Penetration Rates for Prostate Cancer

November 12, 2020

Executive Summary:

Introduction: Prostate cancer is one of the most common cancers among men. The survival rate is vary depending on the stages of the cancer. Prostate cancer has a high probability of being treated if detected at early stages. There are many important characteristics that can predict whether a tumor has penetrated the prostatic capsule. With the results of penetration rate, doctors and nurses would prioritize the patients with the high risk level. On the other hand, if the patients have low rate in penetration, health care workers could find the best treatments for them to slow down prostate growth rate or even treat the patients. With this being said, hostpitals can plan their budget and supplies to focus more on high risk patients and minimize supplies to low risk patients. Knowing penetration rates does not only benefits doctors and nurses, it would also help all men out there to detect if their tumors have penetrated the prostatic capsule. This way, patients would seek professional help at an earlier stage to increase their survival rate. Many questions have proposed in favor of these issues including 1) What variables cause the penetration rate to increase or decrease? 2) What are the likelyhood of penetration rate? 3) How accruate is the model proposed in this analysis? 4) How can we implement the model for everyone to use? and 5) How can we improve the current model? This report will answer all of the questions mentioned above by crucials data analysis process such as data exploratory analysis process to explore the relationship of each variables, model development using multiple logistic regression to built model, and diagnositics process to check the quality of the model. This model will serve the purpose of predicting penetration rate for prostate cancer in men to minimize human errors and provide an effective tool for health care workers.

Methods: A subset of data was provided from the Ohio State University Comprehensive Cancer Center to find the variables to predict if a tumor has penetrated the prostatic capsule. This data set contain a total of 380 subjects, out of which, 153 subjects had a cancer that penetrated the capsule. There are three observations contain missing values. After eleminating missing values to build more quality models, the dataset now have a total of 377 observations. These data contains one identification column with numbers of each subject, two demographic columns including age and race of the subjects, and four health measurements such as the results of digital rectal exam, detection of capsular involvement, prostatic specific antigen value, and total gleason score. Health mesurement variables will be explained in details in the exploretory data analysis section. The goal of this analysis is to find the average tumor penetration of prostatic capsule rate. To accomplish this goal, we will first analyze the relationship between each predictor and target variable, build models using bootstraping method, consider interaction terms to be in the final model, check for the quality of the models, and conclude with an interpretaion of the model output. All analysis will be done in R Studio with R version 3.6.2.

Exploratory Data Analysis: Before picking variables for our model development procress, it is important to access the relationship of each response variable with the target variable. Table A2 shows an descriptive statistics of each independent variable. As seen in this table, the maximum value of Prostatic Specific Antigen value are relatively high compare to the median and mean values suggesting there might be outliers. In addition, there are disproportion weights of each factor in race of the subjects and detection of capsular involvement. This indicates that there might be biased in predicting the outcome if we use these two variables in the models. The rest of the variables seems proportionally balanced and no skewness were detected among them. In terms of the tumor penetration of prostatic capsule, the response variable has a balance proportion of 40% of tumor has penetrated the prostatic capsule out of 377 subjects. Figure 1.1 show the histogram of chances of the tumor penetration of prostatic capsule (1 = penetration, 0 = no penetration).

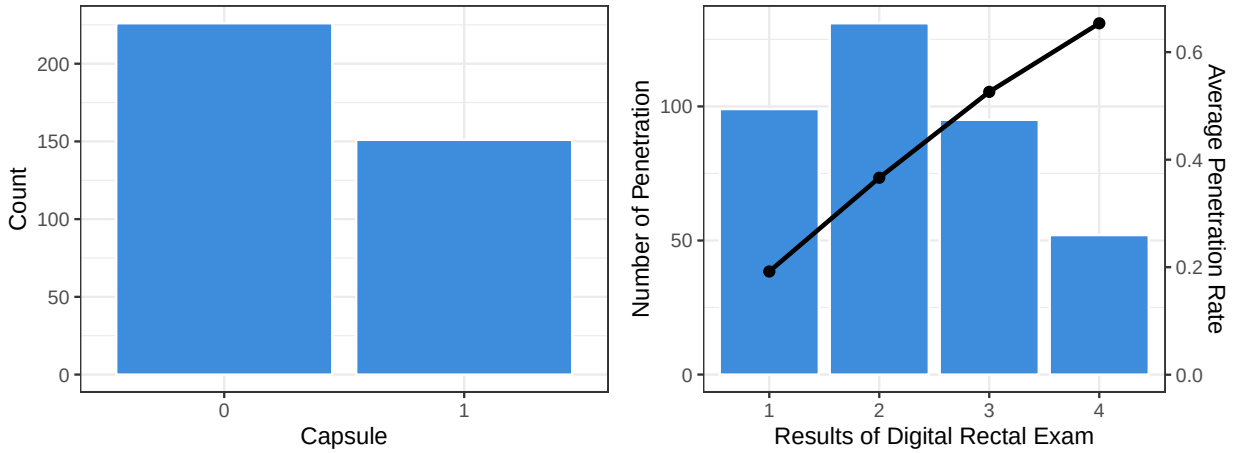

Figure 1

The results of digital rectal exam is a simple exam to check the prostate to determine the size of the prostate and any abnormalities. Clearly, different abnormalities would have different chances of penetration of prostatic capsule. Figure 1.2 shows four categories of the results of digital rectal exam including no nodule, unilobar nodule (left), unilobar nodule (right), and bilobar nodule noting using number from 1 to 4, respectively. Here, this variable is identified as categorical variables with four levels instead of a continuous variable. As seen in plot, these four categories obviously have different average penetration rates. For example, a subject with no nodule would be less likely to penetrate compare to a person with bilobar nodule. It makes sense because if there is no nodule detected in the prostate area, then there is less chances of the tumor will penetrate the prostatic capsule. In addition, Figure 1.2 also depicts the number of subject in each category. Unilobar nodule (left) is the most common and bilobar nodule is the least common among prostate cancer. However, the proportion of each factor for digital rectal exam are not too different. Thus, this variables would provide significant impact for the models.

Detection of capsular involvement can describe if the tumor involving or extending beying the prostate capsule. The tumor is involved when there is capsular involment present. Clearly, the patients who have capsular involement present can be more risky than patient who do not have capsular involment. Figure 2.1 shows the count and proportion of penetration for the detection of capsular involement. This variable has two levels (1 = no, 2 = yes) to suggest if a subject has capsular involvement or not. As seen in the figure, patients with capsule involement would be much

more likely to have tumor penetrated of prostatic capsule compare to patients that do not have capsule involvement. However, the count for each level is not proportional. The majority of the subject have level 1, no capsular involement. Therefore, this variable could be a good variable for our model since it shows a strong signal, however, cautious about the disproportional need to be consider before finalizing the model.

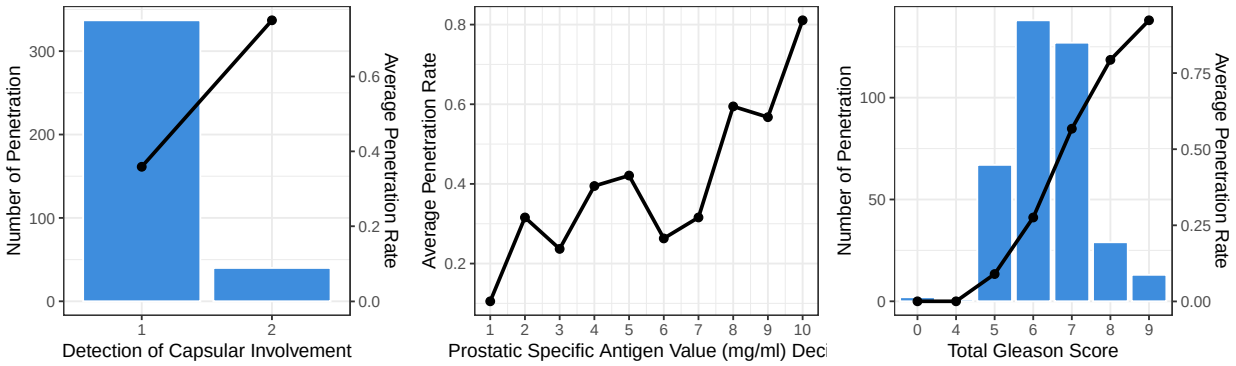

Figure 2

Prostatic Specific Antigen (PSA) can be found in the prostate gland cells. This screening test measure the amount of prostate-specific antigen in patients' blood using miligram per mililitter (mg/ml). Higher PSA level could lead to a high posibility of diagnosting with prostate cancer. Figure 2.2 show a plot of the relationship between PSA and the average penetration of prostatic capsule rate. Here, the data point is being sorted from smallest to highest, divided into 10 equal buckets, and assinged a decide for visualization purposes. As on can see, as the decile number increases by one unit, the chances of penetration rate increases, on average. Since PSA in higher buckets are generally the largest, they have higher risk of getting prostate cancer than PSA in lower buckets. Therefore, this proves that PSA could provide signicant relationship for the likelihood of tumor penetrated of prostatic capsule.

Last but not least, total gleason score can measure the abnormality of the cells to see how the cells are arrnaged using a scale of 1 to 10. Higher scores lead to higher posibility of have penetration in prostatic capsule. On the other hand, cancer cells that looks similar to normal cells can be consider as low risk and have lower scores. Figure **??**.3 show a histogram of total gleason score overlayed by the average risk of having cancer. Total gleason score seems to have a normal distribution with less subjects having low or high scores and most subjects having median score of 6. In addition, the average penetration rate is increasing as the total gleason score is higher. This mean that if a patient's cancer cells looks similar to normal cells, they have less chances of being diagnosed with prostate cancer. On the other hand, if the gleason scores are high, they would have high risk of being diagnosed with prostate cancer. Therefore, this variable can also provide excelent signal for the response variable. Other variables did not mention aboved have little to no relationship with tumor penetration including age and race of the subjects.

After accessing the relationship of predictors and response variable, it is important to check the correlation between predictors. Fortunately, all correlation between predictors are less than 40%. Total gleason score and PSA values have the highest correlation score of 38.60% which is considered as low. This indicate that all variables can be in the same models without raising the multicolineary

issues. However, to make sure, we will check VIF scores after the modeling process. Therefore, we will go ahead and build our model using four variables including results of digital rectal exam ("dpros"), detection ofo capsule involvement ("dcaps"), PSA value ("psa"), and total Gleason score ("gleason").

Model Fitting/Inferences: After four variables were detected to be considered as significant, logistic regression will be used in model development process. Bootstrapping method is a common method for variable selection using random partitions. Here, the models are generated using a combination of each variable using different random partition of training and testing data. An average AUC and AIC of each model was computed after to compare. Table A3 shows the top 10 models with the highest AUC scores. Models with three variables results of digital rectal exam, PSA values, and total Gleason scores seems to be the best model with overall AUC above 80% and low AIC scores compares to other models. Therefore, the model development process will continue using the three variables mentioned previously with a train set of 70% and hold-out test set of 30%.

Before finalizing the model, all combination of interaction terms were added to the model with three variables including results of digital rectal exam, PSA values, and total Gleason score. The results of this model are shown in Table A4. The coefficient of all three variables seems to increase in the model with interaction terms. This proves that the model with interaction terms coefficent has steeper slope than the model without interaction terms (see Table 1). However, p-values for all interaction terms are quite high suggesting that interaction terms are not needed in the model. To further prove this argument, a stepwise model selection was run using the model with interaction terms using AIC as a validation metric. The best model seems to be the model with the original three variables: results of digital rectal exam, PSA values, and total Gleason score. As seen in Table 1, the final model's coefficients are positive consitent with the exploratory analysis. In addition, all variables in this model have low p-values of less than 0.05 and odd ratio confident interval does not contain 1 suggesting all variables are significant. Finally, this model has a low AIC of 248.56, high AUC of 0.82, and high accuracy of 0.77 consistent with high probability of producing accuraate predictions.

term	estimate	std.error	statistic	p.value	OR	2.5 %	97.5 %
(Intercept)	-7.254	1.296	-5.595	0.000	0.001	0.000	0.008
dpros2	0.822	0.461	1.785	0.074	2.276	0.944	5.835
dpros3	1.306	0.483	2.700	0.007	3.689	1.466	9.892
dpros4	1.280	0.577	2.220	0.026	3.598	1.176	11.452
psa	0.041	0.015	2.760	0.006	1.042	1.015	1.075
gleason	0.846	0.200	4.231	0.000	2.331	1.599	3.514

Table 1: Summary regression of final model

To access how good the model fit, ...

Checking the quality of the model is one of the most important step in a data analysis. ...

The student residuals ...

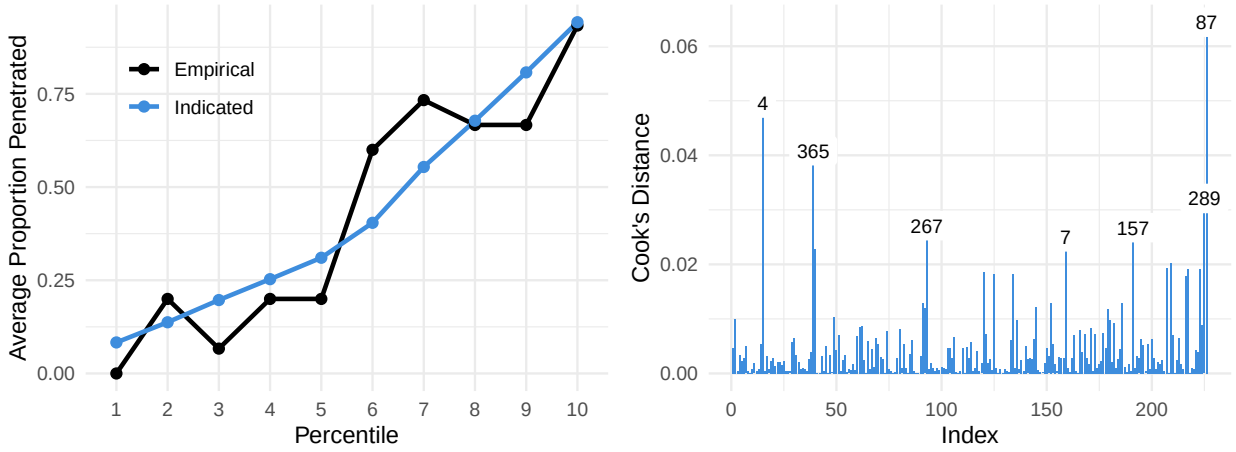

Figure 3

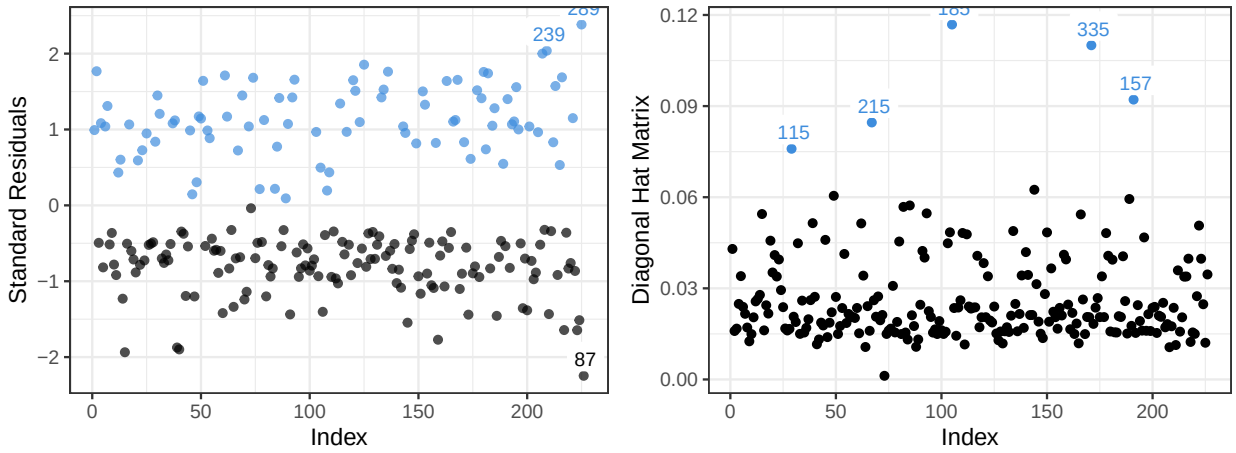

Figure 4

After finishing model development and validation, the model is ready to be interpreted. ...

Conclusion:

Appendix A: Supplemental Tables

Table 2: Summary Statistics for all numerical independent features

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
capsule	377	0.401	0.491	0	0	1	1
age	377	66.066	6.425	47	62	71	79
psa	377	15.256	19.869	0.300	5.000	16.800	139.700
gleason	377	6.379	1.092	0	6	7	9
psa_bins	377	5.472	2.869	1	3	8	10

	random_partition	nrow_train	nrow_test	features	AIC	auc
1	0.900	339	38	dpros,psa,gleason,dcaps	298.077	0.822
2	0.800	301	76	dpros,psa,gleason,dcaps	278.940	0.820
3	0.600	226	151	dpros,psa,gleason	241.799	0.819
4	0.700	263	114	dpros,psa,gleason	260.832	0.818
5	0.600	226	151	dpros,psa,gleason,dcaps	240.567	0.818
6	0.700	263	114	dpros,psa,gleason,dcaps	258.675	0.815
7	0.900	339	38	dpros,psa,gleason	297.050	0.812
8	0.800	301	76	dpros,psa,gleason	277.654	0.809
9	0.600	226	151	psa,gleason,dcaps	251.167	0.805
10	0.900	339	38	dpros,gleason	301.355	0.805

Table 3

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-9.7465	3.8714	-2.52	0.0118
dpros2	1.7503	4.1518	0.42	0.6733
dpros3	2.9024	4.2654	0.68	0.4962
dpros4	4.8459	4.7046	1.03	0.3030
psa	0.2090	0.1227	1.70	0.0886
gleason	1.0748	0.6161	1.74	0.0811
dpros2:psa	-0.0986	0.0638	-1.55	0.1221
dpros3:psa	-0.1225	0.0801	-1.53	0.1260
dpros4:psa	-0.0601	0.0699	-0.86	0.3902
dpros2:gleason	0.0468	0.6584	0.07	0.9434
dpros3:gleason	-0.0354	0.6803	-0.05	0.9585
dpros4:gleason	-0.4452	0.7431	-0.60	0.5491
psa:gleason	-0.0130	0.0168	-0.77	0.4396

Table 4

	capsule	dpros	psa	gleason	rownames
1	0	2	51.20	7	4
2	0	4	31.90	7	7
3	0	4	17.10	9	87
4	1	4	45.30	6	115
5	0	4	25.20	7	121
6	1	1	44.40	6	157
7	1	1	85.40	7	185
8	1	3	17.70	5	196
9	1	4	53.90	6	215
10	1	1	6.70	6	239
11	0	4	25.10	7	254
12	1	4	18.70	5	267
13	1	1	8.90	6	272
14	1	1	6.80	5	289
15	0	3	18.10	8	307
16	1	2	58.00	6	335
17	0	2	48.00	7	365

Table 5

Appendix B: R Code

```r
#### Load packages ####
library(MASS)
library(magrittr)
library(tidyverse)
library(ggcorrplot)
library(car)
library(broom)


#### Functions ####

#' Explore variables function, returning plots of variables count and average proportion
#'
#' @param df dataframe
#' @param xvar independent variable
#' @param count return histogram of count,default is TRUE
#' @param x_axis x axis name
#'
#' @return
#' @export
#'
#' @examples
ExploreVariable <- function(df, xvar, count = TRUE, x_axis){

  group_df <- df %>%
    group_by({{xvar}}) %>%
    summarise(avg_prop_capsule = mean(capsule),
              count = n(),
              .groups = "drop")

  ratio <- max(group_df$count) / max(group_df$avg_prop_capsule)

  if(count == TRUE){
    p <- ggplot(group_df, aes(x = factor({{xvar}}), group = 1)) +
      geom_bar(aes(y = count), stat = "identity", fill = "#3e8ddd", col = "white") +
      geom_point(aes(y = avg_prop_capsule * ratio), size = 2, color = "black") +
      geom_line(aes(y = avg_prop_capsule * ratio), size = 1, color = "black") +
      scale_y_continuous(sec.axis = sec_axis(~./ratio, name = "Average Penetration Rate"))

    y_lab <- "Number of Penetration"

  } else {
    p <- ggplot(group_df, aes(x = {{xvar}}, group = 1)) +
      geom_point(aes(y = avg_prop_capsule), size = 2, color = "black") +
      geom_line(aes(y = avg_prop_capsule), size = 1, color = "black")

    y_lab <- "Average Penetration Rate"
  }
  p +
    labs(y = y_lab, x = x_axis) +
    theme_bw()
}

```

```r
#' Simple binomial regression to run bootstrap for variable selection
#'
#' @param rp random partition
#' @param xvars independent variable/s
#' @param nrounds number of times running the model
#'
#' @return
#' @export
#'
#' @examples
BinaryFit <- function(rp, xvars, nrounds){

  # select varaibles
  df  <- prostate %>% dplyr::select(capsule, all_of(xvars))

  all_auc <- c()
  all_aic <- c()
  all_iteration <- list()
  index <- 1

  for(each_rp in rp){
    for (i in 1:nrounds){
      # split train and test randomly based on proportion
      n <- floor(each_rp * nrow(df))

      train_ind <- sample(seq_len(nrow(df)), size = n)

      train <- df[train_ind, ]

      test <- df[-train_ind, ]

      # build model
      fit <- glm(capsule ~ ., data = train, family = "binomial")

      # predict
      preds <- predict(fit, newdata = test, type = "response")

      test$preds <- preds

      # Validate AUC
      test_roc <- pROC::roc(response = test$capsule, predictor = test$preds)

      auc <- pROC::auc(test_roc)[[1]]

      all_auc <- c(all_auc, auc)
      all_aic <- c(all_aic, fit$aic)
    }
    # iteration log
    all_iteration[[index]] <- data.frame(iteration_date = Sys.time(),
                                         model_type = "glm",
                                         distibution = "binomial",
                                         random_partition = each_rp,
                                         nrow_train = nrow(train),
                                         nrow_test = nrow(test),
                                         nrounds = nrounds,
                                         n_features = length(xvars),
```

```
                                          features = paste(xvars, collapse = ","),
                                          AIC = mean(all_aic),
                                          auc = mean(all_auc))
    index <- index + 1
  }
  all_iteration %<>% bind_rows()

  return(all_iteration)
}

SafelyBinaryFit <- safely(BinaryFit)


#' Function to loop all the models through each variable
#'
#' @param rp
#' @param nrounds
#' @param list_of_xvars
#' @param number_of_xvars
#'
#' @return
#' @export
#'
#' @examples
LoopAllVars <- function(rp, nrounds, list_of_xvars, number_of_xvars){
  # group variables
  vars <- combn(list_of_xvars, number_of_xvars)

  # how many group do we have?
  no_vars <- dim(vars)[[2]]

  # initiate print list and index
  print_list <- list()
  k <- 1

  for (i in 1:no_vars) {
    iteration_log <- SafelyBinaryFit(rp, xvars = vars[,i], nrounds)
    print_list[[k]] <- iteration_log
    k <- k + 1
  }
  # map all separate log to bind them together into a dataframe
  iteration_log <- map(print_list, "result") %>% bind_rows()

  return(iteration_log)
}


#### Load data ####

## Load in prostate data set
prostate <- read.table("data/prostate.txt", header = T)


#### Clean data & Data Pre-processing ####

## check for NA
```

```r
sum(is.na(prostate)) # missing data: 3 in race

## omit NA
prostate %<>%
  na.omit()

## change categorical variables to class factor
prostate %<>%
  modify_at(c("dcaps", "dpros", "race"), as.factor)

## add bins for psa
prostate %<>%
  mutate(psa_bins = ntile(psa, 10))


#### Data Exploratory Analysis ####

## Target Variable - capsule - histogram
ggplot(prostate, aes(factor(capsule))) +
  geom_histogram(stat = "count", fill = "#3e8ddd", col = "white") +
  theme_bw() +
  labs(x = "Capsule",
       y = "Count")

## Predictors
ExploreVariable(prostate, age, count = FALSE, x_axis = "Age")
ExploreVariable(prostate, race, x_axis = "Race")
ExploreVariable(prostate, dpros, x_axis = "Results of Digital Rectal Exam")
ExploreVariable(prostate, dcaps, x_axis = "Detection of Capsular Involvement")
ExploreVariable(prostate, psa_bins, count = FALSE,
                x_axis = "Prostatic Specific Antigen Value (mg/ml)") +
  scale_x_continuous(breaks = seq(1, 10, 1))
ExploreVariable(prostate, gleason, x_axis = "Total Gleason Score")

# after the variable exploratory, we can see that
# - age don't have a relationship with capsule
# - there is little to no relationship with capsule, and unbalance factors
# - dpros has a strong effect of getting penetration by different levels
# - dcaps also has strong effect of getting penetration
# - psa has a strong positive relationship with capsule
# - gleason also has a strong positive relationship with capsule
# we can continue the analysis using 4 variables: dpros, dcaps, psa, and gleason


#### Check Correlation for Continuous variables ####
corr_table <- prostate %>%
  select(age, psa, gleason) %>%
  cor()

ggcorrplot(corr_table, hc.order = TRUE,
           outline.col = "white",
           colors = c("blue", "white", "red"))

# no high correlation between continuous predictors
# all variables can be in the same model

```

```
222
223 #### Model Development ####
224
225 ## parameter for bootstrapping
226
227 xvars <- c("dpros", "psa", "gleason", "dcaps")
228
229 rp_par <- c(0.6, 0.7, 0.8, 0.9)     # random partition parameters
230
231 nrounds <- 10                       # number of rounds per model
232
233 ## Bootstrapping
234
235 # initiate print list
236 print_list <- list()
237
238 # loop
239 for (n_features in 1:4){
240   results <- LoopAllVars(rp = rp_par,
241                          nrounds = nrounds,
242                          list_of_xvars = xvars,
243                          number_of_xvars = n_features)
244   print_list[[n_features]] <- results
245 }
246
247 # bind iteration log into a dataframe
248 iteration_log <- print_list %>% bind_rows()
249
250 # after running the iteration logs, models with variables dpros, psa, and gleason
251 # has the high AUC and low AIC, using train set of 60% prop and test set of 40%
252
253
254 #### Select Model after bootstrap ####
255
256 ## Split data to train/test set using random partition with proportion .7/.3
257 set.seed(1234)
258
259 n <- floor(.6 * nrow(prostate))
260
261 train_ind <- sample(seq_len(nrow(prostate)), size = n)
262
263 train <- prostate[train_ind, ]
264
265 test <- prostate[-train_ind, ]
266
267
268 ## start with initial fit using 4 variables
269 fit1 <- glm(capsule ~ dpros + psa + gleason,
270             family = binomial(link = logit),
271             data = train)
272
273
274 ## Checking for Interaction term
275
276 fit2 <- glm(capsule ~ .*.,
277             family  = "binomial",
```

```
            data    = train %>% dplyr::select(dpros, psa, gleason, capsule))

test$preds2 <- predict(fit2, newdata = test, type = "response")

test_roc <- pROC::roc(response = test$capsule, predictor = test$preds2)
test_auc <- as.data.frame(pROC::auc(test_roc))
aic <- fit2$aic

summary(fit2)

stepAIC(fit2, direction = c("both"))

# although auc for this model is high and AIC is low,
# each coefficient in the model is not statistically significant
# so we're not considering this model to be the final model
# also using model selection function stepAIC,
# interaction terms are not needed in the model

#### Final Model ####

fit_final <- fit1

## generate predictions using test set
test$preds <- predict(fit_final, newdata = test, type = "response")

## create percentile to generate lift chart
test$percentile <- ntile(test$preds, 10)

lift_plot_df <- test %>%
  group_by(percentile) %>%
  summarise("Empirical" = mean(capsule),
            "Indicated" = mean(preds)) %>%
  gather("key", "value", -percentile)

## Lift chart to visualize the quality of the model
ggplot(lift_plot_df, aes(x = factor(percentile), y = value, color = key, group = key
    )) +
  geom_line(size = 1) +
  geom_point(size = 2) +
  theme_minimal() +
  labs(x = "Percentile",
       y = "Average Proportion Penetrated") +
  theme(legend.title = element_blank()) +
  scale_color_manual(values = c("black", "#3e8ddd"))

## compute AIC and AUC of final model
aic <- fit_final$aic
test_roc <- pROC::roc(response = test$capsule, predictor = test$preds)
test_auc <- as.data.frame(pROC::auc(test_roc))

tidy <- broom::tidy(fit_final) %>%
  mutate(OR = exp(fit_final$coefficients)) %>%
  cbind(exp(confint(fit_final))) %>%
  modify_if(is.numeric, round, 4)

## wald test
```

```
aod::wald.test(b = coef(fit_final), Sigma = vcov(fit_final), Terms = 4)


## Generate confusion matrix and accuracy
test %<>%
  mutate(preds = case_when(preds < 0.5 ~ 0.5, TRUE ~ 1.1))

confusion_matrix <- ftable(test$capsule, test$preds)

accuracy <- sum(diag(confusion_matrix))/nrow(test)*100

accuracy


#### Diagnostics ####

#### Outliers test ####
## Bonferonni p-value for most extreme obs
outlierTest(fit_final)


## Influential Observations
influencePlot(fit_final,
              id.method="identify",
              main="Influence Plot",
              sub="Circle size is proportial to Cook's Distance" )

diag <- augment(fit_final) %>%
  mutate(Index = 1:nrow(.))

cutoff <- 4/((nrow(train) - length(fit_final$coefficients) - 2))

diag %<>%
  mutate(high_cooksd = case_when(
    .cooksd > cutoff ~ 1, TRUE ~ 0),
    col_stdresid = case_when(
      .std.resid > 0 ~ 1,
      .std.resid < 0 ~ 0),
    high_hat = case_when(
      .hat > .07 ~ 1,
      TRUE ~ 0))

## cook's distant plot
ggplot(diag, aes(x = Index, y = .cooksd)) +
  geom_bar(stat = "identity", fill = "#3e8ddd", col = "white") +
  labs(y = "Cook's Distance") +
  theme_minimal() +
  geom_label(data = diag %>% filter(.cooksd > cutoff + .005),
             aes(label = .rownames), label.size = NA, size = 3)


## standard residual plot
ggplot(diag, aes(x = Index, y = .std.resid, color = factor(col_stdresid))) +
  geom_point(alpha = 0.7) +
  labs(y = "Standard Residuals") +
  theme_bw() +
```

```
  scale_color_manual(values = c("black", "#3e8ddd")) +
  theme(legend.position = "none") +
  geom_label(data = diag %>% filter(.std.resid > 2 | .std.resid < -2),
             aes(label = .rownames), label.size = NA, size = 3, hjust=0.45, vjust
    =-.15)

## diagonal hat matrix plot
ggplot(diag, aes(x = Index, y = .hat, color = factor(high_hat))) +
  geom_point() +
  labs(y = "Diagonal Hat Matrix") +
  theme_bw() +
  scale_color_manual(values = c("black", "#3e8ddd")) +
  theme(legend.position = "none") +
  geom_label(data = diag %>% filter(.hat > .07),
             aes(label = .rownames), label.size = NA, size = 3, hjust=0.45, vjust
    =-.15)

## student residual distribution
sresid <- studres(fit_final)
hist(sresid, freq=FALSE,
     main="Distribution of Studentized Residuals")
xfit<-seq(min(sresid),max(sresid),length=40)
yfit<-dnorm(xfit)
lines(xfit, yfit, col = "red")
# looks normal

#### Collinearity  ####
vif(fit_final) # variance inflation factors
# all VIF are low = no multicolinearity


#### Find influential point in data ####

inf_points <- (diag %>%
                 filter(.std.resid > 2 | .std.resid < -2 | .cooksd > cutoff | .hat >
    .07))$.rownames %>%
  as.numeric()

prostate[inf_points, ] %>%
  as.data.frame() %>%
  dplyr::select(capsule, dpros, psa, gleason) %>%
  mutate(rownames = inf_points) %>%
  arrange(rownames)
```

Listing 1: Appendix of Code