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ atfibyn_2          : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ hyperyn_2          : num [1:25] 0 1 1 1 1 1 1 1 1 1 ...
##  $ ef50_neg           : num [1:25] 26 11 4 0 0 0 3 0 0 0 ...
##  $ ef50_neg_d         : num [1:25] 4 3 2 1 1 1 2 1 1 1 ...
##  $ priormi_21         : num [1:25] 1 1 0 0 0 0 1 0 0 0 ...
##  $ priormi_22        : num [1:25] 1 1 0 0 0 0 1 0 0 0 ...
##  $ priormi_23        : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ priormi_24        : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ age_d             : num [1:25] 7 6 6 7 7 7 6 8 7 8 ...
##  $ ua_nmi7           : num [1:25] 0 0 1 0 1 0 0 1 1 1 ...
##  $ creatpre_2        : num [1:25] 1.1 0.9 1.2 1.2 1.2 ...
##  $ rf_1              : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ rf_2              : num [1:25] 0.1 0 0.2 0.2 0.2 ...
##  $ rf_3              : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ priority_21       : num [1:25] 1 1 1 1 1 0 1 1 1 1 ...
##  $ priority_22       : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ dm3cat            : num [1:25] 2 2 2 2 2 2 2 2 2 3 ...
##  $ dm3cat_21         : num [1:25] 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22         : num [1:25] 0 0 0 0 0 0 0 0 0 1 ...
##  $ dm3cat_21_2       : num [1:25] 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22_2       : num [1:25] 0 0 0 0 0 0 0 0 0 1 ...
##  $ pci_ta            : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ efvalue_r         : num [1:25] 24 39 46 50 71 58 47 63 66 55 ...
##  $ ef50_neg_r        : num [1:25] 26 11 4 0 0 0 3 0 0 0 ...
##  $ ef50_neg_d_r      : num [1:25] 4 3 2 1 1 1 2 1 1 1 ...
##  $ atfibyn_2_r       : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ priormi_r         : num [1:25] 2 2 0 0 0 0 2 0 0 0 ...
##  $ priormi_21_r      : num [1:25] 1 1 0 0 0 0 1 0 0 0 ...
##  $ priormi_22_r      : num [1:25] 1 1 0 0 0 0 1 0 0 0 ...
##  $ priormi_23_r      : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ priormi_24_r      : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ age_r             : num [1:25] 62.5 51.4 55.5 69.1 68.9 ...
##  $ age_d_r           : num [1:25] 7 6 6 7 7 7 6 8 7 8 ...
##  $ ua_r              : num [1:25] 1 1 1 0 1 0 1 1 1 1 ...
##  $ ua_nmi7_r         : num [1:25] 0 0 1 0 1 0 0 1 1 1 ...
##  $ chf_r             : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ chf2cat_r         : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ rf_r              : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ creatpre_2_r      : num [1:25] 1.1 0.9 1.2 1.2 1.2 ...
##  $ rf_1_r            : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ rf_2_r            : num [1:25] 0.1 0 0.2 0.2 0.2 ...
##  $ rf_3_r            : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ priority_r        : num [1:25] 2 1 2 2 2 3 2 2 2 2 ...
##  $ priority_21_r     : num [1:25] 1 1 1 1 1 0 1 1 1 1 ...
##  $ priority_22_r     : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ sex_r             : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ prcabg_r          : num [1:25] 0 0 0 0 0 1 0 0 0 0 ...
##  $ copd_r            : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ anydm_r           : num [1:25] 1 1 1 1 1 1 1 1 1 1 ...
##  $ dmtx_r            : num [1:25] 0 2 2 1 1 2 2 2 1 3 ...
##  $ dm3cat_r          : num [1:25] 2 2 2 2 2 2 2 2 2 3 ...
##  $ dm3cat_21_r       : num [1:25] 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22_r       : num [1:25] 0 0 0 0 0 0 0 0 0 1 ...
##  $ iabppre_r         : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ anyvad_r          : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ vad_r             : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ hyper_r           : num [1:25] 0 1 2 1 1 1 1 1 2 1 ...
##  $ hyperyn_2_r       : num [1:25] 0 1 1 1 1 1 1 1 1 1 ...
##  $ prptca6_r         : num [1:25] 2 0 0 0 2 2 0 0 0 0 ...
##  $ pci_ta_r           : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ lm3cat_r           : num [1:25] 1 1 1 1 1 3 1 1 1 2 ...
##  $ lm2cat_r           : num [1:25] 0 0 0 0 0 1 0 0 0 1 ...
##  $ aortic_insuff      : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ aortic_insuff_r    : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ aortic_sten        : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ aortic_sten_r      : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ chf_nyha_iv        : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ chf_nyha_ltiv      : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ chf_nyha_iv_r      : num [1:25] 0 1 0 0 0 0 0 0 0 0 ...
##  $ chf_nyha_ltiv_r    : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ smoker_r           : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ cvd                : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ cvd_r              : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ htcm_d             : num [1:25] 17 17 18 18 17 16 17 18 17 17 ...
##  $ htcm_r             : num [1:25] 170 173 185 183 178 ...
##  $ htcm_d_r           : num [1:25] 17 17 18 18 17 16 17 18 17 17 ...
##  $ mitral_insuff      : num [1:25] 1 0 0 0 0 0 0 0 0 0 ...
##  $ mitral_insuff_r    : num [1:25] 1 0 0 0 0 0 0 0 0 0 ...
##  $ carotid_sten       : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ carotid_sten_r     : num [1:25] 0 0 0 1 0 0 0 0 0 0 ...
##  $ pvd                : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ pvd_r              : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ tricusp_insuff     : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ tricusp_insuff_r   : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ bmi_squared        : num [1:25] 604 1158 886 1079 825 ...
##   [list output truncated]
##  - attr(*, "na.action")= 'omit' Named int [1:1626] 1 2 3 4 5 6 7 8 9 10 ...
##   ..- attr(*, "names")= chr [1:1626] "1" "2" "3" "4" ...
```

```r
variable.names <- names(df.set1.2)
variable.names
```

```
##   [1] "readmit30d_yn_state" "dead"                 "obleed"
##   [4] "ocva"                "oleginf"              "otia"
##   [7] "orf"                 "opneu"                "oafib2"
##  [10] "return"              "akin"                 "lm2cat"
##  [13] "rfcr_2"              "iabppre_2"            "atfibyn_2"
##  [16] "hyperyn_2"           "ef50_neg"             "ef50_neg_d"
##  [19] "priormi_21"          "priormi_22"           "priormi_23"
##  [22] "priormi_24"          "age_d"                "ua_nmi7"
##  [25] "creatpre_2"          "rf_1"                 "rf_2"
##  [28] "rf_3"                "priority_21"          "priority_22"
##  [31] "dm3cat"              "dm3cat_21"            "dm3cat_22"
##  [34] "dm3cat_21_2"         "dm3cat_22_2"          "pci_ta"
##  [37] "efvalue_r"           "ef50_neg_r"           "ef50_neg_d_r"
##  [40] "atfibyn_2_r"         "priormi_r"            "priormi_21_r"
##  [43] "priormi_22_r"        "priormi_23_r"         "priormi_24_r"
##  [46] "age_r"               "age_d_r"              "ua_r"
##  [49] "ua_nmi7_r"           "chf_r"                "chf2cat_r"
##  [52] "rf_r"                "creatpre_2_r"         "rf_1_r"
##  [55] "rf_2_r"              "rf_3_r"               "priority_r"
##  [58] "priority_21_r"       "priority_22_r"        "sex_r"
##  [61] "prcabg_r"            "copd_r"               "anydm_r"
##  [64] "dmtx_r"              "dm3cat_r"             "dm3cat_21_r"
##  [67] "dm3cat_22_r"          "iabppre_r"            "anyvad_r"
##  [70] "vad_r"                "hyper_r"              "hyperyn_2_r"
##  [73] "prptca6_r"            "pci_ta_r"             "lm3cat_r"
##  [76] "lm2cat_r"             "aortic_insuff"        "aortic_insuff_r"
##  [79] "aortic_sten"          "aortic_sten_r"        "chf_nyha_iv"
##  [82] "chf_nyha_ltiv"        "chf_nyha_iv_r"        "chf_nyha_ltiv_r"
##  [85] "smoker_r"             "cvd"                  "cvd_r"
##  [88] "htcm_d"               "htcm_r"               "htcm_d_r"
##  [91] "mitral_insuff"        "mitral_insuff_r"      "carotid_sten"
##  [94] "carotid_sten_r"       "pvd"                  "pvd_r"
##  [97] "tricusp_insuff"       "tricusp_insuff_r"     "bmi_squared"
## [100] "bmi_r"                "bmi_squared_r"        "novsl_r"
## [103] "readmit_1y_yn_state" "anyakin"              "creatcat"
## [106] "lm50"                 "anymssd"              "lowoutput"
## [109] "emerg"                "urg"                  "elec"
## [112] "bmicat"               "bmi1"                 "bmi2"
## [115] "bmi3"                 "bmi4"                 "bmi5"
## [118] "bmi6"                 "lvedpm"               "anemiapre"
## [121] "iabpintra"            "lof1"                 "ptimenumban"
## [124] "ctimenumban"          "cardtimenumban"       "cardblood"
## [127] "cardcold"             "hotshot"              "aoxcon"
## [130] "ultrafilyn"           "cabg"                 "valve"
## [133] "cabgvalve"            "gfr60pre"             "male"
## [136] "notcoldcard"          "fluidprel"            "ptime120"
## [139] "heptotl"              "heptot5"              "tcys0"
## [142] "cyspre3cat"           "i_cyspre3cat1"        "i_cyspre3cat2"
## [145] "i_cyspre3cat3"        "logcys0"              "tcys1"
## [148] "cyspost3cat"          "i_cyspost3cat1"       "i_cyspost3cat2"
## [151] "i_cyspost3cat3"       "logcys1"              "cysdiff"
## [154] "tcysdiff"             "cysdiff3cat"          "i_cysdiff3cat1"
## [157] "i_cysdiff3cat2"       "i_cysdiff3cat3"       "logcysdiff"
## [160] "til10_0"              "il10pre3cat"          "i_il10pre3cat1"
## [163] "i_il10pre3cat2"       "i_il10pre3cat3"       "logil100"
## [166] "til10_1"              "il10post3cat"         "i_il10post3cat1"
## [169] "i_il10post3cat2"      "i_il10post3cat3"      "logil101"
## [172] "il10diff"             "til10diff"            "il10diff3cat"
## [175] "i_il10diff3cat1"      "i_il10diff3cat2"      "i_il10diff3cat3"
## [178] "logil10diff"          "til6_0"               "il6pre3cat"
## [181] "i_il6pre3cat1"        "i_il6pre3cat2"        "i_il6pre3cat3"
## [184] "logil60"              "til6_1"               "il6post3cat"
## [187] "i_il6post3cat1"       "i_il6post3cat2"       "i_il6post3cat3"
## [190] "logil61"              "il6diff"              "til6diff"
## [193] "il6diff3cat"          "i_il6diff3cat1"       "i_il6diff3cat2"
## [196] "i_il6diff3cat3"       "logil6diff"           "gal30_adj"
## [199] "tgal3_0"              "gal3pre3cat"          "i_gal3pre3cat1"
## [202] "i_gal3pre3cat2"       "i_gal3pre3cat3"       "loggal30"
## [205] "gal31_adj"            "tgal3_1"              "gal3post3cat"
## [208] "i_gal3post3cat1"      "i_gal3post3cat2"      "i_gal3post3cat3"
## [211] "loggal31"             "gal3diff"             "tgal3diff"
## [214] "gal3diff3cat"         "i_gal3diff3cat1"      "i_gal3diff3cat2"
## [217] "i_gal3diff3cat3"      "loggal3diff"          "ntpro0_adj"
## [220] "tntbnp_0"             "ntbnppre3cat"         "i_ntbnppre3cat1"
## [223] "i_ntbnppre3cat2"      "i_ntbnppre3cat3"      "logntp0"
## [226] "ntpro1_adj"           "tntbnp_1"             "ntbnppost3cat"
## [229] "i_ntbnppost3cat1"    "i_ntbnppost3cat2"    "i_ntbnppost3cat3"
## [232] "logntp1"             "ntbnpdiff"           "tntbnpdiff"
## [235] "ntbnpdiff3cat"       "i_ntbnpdiff3cat1"    "i_ntbnpdiff3cat2"
## [238] "i_ntbnpdiff3cat3"    "logntbnpdiff"        "st20_adj"
## [241] "tst2_0"              "st2pre3cat"          "i_st2pre3cat1"
## [244] "i_st2pre3cat2"       "i_st2pre3cat3"       "logst20"
## [247] "st21_adj"            "tst2_1"              "st2post3cat"
## [250] "i_st2post3cat1"      "i_st2post3cat2"      "i_st2post3cat3"
## [253] "logst21"             "st2diff"             "tst2diff"
## [256] "st2diff3cat"         "i_st2diff3cat1"      "i_st2diff3cat2"
## [259] "i_st2diff3cat3"      "logst2diff"
```

helper function to get the variable names from a data frame and return list of names

```
get_var_names <- function(df.input, varsBool){
  names_list <- c()
  df.vars <- df.input[, varsBool]
  names <- names(df.vars)
  names_list <- append(names_list, names)
}
```

gets the interaction variable names

```
get_interactions <- function(df.input){
  variable_names <- names(df.input)
  interaction_vars <- startsWith(variable_names, "i_")
  interaction.names <- get_var_names(df.input, interaction_vars)
  return(interaction.names)
}
dim(df.set1.2)
```

```
## [1]  25 260
```

```
interaction.vars <- get_interactions(df.set1.2)
(interaction.vars)
```

```
##  [1] "i_cyspre3cat1"    "i_cyspre3cat2"    "i_cyspre3cat3"    "i_cyspost3cat1"
##  [5] "i_cyspost3cat2"   "i_cyspost3cat3"   "i_cysdiff3cat1"   "i_cysdiff3cat2"
##  [9] "i_cysdiff3cat3"   "i_il10pre3cat1"   "i_il10pre3cat2"   "i_il10pre3cat3"
## [13] "i_il10post3cat1"  "i_il10post3cat2"  "i_il10post3cat3"  "i_il10diff3cat1"
## [17] "i_il10diff3cat2"  "i_il10diff3cat3"  "i_il6pre3cat1"    "i_il6pre3cat2"
## [21] "i_il6pre3cat3"    "i_il6post3cat1"   "i_il6post3cat2"   "i_il6post3cat3"
## [25] "i_il6diff3cat1"   "i_il6diff3cat2"   "i_il6diff3cat3"   "i_gal3pre3cat1"
## [29] "i_gal3pre3cat2"   "i_gal3pre3cat3"   "i_gal3post3cat1"  "i_gal3post3cat2"
## [33] "i_gal3post3cat3"  "i_gal3diff3cat1"  "i_gal3diff3cat2"  "i_gal3diff3cat3"
## [37] "i_ntbnppre3cat1"  "i_ntbnppre3cat2"  "i_ntbnppre3cat3"  "i_ntbnppost3cat1"
## [41] "i_ntbnppost3cat2" "i_ntbnppost3cat3" "i_ntbnpdiff3cat1" "i_ntbnpdiff3cat2"
## [45] "i_ntbnpdiff3cat3" "i_st2pre3cat1"    "i_st2pre3cat2"    "i_st2pre3cat3"
## [49] "i_st2post3cat1"   "i_st2post3cat2"   "i_st2post3cat3"   "i_st2diff3cat1"
## [53] "i_st2diff3cat2"   "i_st2diff3cat3"
```

returns data set with variables excluded

```
df_excl_vars <- function(df.input, excluded_vars){
  new_df <- df.input[, !colnames(df.input) %in% excluded_vars]
  return(new_df)
}
#data frame without interactions
```

```r
df.no.interactions <- df_excl_vars(df.set1.2, interaction.vars)
```

get continous variable names

```r
get_cont_vars <- function(df.input, endings, start_labels, match_labels){
  variable_names <- names(df.input)
  continous_vars <- c()
  for (ending in endings){
    cont.vars <- endsWith(variable_names, ending)
    continous_vars <- append(continous_vars, get_var_names(df.input, cont.vars))
  }
  for(start in start_labels){
    cont.vars.2 <- startsWith(variable_names, start)
    continous_vars <- append(continous_vars, get_var_names(df.input, cont.vars.2))
  }
  cont.vars.3 <- tidyselect::vars_select(variable_names, one_of(match_labels))
  continous_vars <- append(continous_vars, get_var_names(df.input,cont.vars.3))
  return(continous_vars)
}

endings <- c("_min", "_avg", "_max", "_first", "_last", "_count", "_in_ed", "_duration",
             "_level_diff")
start_labels <- c("cubic_", "quad_", "cub_")
match_labels <- c("grace_score", "lace_score", "charlson_deyo_score", "index_los",
                  "hospital_score", "age_at_admit", "los_new", "los", "ck_level_max2",
                  "creatinine_diff","ed_visit_prior_30_days_minutes_i", "year")
dim(match_labels)
```

```
## NULL
```

```r
cont.vars <- get_cont_vars(df.no.interactions, endings, start_labels, match_labels)
```

```
## Warning: Unknown columns: `grace_score`, `lace_score`, `charlson_deyo_score`,
## `index_los`, `hospital_score`, `age_at_admit`, `los_new`, `los`,
## `ck_level_max2`, `creatinine_diff`, `ed_visit_prior_30_days_minutes_i`, `year`
```

```r
#data frame without continous variables (only factored variables left)
df.raw.factors <- df_excl_vars(df.no.interactions, cont.vars)
```

```r
#create data frame as factors
df.as.factors <- data.frame(lapply(df.raw.factors, factor))
str(df.as.factors)
```

```
## 'data.frame':    25 obs. of  206 variables:
##  $ readmit30d_yn_state: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dead               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ obleed             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ ocva               : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ oleginf            : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ otia               : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ orf                : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ opneu              : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ oafib2             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ return             : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ akin               : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ lm2cat             : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 2 ...
##  $ rfcr_2           : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ iabppre_2        : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ atfibyn_2        : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ hyperyn_2        : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ ef50_neg         : Factor w/ 8 levels "0","3","4","5",..: 8 5 3 1 1 1 2 1 1 1 ...
##  $ ef50_neg_d       : Factor w/ 4 levels "1","2","3","4": 4 3 2 1 1 1 2 1 1 1 ...
##  $ priormi_21       : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_22       : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_23       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priormi_24       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ age_d            : Factor w/ 5 levels "5","6","7","8",..: 3 2 2 3 3 3 2 4 3 4 ...
##  $ ua_nmi7          : Factor w/ 2 levels "0","1": 1 1 2 1 2 1 1 2 2 2 ...
##  $ creatpre_2       : Factor w/ 10 levels "0.69999999","0.80000001",..: 5 3 6 6 6 6 4 6 4 4 ...
##  $ rf_1             : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ rf_2             : Factor w/ 7 levels "0","0.10000002",..: 2 1 3 3 3 3 1 3 1 1 ...
##  $ rf_3             : Factor w/ 2 levels "0","0.39999998": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priority_21      : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 2 ...
##  $ priority_22      : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ dm3cat           : Factor w/ 2 levels "2","3": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_21        : Factor w/ 1 level "1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_21_2      : Factor w/ 1 level "1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22_2      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ pci_ta           : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ efvalue_r        : Factor w/ 18 levels "24","29","37",..: 1 4 6 8 18 12 7 14 15 10 ...
##  $ ef50_neg_r       : Factor w/ 8 levels "0","3","4","5",..: 8 5 3 1 1 1 2 1 1 1 ...
##  $ ef50_neg_d_r     : Factor w/ 4 levels "1","2","3","4": 4 3 2 1 1 1 2 1 1 1 ...
##  $ atfibyn_2_r      : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priormi_r        : Factor w/ 4 levels "0","2","3","4": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_21_r     : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_22_r     : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_23_r     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priormi_24_r     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ age_r            : Factor w/ 25 levels "43.04723","51.405888",..: 8 2 5 15 14 9 7 18 11 19 ...
##  $ age_d_r          : Factor w/ 5 levels "5","6","7","8",..: 3 2 2 3 3 3 2 4 3 4 ...
##  $ ua_r             : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 2 ...
##  $ ua_nmi7_r        : Factor w/ 2 levels "0","1": 1 1 2 1 2 1 1 2 2 2 ...
##  $ chf_r            : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ chf2cat_r        : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ rf_r             : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ creatpre_2_r     : Factor w/ 10 levels "0.69999999","0.80000001",..: 5 3 6 6 6 6 4 6 4 4 ...
##  $ rf_1_r           : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ rf_2_r           : Factor w/ 7 levels "0","0.10000002",..: 2 1 3 3 3 3 1 3 1 1 ...
##  $ rf_3_r           : Factor w/ 2 levels "0","0.39999998": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priority_r       : Factor w/ 3 levels "1","2","3": 2 1 2 2 2 3 2 2 2 2 ...
##  $ priority_21_r    : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 2 ...
##  $ priority_22_r    : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ sex_r            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ prcabg_r         : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ copd_r           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ anydm_r          : Factor w/ 1 level "1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dmtx_r           : Factor w/ 4 levels "0","1","2","3": 1 3 3 2 2 3 3 3 2 4 ...
##  $ dm3cat_r         : Factor w/ 2 levels "2","3": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_21_r      : Factor w/ 1 level "1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dm3cat_22_r       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ iabppre_r         : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ anyvad_r          : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ vad_r             : Factor w/ 4 levels "0","1","2","3": 1 1 1 2 1 1 1 1 1 1 ...
##  $ hyper_r           : Factor w/ 3 levels "0","1","2": 1 2 3 2 2 2 2 2 3 2 ...
##  $ hyperyn_2_r       : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ prptca6_r         : Factor w/ 2 levels "0","2": 2 1 1 1 2 2 1 1 1 1 ...
##  $ pci_ta_r          : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ lm3cat_r          : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 3 1 1 1 2 ...
##  $ lm2cat_r          : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 2 ...
##  $ aortic_insuff     : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ aortic_insuff_r   : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ aortic_sten       : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ aortic_sten_r     : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ chf_nyha_iv       : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ chf_nyha_ltiv     : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ chf_nyha_iv_r     : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ chf_nyha_ltiv_r   : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ smoker_r          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ cvd               : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ cvd_r             : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ htcm_d            : Factor w/ 6 levels "12","13","15",..: 5 5 6 6 5 4 5 6 5 5 ...
##  $ htcm_r            : Factor w/ 22 levels "122.1","137.11",..: 10 14 22 21 17 9 15 20 15 12 ...
##  $ htcm_d_r          : Factor w/ 6 levels "12","13","15",..: 5 5 6 6 5 4 5 6 5 5 ...
##  $ mitral_insuff     : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ mitral_insuff_r   : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ carotid_sten      : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ carotid_sten_r    : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ pvd               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ pvd_r             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ tricusp_insuff    : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ tricusp_insuff_r  : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ bmi_squared       : Factor w/ 25 levels "593.55017","603.56079",..: 2 19 15 18 11 13 21 16 8 7 .
##   [list output truncated]
```

```r
#double check to make sure levels aren't super large for any variables
factor.levels <- sapply(df.as.factors[,sapply(df.as.factors, is.factor)], nlevels)
```

```r
#get factors that have only 1 level
factors_w_1level <- function(factor_levels){
  drop.factors <- c()
  factor.levels.names <- names(factor_levels)
  for (name in factor.levels.names){
    if (factor_levels[name] == 1){
      drop.factors <- append(drop.factors, name)
    }
  }
  return(drop.factors)
}

drop_factors <- factors_w_1level(factor.levels)

df.factors.only <- df_excl_vars(df.as.factors, drop_factors)
```

```
#final data frame with factored variables and continous variables
df.set1.final <- data.frame(df.factors.only, df.set1.2[, colnames(df.set1.2) %in% cont.vars], df.set1.2
str(df.set1.final)
```

```
## 'data.frame':    25 obs. of  225 variables:
##  $ readmit30d_yn_state: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dead               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ obleed             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ oafib2             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ akin               : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ lm2cat             : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 2 ...
##  $ hyperyn_2          : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ ef50_neg           : Factor w/ 8 levels "0","3","4","5",..: 8 5 3 1 1 1 2 1 1 1 ...
##  $ ef50_neg_d         : Factor w/ 4 levels "1","2","3","4": 4 3 2 1 1 1 2 1 1 1 ...
##  $ priormi_21         : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_22         : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_23         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priormi_24         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ age_d              : Factor w/ 5 levels "5","6","7","8",..: 3 2 2 3 3 3 2 4 3 4 ...
##  $ ua_nmi7            : Factor w/ 2 levels "0","1": 1 1 2 1 2 1 1 2 2 2 ...
##  $ creatpre_2         : Factor w/ 10 levels "0.69999999","0.80000001",..: 5 3 6 6 6 6 4 6 4 4 ...
##  $ rf_2               : Factor w/ 7 levels "0","0.10000002",..: 2 1 3 3 3 3 1 3 1 1 ...
##  $ rf_3               : Factor w/ 2 levels "0","0.39999998": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priority_21        : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 2 ...
##  $ priority_22        : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ dm3cat             : Factor w/ 2 levels "2","3": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_22          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_22_2        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ efvalue_r          : Factor w/ 18 levels "24","29","37",..: 1 4 6 8 18 12 7 14 15 10 ...
##  $ ef50_neg_r         : Factor w/ 8 levels "0","3","4","5",..: 8 5 3 1 1 1 2 1 1 1 ...
##  $ ef50_neg_d_r       : Factor w/ 4 levels "1","2","3","4": 4 3 2 1 1 1 2 1 1 1 ...
##  $ priormi_r          : Factor w/ 4 levels "0","2","3","4": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_21_r       : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_22_r       : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 2 1 1 1 ...
##  $ priormi_23_r       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priormi_24_r       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ age_r              : Factor w/ 25 levels "43.04723","51.405888",..: 8 2 5 15 14 9 7 18 11 19 ...
##  $ age_d_r            : Factor w/ 5 levels "5","6","7","8",..: 3 2 2 3 3 3 2 4 3 4 ...
##  $ ua_r               : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 2 ...
##  $ ua_nmi7_r          : Factor w/ 2 levels "0","1": 1 1 2 1 2 1 1 2 2 2 ...
##  $ chf_r              : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ chf2cat_r          : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ creatpre_2_r       : Factor w/ 10 levels "0.69999999","0.80000001",..: 5 3 6 6 6 6 4 6 4 4 ...
##  $ rf_2_r             : Factor w/ 7 levels "0","0.10000002",..: 2 1 3 3 3 3 1 3 1 1 ...
##  $ rf_3_r             : Factor w/ 2 levels "0","0.39999998": 1 1 1 1 1 1 1 1 1 1 ...
##  $ priority_r         : Factor w/ 3 levels "1","2","3": 2 1 2 2 2 3 2 2 2 2 ...
##  $ priority_21_r      : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 2 ...
##  $ priority_22_r      : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ sex_r              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ prcabg_r           : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ copd_r             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dmtx_r             : Factor w/ 4 levels "0","1","2","3": 1 3 3 2 2 3 3 3 2 4 ...
##  $ dm3cat_r           : Factor w/ 2 levels "2","3": 1 1 1 1 1 1 1 1 1 2 ...
##  $ dm3cat_22_r        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ anyvad_r           : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ vad_r              : Factor w/ 4 levels "0","1","2","3": 1 1 1 2 1 1 1 1 1 1 ...
##  $ hyper_r            : Factor w/ 3 levels "0","1","2": 1 2 3 2 2 2 2 2 3 2 ...
##  $ hyperyn_2_r        : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ prptca6_r          : Factor w/ 2 levels "0","2": 2 1 1 1 2 2 1 1 1 1 ...
##  $ lm3cat_r           : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 3 1 1 1 2 ...
##  $ lm2cat_r           : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 2 ...
##  $ chf_nyha_iv        : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ chf_nyha_iv_r      : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ smoker_r           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ cvd                : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ cvd_r              : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ htcm_d             : Factor w/ 6 levels "12","13","15",..: 5 5 6 6 5 4 5 6 5 5 ...
##  $ htcm_r             : Factor w/ 22 levels "122.1","137.11",..: 10 14 22 21 17 9 15 20 15 12 ...
##  $ htcm_d_r           : Factor w/ 6 levels "12","13","15",..: 5 5 6 6 5 4 5 6 5 5 ...
##  $ mitral_insuff      : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ mitral_insuff_r    : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ carotid_sten       : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ carotid_sten_r     : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ pvd                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ pvd_r              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ bmi_squared        : Factor w/ 25 levels "593.55017","603.56079",..: 2 19 15 18 11 13 21 16 8 7 .
##  $ bmi_r              : Factor w/ 25 levels "24.362886","24.567474",..: 2 19 15 18 11 13 21 16 8 7 .
##  $ bmi_squared_r      : Factor w/ 25 levels "593.55023","603.56079",..: 2 19 15 18 11 13 21 16 8 7 .
##  $ novsl_r            : Factor w/ 2 levels "2","3": 2 2 1 1 1 2 2 1 1 2 ...
##  $ readmit_1y_yn_state: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
##  $ anyakin            : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ creatcat           : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ lm50               : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 2 ...
##  $ lowoutput          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ emerg              : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ urg                : Factor w/ 2 levels "0","1": 2 1 2 2 2 1 2 2 2 2 ...
##  $ elec               : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ bmicat             : Factor w/ 5 levels "2","3","4","5",..: 1 3 2 3 2 2 4 3 2 2 ...
##  $ bmi2               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ bmi3               : Factor w/ 2 levels "0","1": 1 1 2 1 2 2 1 1 2 2 ...
##  $ bmi4               : Factor w/ 2 levels "0","1": 1 2 1 2 1 1 1 2 1 1 ...
##  $ bmi5               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ bmi6               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ lvedpm             : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 2 1 1 ...
##  $ anemiapre          : Factor w/ 2 levels "0","1": 2 2 1 2 1 2 1 1 2 1 ...
##  $ lof1               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ ptimenumban        : Factor w/ 22 levels "0","12.4","16",..: 19 16 4 6 12 21 1 9 5 15 ...
##  $ ctimenumban        : Factor w/ 20 levels "0","13","13.166667",..: 15 2 7 2 10 17 1 4 6 13 ...
##  $ cardtimenumban     : Factor w/ 17 levels "0","2","2.4000001",..: 7 10 12 14 5 17 1 2 3 15 ...
##  $ cardblood          : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 1 2 2 2 ...
##  $ aoxcon             : Factor w/ 2 levels "0","1": 2 1 2 2 1 2 1 2 2 2 ...
##  $ ultrafilyn         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ cabg               : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ cabgvalve          : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##   [list output truncated]
```

```r
df.set1.final.noInteractions <- data.frame(df.factors.only, df.set1.2[,colnames(df.set1.2) %in% cont.va
#Double check to make sure levels are >1 and not too big
df.set1.final.levels <- sapply(df.set1.final[,sapply(df.set1.final, is.factor)], nlevels)


endings <- c("_min", "_avg", "_max", "_first", "_last", "_count", "_in_ed", "_duration",
           "_level_diff")
start_labels <- c("cubic_", "quad_", "cub_")
match_labels <- c("grace_score", "lace_score", "charlson_deyo_score", "index_los",
                  "hospital_score", "age_at_admit", "los_new", "los", "ck_level_max2",
                  "creatinine_diff","ed_visit_prior_30_days_minutes_i", "year")
```

```r
clean_data <- function(df, excl.vars, endings, start_labels, match_labels, include_interactions){
  print("In clean_data function")
  names(df) <- tolower(names(df))
  #exclude inputted variables
  df.2 <- na.omit(df[, ! colnames(df) %in% excl.vars])

  str(df.2)
  variable.names <- names(df.2)

  #Get interaction variables from df
  interaction.vars <- get_interactions(df.2)

  #data frame without interactions
  df.no.interactions <- df_excl_vars(df.2, interaction.vars)

  #Get continous variables
  cont.vars <- get_cont_vars(df.no.interactions, endings, start_labels, match_labels)

  #data frame without continous variables (only factored variables left)
  df.raw.factors <- df_excl_vars(df.no.interactions, cont.vars)

  #create data frame as factors
  df.as.factors <- data.frame(lapply(df.raw.factors, factor))
  str(df.as.factors)

  #Drop the factored variables with only 1 level
  factor.levels <- sapply(df.as.factors[,sapply(df.as.factors, is.factor)], nlevels)
  drop_factors <- factors_w_1level(factor.levels)

  df.factors.only <- df_excl_vars(df.as.factors, drop_factors)

  #final data frame with factored variables and continous variables
  df.final <- data.frame(df.factors.only, df.2[, colnames(df.2) %in% cont.vars], df.2[, colnames(df.2)

  str(df.final)

  #final data frame with factored variables and continous variables
  df.final.noInteractions <- data.frame(df.factors.only, df.2[,colnames(df.2) %in% cont.vars])

  str(df.final.noInteractions)

  if(include_interactions == "y"){
    return(df.final)
  }
  else if (include_interactions == "n"){
    return(df.final.noInteractions)
  }
}
```

```r
y <- df.set1.final$dead #separate outcome var
X <- df_excl_vars(df.set1.final, df.set1.final$obleed)
```

```r
#Function to split train and test set
create_train_test <- function(df.final){
  print("Splitting train and test set")
  set.seed(539847)
  #Split train and test set (80%/20%)
  smp_size <- floor(0.8 * nrow(df.final))
  train_idx <- sample(1:nrow(df.final), size = smp_size)

  train_set <- df.final[train_idx,] #80%
  test_set <- df.final[-train_idx,] #20%

  #Create list to hold both the train and test and return df
  train_test <- list("train" = train_set, "test" = test_set)
  return(train_test)
}
```

```r
#train and test set with interactions
d <- create_train_test(df.set1.final)
```

```
## [1] "Splitting train and test set"
```

```r
trainset <- d$train
testset <- d$test
```

```r
#train and test set without interactions
d_noInteractions <- create_train_test(df.set1.final.noInteractions)
```

```
## [1] "Splitting train and test set"
```

```r
trainset_noInt <- d_noInteractions$train
testset_noInt <- d_noInteractions$test
```

```r
#Create helper function for getting ROC/AUROC details
get_auroc <- function(dataset, pred_set){
  ROC.mod <- pROC::roc(dataset$dead, pred_set)
  AUROC.mod <- pROC::auc(ROC.mod)
  CI.AUROC.mod <- pROC::ci.auc(AUROC.mod)
  return(AUROC.mod)
}
```

```r
#Need to figure out how to run function without specifying "dead"
#Want to add outcome_var as a parameter
run_randomForest <- function(train_set, test_set){
  set.seed(539847)
  print("Running Random Forest model")

  #Train the random forest model and predict on testset
  rf.model <- randomForest(train_set$dead ~., data = train_set)
  rf.pred <- predict(rf.model, test_set, type = "prob")[,2]

  #Get the corresponding C-statistic for testset (AUROC)
  AUROC.rf <- get_auroc(test_set, rf.pred)

  #Get the corresponding C-statistic for trainset (AUROC)
  rf.predTrain <- predict(rf.model, type = "prob")[,2]
  AUROC.rfTrain <- get_auroc(train_set, rf.predTrain)

  #Get the importance plot/look at variable plots
  rf.ImpVars <- importance(rf.model)
  varImpPlot(rf.model)

  #Create list of AUROC for both trainset and testset
  rf.stats <- list(AUROC.rfTrain, AUROC.rf)
  names(rf.stats) <- c("AUROC train_set", "AUROC test_set")

  return(rf.stats)
}
```

```r
#run random Forest model
rf.model.stats <- randomForest(trainset_noInt, test_set = testset_noInt)
rf.model.stats
```

```
##
## Call:
##  randomForest(x = trainset_noInt, test_set = testset_noInt)
##                Type of random forest: unsupervised
##                      Number of trees: 500
## No. of variables tried at each split: 13
```

```r
run_specific_mod <- function(x, y, x_test, spec_mod, train_set, test_set){
  #Train specific model
  model <- glmnet(x, y, alpha = spec_mod, family= "binomial")
  plot(model, xvar = "lambda") #plot of the training model
  cv.out<- cv.glmnet(x, y, alpha = spec_mod, family= "binomial") #10-fold cross-validation
  bestlam = cv.out$lambda.min

  #Check performance on validation set
  pred <- predict(model, s= bestlam, newx= x_test, type= "response") #uses best lambda to predict test
  AUROC <- get_auroc(test_set, as.numeric(pred))

  #Check performance on train set
  predTrain <- predict(model, s= bestlam, newx= x, type= "response")
  AUROC.Train <- get_auroc(train_set, as.numeric(predTrain))

  #Get top 10 coefficients in the positive direction and negative direction
  negativeCoef <- tidy(coef(cv.out, s="lambda.min")) %>% arrange(value) %>% head(10)
  positiveCoef <- tidy(coef(cv.out, s="lambda.min")) %>% arrange(desc(value)) %>% head(10)

  #Export the top 10 coefficients to csv files depending on the model
  if (spec_mod == 1){
    write.csv(positiveCoef, paste("/Users/parinithakompala/Desktop/", "LassoPosCoef12.csv", sep=""))
    write.csv(negativeCoef, paste("/Users/parinithakompala/Desktop/", "LassoNegCoef12.csv", sep=""))
  }
  else if (spec_mod == 0.5){
    write.csv(positiveCoef, paste("/Users/parinithakompala/Desktop/", "ElasticNetPosCoef12.csv", sep=""
    write.csv(negativeCoef, paste("/Users/parinithakompala/Desktop/", "ElasticNetNegCoef12.csv", sep=""
  }

  else if (spec_mod == 0){
    write.csv(positiveCoef, paste("/Users/parinithakompala/Desktop/", "RidgePosCoef12.csv", sep=""))
    write.csv(negativeCoef, paste("/Users/parinithakompala/Desktop/", "RidgeNegCoef12.csv", sep=""))
  }

  stats <- list(AUROC.Train, AUROC, positiveCoef, negativeCoef)
  return(stats)
}
```

```
#Main function for running lasso, elastic net, and ridge regression
run_regressions <- function(train_set, test_set, run_lasso= "y", run_elasticNet = "y", run_ridge = "y")
  set.seed(539847)
  trainset <- train_set
  testset <- test_set

  #lasso training and test data turned into matrixes (Double check to find a different way to do this)
  x_train <- model.matrix(dead ~.-1, data= trainset)
  x_test <- model.matrix(dead ~.-1, data= testset)
  y_train <- trainset$dead
  y_test <- testset$dead

  #List to hold all the AUROC info from all the models
  all.stats <- list()

  #Run Lasso Regression
  if (run_lasso == "y"){
    print("Running Lasso Regression")

    lasso.stats <- run_specific_mod(x= x_train, y= y_train, x_test = x_test, spec_mod = 1, train_set= t
    names(lasso.stats) <- c("AUROC Lasso train_set", "AUROC Lasso test_set", "Lasso Negative coefficien

    all.stats <- append(all.stats, lasso.stats)
  }

  if (run_elasticNet == "y"){
    print("Running Elastic Net Regression")

    elasticNet.stats <- run_specific_mod(x= x_train, y= y_train, x_test = x_test, spec_mod = 0.5, train
    names(elasticNet.stats) <- c("AUROC Elastic Net train_set", "AUROC Elastic Net test_set",
                                 "Elastic Net Negative coefficients", "Elastic Net Positive coefficient

    all.stats <- append(all.stats, elasticNet.stats)
  }

  if (run_ridge == "y"){
    print("Running Ridge Regression")
    ridge.stats <- run_specific_mod(x= x_train, y= y_train, x_test = x_test, spec_mod = 0, train_set= tr
    names(ridge.stats) <- c("AUROC Ridge train_set", "AUROC Ridge test_set", "Ridge Negative coefficien

    all.stats <- append(all.stats, ridge.stats)
  }

  return(all.stats)
}
```

```
regression.mods.stats <- lm(dead~obleed, data= trainset)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
#SVM Linear and Radial models
run_svm <- function(train_set, test_set){
  set.seed(539847)
  print("Running Support Vector Machine model")

  #Run Linear svm model
  svm.linear <- e1071::svm(dead~., data=train_set, kernel= "linear")
  svm.linearPred <- predict(svm.linear, newdata= test_set)

  #Test linear svm model with train data
  svm.linearPredTrain <- predict(svm.linear, train_set)


  #Run svm with radial kernel (non-linear)
  svm.radial <- e1071::svm(dead~., data=train_set, kernel= "radial")
  svm.radialPred <- predict(svm.radial, newdata= test_set)

  #Test radial svm model with train data
 svm.radialPredTrain <- predict(svm.radial, newdata= train_set)


  #Get confusion matrices and accuracies
  svm.linear.confusionMat <- table(svm.linearPred, test_set$dead)
  svm.radial.confusionMat <- table(svm.radialPred, test_set$dead)
 svm.linearAccuracy <- mean(svm.linearPred == test_set$dead)
  svm.radialAccuracy <- mean(svm.radialPred == test_set$dead)

 svm.stats <- list(svm.linear.confusionMat, svm.radial.confusionMat, svm.linearAccuracy,
                   svm.radialAccuracy)
  names(svm.stats) <- c("SVM linear Confusion Matrix", "SVM radial Confusion Matrix", "SVM linear accura
                        "SVM radial accuracy")

 write.csv(svm.linear.confusionMat, paste("/Users/parinithakompala/Desktop", "svmLinearConfusionmat.csv
  write.csv(svm.radial.confusionMat, paste("/Users/parinithakompala/Desktop", "svmRadialConfusionmat.cs

  return(svm.stats)
}
```

```
#Run svm models
svm.model.stats <- run_svm(train_set = trainset, test_set= testset)
```

```
## [1] "Running Support Vector Machine model"
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'i_il6post3cat3' and 'i_il6diff3cat3' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'i_il6post3cat3' and 'i_il6diff3cat3' constant. Cannot scale data.
```

```
svm.model.stats2 <- run_svm(train_set = trainset_noInt, test_set = testset_noInt)
```

```
## [1] "Running Support Vector Machine model"
```

```
#SVM Linear and Radial models
run_svm <- function(train_set, test_set, cont.vars){
  set.seed(539847)
  print("Running Test Support Vector Machine model")

  #Coerce factored variables into int/num in test_set
  #Also separating outcome variable into 0--> not readmitted and 1--> readmitted
  omit.var.levels <- c("dead")

  trainset2 <- na.omit(train_set[, colnames(train_set) %in% cont.vars])

  for (i in colnames(train_set[, sapply(train_set, is.factor)])){
    for (level in unique(train_set[, i])){
      trainset2[paste(i, level, sep = "_")] =
        as.integer(ifelse(train_set[, i] == level, 1, 0))
    }
  }

  type.col.num.train <- grep("dead",names(trainset2))
  trainset2 <- na.omit(trainset2[, ! colnames(trainset2) %in% omit.var.levels])

  testset2 <- na.omit(test_set[, colnames(test_set) %in% cont.vars])

  for (i in colnames(test_set[, sapply(test_set, is.factor)])){
    for (level in unique(test_set[, i])){
      testset2[paste(i, level, sep = "_")] =
        as.integer(ifelse(test_set[, i] == level, 1, 0))
    }
  }

  type.col.num.test <- grep("dead",names(testset2))
  testset2 <- na.omit(testset2[, ! colnames(testset2) %in% omit.var.levels])

  for (name in names(trainset2[, ! colnames(trainset2) %in% names(testset2)])) {
    v <- vector(mode = "numeric", dim(testset2)[1])
    testset2[name] <- v
  }

  cv.gbm <- gbm(dead ~ ., data = trainset2, cv.folds = 10, distribution = "bernoulli")
  cv.gbm.pred <- predict.gbm(cv.gbm, testset2, n.trees= best.iter, type= "response")
  ROC.cv.gbm <- pROC::roc(testset2$dead, cv.gbm.pred)
```

```r
  AUROC.cv.gbm <- pROC::auc(ROC.cv.gbm)


  #Run Linear svm model
  svm.linear <- e1071::svm(dead~., data=trainset2, kernel= "linear")
  svm.linearPred <- predict(svm.linear, newdata= testset2, decision.values= TRUE, probability= TRUE)
  ROC.linearPred <- pROC::roc(testset2$dead, svm.linearPred)
  AUROC.linearPred <- pROC::auc(ROC.linearPred)

  #Test linear svm model with train data
  svm.linearPredTrain <- predict(svm.linear, train_set)
  predict(rf.model, test_set, type = "prob")[,2]

  #Run svm with radial kernel (non-linear)
  svm.radial <- e1071::svm(dead~., data=trainset2, kernel= "radial")
  svm.radialPred <- predict(svm.radial, newdata= testset2, decision.values= TRUE, probability= TRUE)
  ROC.radialPred <- pROC::roc(testset2$dead, svm.radialPred)
  AUROC.radialPred <- pROC::auc(ROC.radialPred)

  #Test radial svm model with train data
  svm.radialPredTrain <- predict(svm.radial, newdata= train_set)


  #Get confusion matrices and accuracies
  svm.linear.confusionMat <- table(svm.linearPred, testset2$dead)
  svm.radial.confusionMat <- table(svm.radialPred, testset2$dead)
  svm.linearAccuracy <- mean(svm.linearPred == test_set$dead)
  svm.radialAccuracy <- mean(svm.radialPred == test_set$dead)

  svm.stats <- list(svm.linear.confusionMat, svm.radial.confusionMat, svm.linearAccuracy,
                    svm.radialAccuracy)
  names(svm.stats) <- c("SVM linear Confusion Matrix", "SVM radial Confusion Matrix", "SVM linear accura
                        "SVM radial accuracy")

  svm.stats <- list(svm.linear.confusionMat, svm.radial.confusionMat, AUROC.linearPred,  AUROC.radialPre
  names(svm.stats) <- c("SVM linear confusion matrix", "SVM radial confusion matrix","SVM linear AUROC"

  #Export confusion matrices to csv files
  write.csv(svm.linear.confusionMat, paste("/Users/parinithakompala/Desktop", "svmLinearConfusionmat.csv
  write.csv(svm.radial.confusionMat, paste("/Users/parinithakompala/Desktop", "svmLinearConfusionmat.csv

  return(svm.stats)
}
```

```r
run_gbm <- function(train_set, test_set, cont.vars){
  print("Running Gradient Boosting Model")
  #Coercing factored variables back into int/num in train_set for gradient boosting to run
  #Also separating outcome variable into 0--> not readmitted and 1--> readmitted
  omit.var.levels <- c("dead")

  trainset2 <- na.omit(train_set[, colnames(train_set) %in% cont.vars])

  for (i in colnames(train_set[, sapply(train_set, is.factor)])){
    for (level in unique(train_set[, i])){
```

```
    trainset2[paste(i, level, sep = "_")] =
      as.integer(ifelse(train_set[, i] == level, 1, 0))
  }
}

type.col.num.train <- grep("dead",names(trainset2))
trainset2 <- na.omit(trainset2[, ! colnames(trainset2) %in% omit.var.levels])

#Coerce factored variables into int/num in test_set
#Also separating outcome variable into 0--> not readmitted and 1--> readmitted
testset2 <- na.omit(test_set[, colnames(test_set) %in% cont.vars])

for (i in colnames(test_set[, sapply(test_set, is.factor)])){
  for (level in unique(test_set[, i])){
    testset2[paste(i, level, sep = "_")] =
      as.integer(ifelse(test_set[, i] == level, 1, 0))
  }
}

type.col.num.test <- grep("dead",names(testset2))
testset2 <- na.omit(testset2[, ! colnames(testset2) %in% omit.var.levels])

for (name in names(trainset2[, ! colnames(trainset2) %in% names(testset2)])) {
  v <- vector(mode = "numeric", dim(testset2)[1])
  testset2[name] <- v
}


print("In Gradient boosting function, finished coercing vars, actually running gbm model now")
#Train model using 10-fold cross validation
set.seed(539847)
cv.gbm <- gbm(dead ~ ., data = trainset2, cv.folds = 10, distribution = "bernoulli")
best.iter <- gbm.perf(cv.gbm, method="cv") #get the number of trees with best performance

#get most influential variables in gbm model
all.imp.vars <- summary(cv.gbm, n.trees=best.iter)
imp.vars <- all.imp.vars[all.imp.vars$rel.inf > 0, ][2]

#trainset predictions
cv.gbm.predTrain <- predict.gbm(cv.gbm, trainset2, n.trees= best.iter, type= "response")
ROC.cv.gbmTrain <- pROC::roc(trainset2$dead, cv.gbm.predTrain)
AUROC.cv.gbmTrain <- pROC::auc(ROC.cv.gbmTrain)

#testset predictions
cv.gbm.pred <- predict.gbm(cv.gbm, testset2, n.trees= best.iter, type= "response")
ROC.cv.gbm <- pROC::roc(testset2$dead, cv.gbm.pred)
AUROC.cv.gbm <- pROC::auc(ROC.cv.gbm)

#Create list of AUROC for both trainset and testset and variable influence for cv.gbm
cv.gbm.stats <- list(AUROC.cv.gbmTrain, AUROC.cv.gbm, imp.vars)
names(cv.gbm.stats) <- c("AUROC train_set", "AUROC test_set", "Relative influence of variables > 0")

#Export the influential variables to csv file
```

```
  write.csv(imp.vars, paste("/Users/parinithakompala/Desktop", "gbmImpVars12.csv", sep=""))

  return(cv.gbm.stats)
}
```

```
gbm.stats <- lm(dead~obleed, data= trainset)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
#export results to a csv file
AUROC.rfTrain <- rf.model.stats$`AUROC train_set`[1]
AUROC.rfTest <- rf.model.stats$`AUROC test_set`[1]

AUROC.lassoTrain <- regression.mods.stats$`AUROC Lasso train_set`[1]
AUROC.lassoTest <- regression.mods.stats$`AUROC Lasso test_set`[1]

AUROC.elasNetTrain <- regression.mods.stats$`AUROC Elastic Net train_set`[1]
AUROC.elasNetTest <- regression.mods.stats$`AUROC Elastic Net test_set`[1]

AUROC.ridgeTrain <- regression.mods.stats$`AUROC Ridge train_set`[1]
AUROC.ridgeTest <- regression.mods.stats$`AUROC Ridge test_set`[1]

Acc.svmLinear <- svm.model.stats$`SVM linear accuracy`[1]
Acc.svmRadial <- svm.model.stats$`SVM radial accuracy`[1]

AUROC.gradBoostTrain <- gbm.stats$`AUROC train_set`[1]
AUROC.gradBoostTest <- gbm.stats$`AUROC test_set`[1]
```

```
dataset_number <- 1

mat.results <- cbind(dataset_number, AUROC.rfTrain, AUROC.rfTest, AUROC.lassoTrain,
                     AUROC.lassoTest, AUROC.elasNetTrain, AUROC.elasNetTest, AUROC.ridgeTrain,
                     AUROC.ridgeTest,
                      Acc.svmLinear, Acc.svmRadial)
write.csv(mat.results, paste("/Users/parinithakompala/Desktop", "results", ".csv", sep=""))
```

```
#Want to make sure that I increment dataset_num
export_results <- function(rf.stats, regression.stats, svm.stats, gbm.stats, file_name){
  AUROC.rf <- cbind()
  AUROC.lasso <- cbind()
  AUROC.elasNet <- cbind()
  AUROC.ridge <- cbind()
  AUROC.svm <- cbind()
  #AUROC.gradBoost <- cbind()

  print("Exporting results")
  AUROC.rfTrain <- rf.stats$`AUROC train_set`[1]
  AUROC.rfTest <- rf.stats$`AUROC test_set`[1]

  AUROC.lassoTrain <- regression.stats$`AUROC Lasso train_set`[1]
  AUROC.lassoTest <- regression.stats$`AUROC Lasso test_set`[1]

  AUROC.elasNetTrain <- regression.stats$`AUROC Elastic Net train_set`[1]
```

```r
AUROC.elasNetTest <- regression.stats$`AUROC Elastic Net test_set`[1]

AUROC.ridgeTrain <- regression.stats$`AUROC Ridge train_set`[1]
AUROC.ridgeTest <- regression.stats$`AUROC Ridge test_set`[1]

AUROC.svmLinear <- svm.stats$`SVM linear AUROC`[1]
AUROC.svmRadial <- svm.stats$`SVM radial AUROC`[1]

#Acc.svmLinear <- svm.stats$`SVM linear accuracy`[1]
#Acc.svmRadial <- svm.stats$`SVM radial accuracy`[1]

#AUROC.gradBoostTrain <- gbm.stats$`AUROC train_set`[1]
#AUROC.gradBoostTest <- gbm.stats$`AUROC test_set`[1]

#AUROC.rf <- append(AUROC.rf, AUROC.rfTrain)

AUROC.lasso <- append()
AUROC.elasNet <- append()
AUROC.ridge <- append()
AUROC.svm <- append()
#AUROC.gradBoost <- append()



cur.mat.stats <- cbind(AUROC.rfTrain, AUROC.rfTest
                       ,AUROC.lassoTrain,
                       AUROC.lassoTest, AUROC.elasNetTrain, AUROC.elasNetTest, AUROC.ridgeTrain,
                       AUROC.ridgeTest, AUROC.gradBoostTrain,
                       #AUROC.gradBoostTest,
                       AUROC.svmLinear, AUROC.svmRadial)

#prev.mat.stats <- prev.mat
mat.results <- cur.mat.stats
dimnames(mat.results)[[2]] <- c("AUROC Random Forest Train set", "AUROC Random Forest Test set",
                                "AUROC Lasso Train set", "AUROC Lasso Test set","AUROC Elastic Net Tra
                                "AUROC Elastic Net Test set", "AUROC Ridge Train set",
                                "AUROC Ridge Test set", "AUROC Gradient Boosting Train set",
                                "AUROC Gradient Boosting Test set", "AUROC SVM Linear model",
                                "AUROC SVM Radial model")
fileName <- file_name

write.csv(mat.results, paste("/Users/parinithakompala/Desktop", "results", ".csv", sep=""))
if (dataset_num == 1){

  print("dataset num <= 1")
  write.csv(mat.results, paste("/Users/parinithakompala/Desktop", file_name, ".csv", sep=""))
  print("Exporting results to", paste(("/Users/parinithakompala/Desktop/results.csv")))

  prev.mat.stats <- mat.results
  return(prev.mat.stats)
}
else if (dataset_num > 1){
  prev.mat.stats <- prev.mat
```

```
    mat.results <- rbind(prev.mat.stats, cur.mat.stats)

   print("dataset num > 1")

    print("Exporting results to", paste(("/Users/parinithakompala/Desktop/results.csv")))
   write.csv(mat.results, paste("/Users/parinithakompala/Desktop/", file_name, ".csv", sep=""))
    return(mat.results)
}
  prev.mat.stats <- mat.results
}
```

```
#run main script
main <- function(df.set, filename){
  print(paste("Running on dataset number", i))
  #will have to set these as a parameter to this function
  excl.vars <- tolower(c("_Imputation_", "x_imputation_","MRN", "PERSON_ID", "SSN", "FIRST_NAME", "LAST
                         "DISCHARGE_DATETIME", "ADMIT_DATE", "DISCHARGE_DATE", "INDEX_ADMIT_DATE", "INDE
                         "INDEX_VISIT_OCCURENCE_ID", "VISIT_OCCURENCE_ID", "ED_VISIT_PRIOR_30DAYS_TIME_
                         "READMISSION", "Readmissions_sum", "Seq_ID", "Flg_30d_sum", "previous_yr_sum",
                         "race_ethn", "RACE2", "gender2", "HEMOGLOBIN_diff", "BNP_LEVEL_DIFF", "ed_visi
                         "visit_occurrence_id", "gap", "flg_30d", "readmissions", "previous_yr",
                         "more_previous_yr", "previous_30_day", "readmit_record_30_day_flag",
                         "planned_readmit_record_30_day_fl", "enrichd_score", "readmit_present_30_day_f

  endings <- c("_min", "_avg", "_max", "_first", "_last", "_count", "_in_ed", "_duration",
               "_level_diff")
  start_labels <- c("cubic_", "quad_", "cub_")
  match_labels <- c("grace_score", "lace_score", "charlson_deyo_score", "index_los",
                    "hospital_score", "age_at_admit", "los_new", "los", "ck_level_max2",
                    "creatinine_diff","ed_visit_prior_30_days_minutes_i", "year")


  #run function to get name of data set
  #testing for right now
  df.set1 <- read.csv(paste(folder, "/set1.csv", sep=""))
  df.set <- df.set
  names(df.set) <- tolower(names(df.set))

  #get continous vars
  df.set1.2 <- na.omit(df.set[, ! colnames(df.set) %in% excl.vars])
  interaction.vars <- get_interactions(df.set1.2)
  df.no.interactions <- df_excl_vars(df.set1.2, interaction.vars)
  CONT.VARS <- get_cont_vars(df.no.interactions, endings, start_labels, match_labels)


  #run function to clean data and get final data frame
  df.final <- clean_data(df.set1.2, excl.vars, endings = endings, start_labels = start_labels, match_la
  df.final.noInteractions <- clean_data(df.set1.2, excl.vars, endings = endings, start_labels = start_l


  #split data into train and test
  d <- create_train_test(df.final = df.final)
  trainset <- d$train
  testset <- d$test
  #train and test set without interactions
  d_noInteractions <- create_train_test(df.final.noInteractions)
  trainset_noInt <- d_noInteractions$train
  testset_noInt <- d_noInteractions$test

  #run random forest function
  rf.model.stats <- run_randomForest(train_set = trainset_noInt, test_set = testset_noInt)

  #run regression function
  regression.mods.stats <- run_regressions(train_set = trainset, test_set= testset, "y", "y", "y")

  #run svm function
  svm.model.stats <- run_svm(train_set = trainset, test_set= testset, cont.vars= CONT.VARS)
  #svm.test.model.stats <- run_Testsvm(train_set = trainset, test_set= testset)

  #run gradient boosting function
  gbm.stats <- run_gbm(train_set = trainset, test_set = testset, cont.vars = CONT.VARS)

  #run function to put info in excel sheet
  file_name <- filename
  export_results(rf.model.stats, regression.mods.stats, svm.model.stats, gbm.stats, file_name)









  all_model_info <- list(rf.model.stats, regression.mods.stats, svm.model.stats, gbm.stats)
  return(all_model_info)
}
```