ECON_390_Final_Project

Rahul Narvekar, Alex Ozhakanat, Alison Zhang, Shivangi Shah, Justin Geletko

12/09/2021

Question 1: Download Data

```
#check to see if the first year exists
exists = file.exists("hcris_raw/rpt1998.csv")
if(!exists) {
  for(i in 1998:2010) {
    #download into local folder
    download.file(paste0("http://www.nber.org/hcris/265-94/rnl_rpt265_94_",i,".csv"), paste0("hcris_raw
    download.file(paste0("http://www.nber.org/hcris/265-94/rnl_nmrc265_94_",i,"_long.csv"), paste0("hcr
    download.file(paste0("http://www.nber.org/hcris/265-94/rnl_alpha265_94_",i,"_long.csv"), paste0("hc
  }
}
print("data downloaded!")
```

```
## [1] "data downloaded!"
```

Question 2: Cleaning The Data

```
#helper function made the name variable features
colNameMaker = function(original) {
  return(strsplit(original, split = "[.]")[[1]][2])
}

#if the file for the csv does not exist, go ahead and generate it
if(!file.exists("hcris_raw/HcrisPanel.csv")) {
  library(stringr)

  #name of all features
  allFeatures = c("rpt_rec_num","facility_name","non_medicare_sessions",
                  "non_medicare_sessions_indirect","avg_weekly_sessions","avg_days_open_per_week",
                  "avg_session_time","num_machines_regular","num_machines_standby",
                  "dialyzer_type","dialyser_reuse_times","epo_total",
                  "epo_cost" ,"epo_rebates","epo_net_cost",
                  "chain_indicator","chain_identity","prvdr_num",
                  "ever_hospital_based","certification_date","report_start_date",
                  "report_end_date","total_treatments_hd","total_costs_hd_housekeeping",
                  "total_costs_hd_machines","total_costs_hd_salaries","total_costs_hd_benefits",
```

```
                "total_costs_hd_drugs","total_costs_hd_supplies","total_costs_hd_labs",
                "total_costs_hd_other","supplies","lab_services",
                "total_treatments_pd","zip_code","state",
                "fy_bgn_dt","fy_end_dt","year")

#initialize a blank data frame
HcrisPanel =  data.frame(matrix(ncol=39, nrow = 0))
colnames(HcrisPanel) = allFeatures

#generate keys from var code csv
varCodes = read.csv('hcris_raw/variable_codes.csv')
keysVarCodes = rep("", length(varCodes))
for(i in 1:NROW(varCodes)) {
  worksheet = varCodes[i,2]
  line = str_pad(varCodes[i,3], 5, pad = "0")
  column = str_pad(varCodes[i,4], 4, pad = "0")
  keysVarCodes[i] = paste0(worksheet, line, column)
}
varCodes$keys = keysVarCodes

#loop through every year and create a data frame to rbind to HcrisPanel
for (j  in 1998:2010) {

  #read in aplha, nmrc, and rpt files for current year
  alpha = read.csv(paste0('hcris_raw/alpha',j,'.csv'))
  nmrc = read.csv(paste0('hcris_raw/nmrc',j,'.csv'))
  rpt = read.csv(paste0('hcris_raw/rpt',j,'.csv'))
  keysAlpha = rep("", length(alpha))
  keysNmrc = rep("", length(nmrc))

  #generate keys for nmrc and alpha
  for(i in 1:NROW(nmrc)) {
    worksheet = nmrc[i,2]
    line = str_pad(nmrc[i,3], 5, pad = "0")
    column = str_pad(nmrc[i,4], 4, pad = "0")
    keysNmrc[i] = paste0(worksheet, line, column)
  }
  nmrc$keys = keysNmrc
  for(i in 1:NROW(alpha)) {
    worksheet = alpha[i,2]
    line = str_pad(alpha[i,3], 5, pad = "0")
    column = str_pad(alpha[i,4], 4, pad = "0")
    keysAlpha[i] = paste0(worksheet, line, column)
  }
  alpha$keys = keysAlpha

  #merge var nmrc and alpha with var code
  alphaWithVars = merge(varCodes[,c(1,5)], alpha[,c(1,5,6)], by = 'keys', all.x = TRUE)
  nmrcWithVars = merge(varCodes[,c(1,5)], nmrc[,c(1,5,6)], by = 'keys', all.x = TRUE)

  #reshape data frames from long to wide
  wideAlpha = reshape(alphaWithVars[,-1], idvar = "rpt_rec_num", timevar = "variable", direction = "w
  wideNmrc = reshape(nmrcWithVars[,-1], idvar = "rpt_rec_num", timevar = "variable", direction = "wid
```

```
    screwedUpMerge = merge(wideAlpha, wideNmrc, by = 'rpt_rec_num', all.x = T)

    #grab required features
    features = c("rpt_rec_num", "alphnmrc_itm_txt.facility_name","itm_val_num.non_medicare_sessions",
      "itm_val_num.non_medicare_sessions_indirect","itm_val_num.avg_weekly_sessions","itm_val_num.avg_da
      "itm_val_num.avg_session_time","itm_val_num.num_machines_regular","itm_val_num.num_machines_standb
      "itm_val_num.dialyzer_type","itm_val_num.dialyser_reuse_times","itm_val_num.epo_total",
      "itm_val_num.epo_cost","itm_val_num.epo_rebates","itm_val_num.epo_net_cost",
      "alphnmrc_itm_txt.chain_indicator","alphnmrc_itm_txt.chain_identity","alphnmrc_itm_txt.prvdr_num"
      "alphnmrc_itm_txt.ever_hospital_based","alphnmrc_itm_txt.certification_date","alphnmrc_itm_txt.rep
      "alphnmrc_itm_txt.report_end_date","itm_val_num.total_treatments_hd","itm_val_num.total_costs_hd_h
      "itm_val_num.total_costs_hd_machines","itm_val_num.total_costs_hd_salaries","itm_val_num.total_cos
      "itm_val_num.total_costs_hd_drugs","itm_val_num.total_costs_hd_supplies","itm_val_num.total_costs_
      "itm_val_num.total_costs_hd_other","itm_val_num.supplies","itm_val_num.lab_services",
      "itm_val_num.total_treatments_pd","alphnmrc_itm_txt.zip_code","alphnmrc_itm_txt.state")
    cleanedMerge = screwedUpMerge[features]

    #drop item_val_nul and alphnmrc_itm_text from variable names
    for(i in 2:length(features)) {
      features[i] = colNameMaker(features[i])
    }
    colnames(cleanedMerge) = features

    #merge in fy bgn and end date from rpt data set
    currentYear = merge(cleanedMerge, rpt[,c(1,13,14)], by = 'rpt_rec_num', all.y = TRUE)

    #add current year
    currentYear$year = j

    #rbind HcrisPanel with the current year
    HcrisPanel = rbind(HcrisPanel, currentYear)
  }

  #write out the final csv file
  write.csv(HcrisPanel, "hcris_raw/HcrisPanel.csv", row.names = FALSE)
}

#read in merged csv file
HcrisPanel = read.csv("hcris_raw/HcrisPanel.csv")
print("data frames merged!")
```

```
## [1] "data frames merged!"
```

```
library(stringr)
library(tidyverse)
```

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

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   2.0.1
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
raw_data = HcrisPanel

# 1. Drop NA prvdr_num rows
d = raw_data %>% drop_na(prvdr_num)

# 2. Account for negative epo variable records
d$epo_cost = abs(d$epo_cost)
d$epo_net_cost = abs(d$epo_net_cost)
d$epo_rebates = abs(d$epo_rebates)

# 3. Replace missing values with 0 for epo_rebates
d$epo_rebates = replace(d$epo_rebates, is.na(d$epo_rebates), 0)

# 4. Check missing epo_cost and epo_net_cost observations
d$epo_cost = ifelse((is.na(d$epo_cost) & !is.na(d$epo_net_cost) & d$epo_rebates == 0), d$epo_net_cost, c

d$epo_cost = ifelse((is.na(d$epo_cost) & !is.na(d$epo_net_cost) & d$epo_rebates != 0), d$epo_net_cost +

blank_idxes = rep(NA, 1)

for (x in 1:length(d$epo_cost)) {
  if (is.na(d$epo_cost[[x]]) & is.na(d$epo_net_cost[[x]])) {
    if (d$epo_rebates[[x]] == 0) {
      d$epo_cost[[x]] = 0
      d$epo_net_cost[[x]] = 0
    } else {
      blank_idxes = c(blank_idxes, x)
    }
  }
}

d$epo_net_cost = ifelse(!is.na(d$epo_cost) & is.na(d$epo_net_cost), d$epo_cost - d$epo_rebates, d$epo_n

# 5. Switch epo_cost and epo_net_cost
for (x in 1:length(d$epo_cost)) {
  if (!is.na(d$epo_cost[[x]]) & !is.na(d$epo_net_cost[[x]]) & d$epo_cost[[x]] < d$epo_net_cost[[x]]) {
    tmp = d$epo_cost[[x]]
    d$epo_cost[[x]] = d$epo_net_cost[[x]]
    d$epo_net_cost[[x]] = tmp
  }
```

```
}

# 6. Change prvdr_num from "322664" to "342664"
d$prvdr_num = ifelse(d$prvdr_num == 322664, 342664, d$prvdr_num)

# 7. Convert fy_bgn_dt, fy_end_dt, report_start_date, and report_end_date to date format
d$fy_bgn_dt = mdy(d$fy_bgn_dt)
d$fy_end_dt = mdy(d$fy_end_dt)
d$report_start_date = mdy(d$report_start_date)
d$report_end_date = mdy(d$report_end_date)

# 8. Use fy_bgn_dt and fy_end_dt for missing report start and end dates
for (x in 1:length(d$report_start_date)) {
  if (is.na(d$report_start_date[[x]])) {
    d$report_start_date[[x]] = d$fy_bgn_dt[[x]];
  }

  if (is.na(d$report_end_date[[x]])) {
    d$report_end_date[[x]] = d$fy_end_dt[[x]];
  }
}

# 9. Reformat zip codes
for (x in 1:length(d$zip_code)) {
trimmed = trimws(d$zip_code[[x]], whitespace = "[\t\r\n]")
d$zip_code[[x]] = as.numeric(substr(trimmed, 1, 5))
}
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
# 9.5 (Failed) attempt to clean zip codes further
# getMode = function(v) {
# uniqv = unique(v)
# uniqv[which.max(tabulate(match(v, uniqv)))]
# }
#
# providers = unique(d$prvdr_num, incomparables = c(NA))
#
# for (p in providers) {
# prvdr_zips = subset(d, d$prvdr_num == p)$zip_code
# prvdr_zips = prvdr_zips[!is.na(prvdr_zips)]
#
# if (length(prvdr_zips) > 0) {
# curr_zip = getMode(prvdr_zips)
# print(curr_zip)
#
# d$zip_code = ifelse(d$prvdr_num == p, curr_zip, d$zip_code)
# }
# }

# 10. Map zip_codes to states
states <- read.csv("zipcode_state.csv")


states = states[,c(1,4)]

for (zip in 1:length(d$zip_code)) {
d$state[zip] = ifelse(is.na(d$state[zip]) | d$state[zip] == '', states[match(d$zip_code[zip], states$Zi
}

# 11. Reformat Provider Chain Identities
d$chain_identity = sub("^DA.{3,5}A.*$", "DAVITA", d$chain_identity)
d$chain_identity = sub("^(FR|FE).{3,9}?S.*$", "FRESENIUS", d$chain_identity)
d$chain_id = ifelse(d$chain_indicator == 'N', 0, ifelse(d$chain_identity == 'DAVITA', 2, ifelse(d$chain
```

```r
# 12. Remake chain_indicator based on chain_identity
d$chain_indicator = ifelse(d$chain_identity == 'DAVITA' | d$chain_identity == 'FRESENIUS', 'Y', d$chain_
```

Question 3 Analysis

Our goal of this analysis was to compare clinics operated by chains with independent clinics to find any discrepancies in costs and behaviors. To accomplish this, we investigated three questions using the data cleaned above.

Research Question 1: EPO Net Cost, Cost, and Rebates

```r
attach(d)

print("Linear Regression on Indepdents")
```

```
## [1] "Linear Regression on Indepdents"
```

```r
avgEpoNetIndependent = glm(epo_net_cost ~ non_medicare_sessions + non_medicare_sessions_indirect +
                                avg_days_open_per_week + avg_session_time + num_machines_regular +
                                dialyser_reuse_times + epo_total + epo_cost + epo_rebates +
                                total_treatments_hd + total_costs_hd_drugs + total_costs_hd_housekeepin
                                total_costs_hd_machines + total_costs_hd_salaries + total_costs_hd_bene
                data = d, subset = which(d$chain_id == 0))
summary(avgEpoNetIndependent)
```

```
##
## Call:
## glm(formula = epo_net_cost ~ non_medicare_sessions + non_medicare_sessions_indirect +
##     avg_days_open_per_week + avg_session_time + num_machines_regular +
##     dialyser_reuse_times + epo_total + epo_cost + epo_rebates +
##     total_treatments_hd + total_costs_hd_drugs + total_costs_hd_housekeeping +
##     total_costs_hd_machines + total_costs_hd_salaries + total_costs_hd_benefits,
##     data = d, subset = which(d$chain_id == 0))
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -5.821e-10  -2.037e-10   0.000e+00   1.701e-10   1.863e-09
##
## Coefficients:
##                                 Estimate Std. Error    t value Pr(>|t|)
## (Intercept)                    2.941e-10  1.595e-10  1.844e+00 0.066357 .
## non_medicare_sessions         -1.216e-13  1.424e-14 -8.535e+00 1.29e-15 ***
## non_medicare_sessions_indirect  3.753e-14  1.443e-14  2.600e+00 0.009861 **
## avg_days_open_per_week          2.501e-11  1.895e-11  1.320e+00 0.188157
## avg_session_time               -5.241e-11  2.784e-11 -1.883e+00 0.060910 .
## num_machines_regular            1.710e-11  5.067e-12  3.375e+00 0.000854 ***
## dialyser_reuse_times           -3.825e-15  5.984e-14 -6.400e-02 0.949088
## epo_total                      -1.086e-17  5.962e-18 -1.821e+00 0.069826 .
## epo_cost                        1.000e+00  1.157e-16  8.645e+15  < 2e-16 ***
```

```
## epo_rebates                    -1.000e+00  8.700e-16 -1.149e+15  < 2e-16 ***
## total_treatments_hd            -2.521e-14  1.154e-14 -2.185e+00 0.029768 *
## total_costs_hd_drugs           -6.502e-16  3.707e-16 -1.754e+00 0.080642 .
## total_costs_hd_housekeeping     2.715e-16  2.359e-16  1.151e+00 0.250738
## total_costs_hd_machines         3.052e-16  3.633e-16  8.400e-01 0.401666
## total_costs_hd_salaries        -3.556e-17  1.129e-16 -3.150e-01 0.753061
## total_costs_hd_benefits         8.141e-16  3.394e-16  2.399e+00 0.017165 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.10238e-19)
##
##     Null deviance: 4.5636e+13  on 269  degrees of freedom
## Residual deviance: 2.8000e-17  on 254  degrees of freedom
##   (5660 observations deleted due to missingness)
## AIC: -11002
##
## Number of Fisher Scoring iterations: 1
```

```
print("Linear Regression on Chains")
```

```
## [1] "Linear Regression on Chains"
```

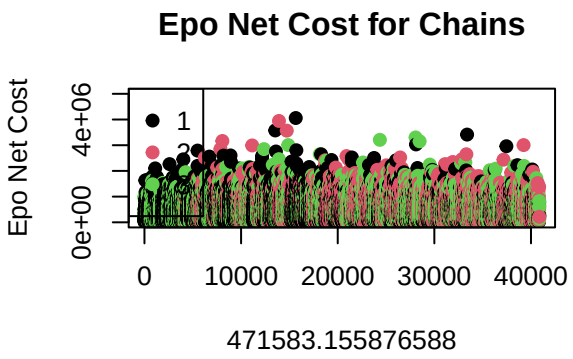
```
avgEpoNetChain = glm(epo_net_cost ~ non_medicare_sessions + non_medicare_sessions_indirect +
                               avg_days_open_per_week + avg_session_time + num_machines_regular +
                               dialyser_reuse_times + epo_total + epo_cost + epo_rebates +
                               total_treatments_hd + total_costs_hd_drugs + total_costs_hd_housekeepin
                               total_costs_hd_machines + total_costs_hd_salaries + total_costs_hd_bene
                 data = d, subset = which(d$chain_id != 0))
summary(avgEpoNetChain)
```

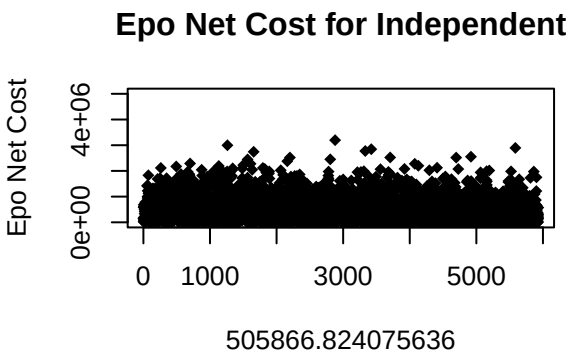
```
##
## Call:
## glm(formula = epo_net_cost ~ non_medicare_sessions + non_medicare_sessions_indirect +
##     avg_days_open_per_week + avg_session_time + num_machines_regular +
##     dialyser_reuse_times + epo_total + epo_cost + epo_rebates +
##     total_treatments_hd + total_costs_hd_drugs + total_costs_hd_housekeeping +
##     total_costs_hd_machines + total_costs_hd_salaries + total_costs_hd_benefits,
##     data = d, subset = which(d$chain_id != 0))
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -3.493e-09  -5.821e-10  -2.328e-10   0.000e+00   1.397e-09
##
## Coefficients:
##                                  Estimate Std. Error    t value Pr(>|t|)
## (Intercept)                     1.579e-10  1.553e-10  1.017e+00  0.30965
## non_medicare_sessions          -4.684e-14  1.067e-14 -4.389e+00 1.27e-05 ***
## non_medicare_sessions_indirect  4.039e-14  1.168e-14  3.458e+00  0.00057 ***
## avg_days_open_per_week         -1.223e-10  2.521e-11 -4.852e+00 1.44e-06 ***
## avg_session_time               -7.641e-14  1.494e-13 -5.110e-01  0.60915
## num_machines_regular            7.251e-11  3.962e-12  1.830e+01  < 2e-16 ***
## dialyser_reuse_times           -1.431e-12  1.737e-12 -8.240e-01  0.41011
```

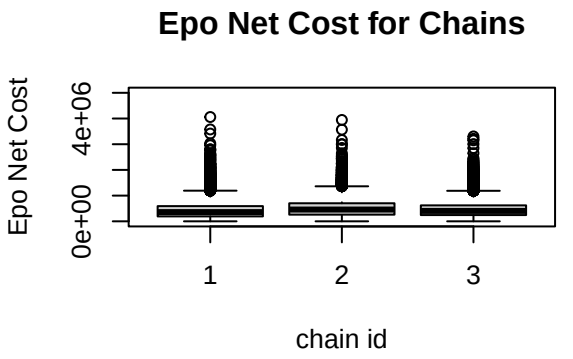
```
## epo_total                      -1.120e-18  5.834e-18 -1.920e-01  0.84774
## epo_cost                        1.000e+00  1.367e-16  7.317e+15  < 2e-16 ***
## epo_rebates                    -1.000e+00  4.319e-16 -2.316e+15  < 2e-16 ***
## total_treatments_hd             6.051e-15  9.057e-15  6.680e-01  0.50426
## total_costs_hd_drugs            1.699e-15  6.813e-16  2.494e+00  0.01282 *
## total_costs_hd_housekeeping    -5.748e-16  2.927e-16 -1.963e+00  0.04990 *
## total_costs_hd_machines        -1.759e-16  4.805e-16 -3.660e-01  0.71438
## total_costs_hd_salaries        -5.811e-17  1.990e-16 -2.920e-01  0.77039
## total_costs_hd_benefits         6.133e-16  6.708e-16  9.140e-01  0.36077
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 3.661665e-19)
##
##     Null deviance: 1.1037e+14  on 908  degrees of freedom
## Residual deviance: 3.2699e-16  on 893  degrees of freedom
##   (39949 observations deleted due to missingness)
## AIC: -35991
##
## Number of Fisher Scoring iterations: 1
```

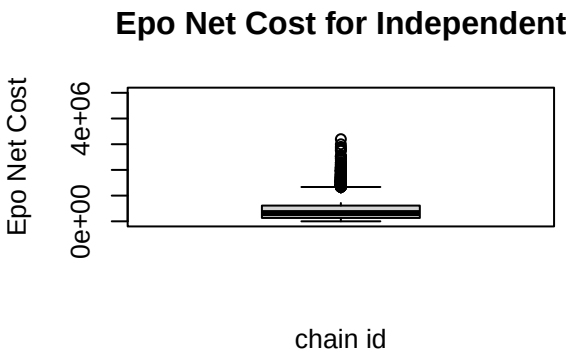
```
chains = subset(d, d$chain_id != 0)
independent = subset(d, d$chain_id == 0)

par(mfrow=c(2,2))
plot(chains$epo_net_cost, ylim = c(0,5000000), main ="Epo Net Cost for Chains", col = factor(chains$cha
legend("topleft", legend = levels(factor(chains$chain_id)), col = factor(levels(factor(chains$chain_id))
plot(independent$epo_net_cost, ylim = c(0,5000000), main ="Epo Net Cost for Independent", col = factor(
boxplot(chains$epo_net_cost~ chains$chain_id, data = chains, ylab = "Epo Net Cost", xlab="chain id", ma
boxplot(independent$epo_net_cost ~ independent$chain_id, data = chains, ylab = "Epo Net Cost", xlab="cha
```

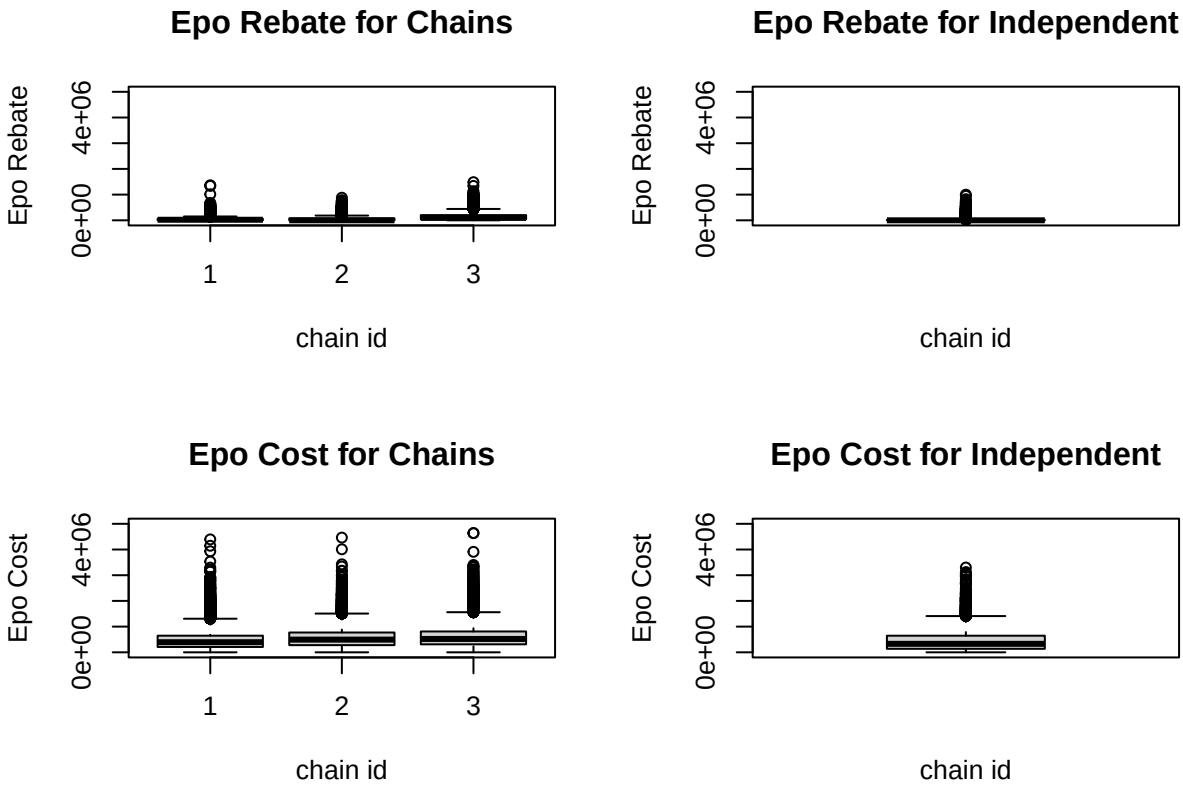









```
par(mfrow=c(2,2))
boxplot(chains$epo_rebates ~ chains$chain_id, data = chains, ylab = "Epo Rebate", xlab="chain id", main
boxplot(independent$epo_rebates ~ independent$chain_id, data = chains, ylab = "Epo Rebate", xlab="chain
boxplot(chains$epo_cost~ chains$chain_id, data = chains, ylab = "Epo Cost", xlab="chain id", main ="Epo
boxplot(independent$epo_cost ~ independent$chain_id, data = chains, ylab = "Epo Cost", xlab="chain id",
```



We first set out to run separate linear regressions based on chain and independent firms and regress them for epo_net_cost against all numeric features. We found that EPO cost and EPO rebates are strong significant predictors of EPO net cost based on our regression. This should come as no surprise, as EPO total cost - rebate should give us net cost. From this, we created box plots that visualized the EPO rebates and EPO costs for the different chain categories and the independent clinics. This showed that, on average, EPO rebates and costs were higher for chain clinics than independent ones. However, we also found that, on average, the net costs for chains were lower than independent counterparts by about $34,000. This aligns with the research paper's specific studies on EPO, which found that dosage more than doubled post-acquisition. Since Medicare reimbursement for drugs like EPO are determined by quantity administered, firms would be incentivized to increase dosage in their patients, despite medical evidence that excessive EPO increases risk for cardiovascular events. Although we cannot conclusively attribute a causal effect to our findings, our regression does match the research paper's study. Given more granular patient and treatment data, we could better analyze this relationship.

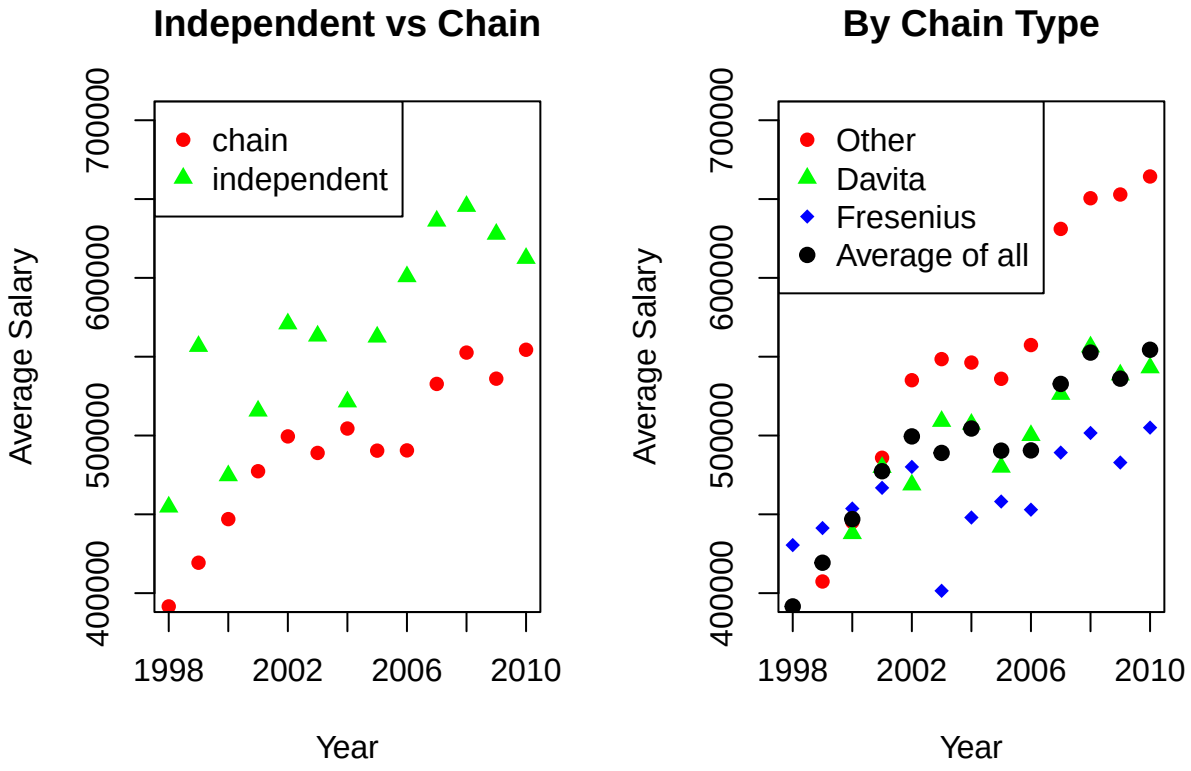
Research Question 2: Salaries

```
averageOfEachYearIndependent = rep(0, 12)
averageOfEachYearChain = rep(0, 12)
averageOfEachOher = rep(0, 12)
averageOfEachDavita = rep(0, 12)
averageOfEachFresenius = rep(0, 12)
count = 1
for(i in 1998:2010) {
  yearSubsetIndependent = subset(independent, year == i)
  yearSubsetChain = subset(chains, year == i)
```

```
  yearSubsetOther = subset(chains, year == i & chain_id == 1)
  yearSubsetDavita = subset(chains, year == i & chain_id == 2)
  yearSubsetFresenius = subset(chains, year == i & chain_id == 3)
  averageOfEachYearIndependent[count] = mean(na.omit(yearSubsetIndependent$total_costs_hd_salaries))
  averageOfEachYearChain[count] = mean(na.omit(yearSubsetChain$total_costs_hd_salaries))
  averageOfEachOher[count] = mean(na.omit(yearSubsetOther$total_costs_hd_salaries))
  averageOfEachDavita[count] = mean(na.omit(yearSubsetDavita$total_costs_hd_salaries))
  averageOfEachFresenius[count] = mean(na.omit(yearSubsetFresenius$total_costs_hd_salaries))
  count = count + 1
}

par(mfrow=c(1,2))
plot(1998:2010, averageOfEachYearChain, pch = 16, col = "red", ylim = c(400000, 700000), ylab ="Average
par(new = TRUE)
plot(1998:2010, averageOfEachYearIndependent, pch = 17, col = "green",axes = FALSE, xlab = "", ylab = "
legend("topleft",legend = c("chain", "independent"), col=c("red", "green"), pch=c(16,17))

plot(1998:2010, averageOfEachOher, pch = 16, col = "red", ylim = c(400000, 700000), ylab ="Average Sala
par(new = TRUE)
plot(1998:2010, averageOfEachDavita, pch = 17, col = "green", axes = FALSE, xlab = "", ylab = "", ylim =
par(new = TRUE)
plot(1998:2010, averageOfEachFresenius, pch = 18, col = "blue", axes = FALSE, xlab = "", ylab = "", yli
par(new = TRUE)
plot(1998:2010, averageOfEachYearChain, pch = 19, col = "black", axes = FALSE, xlab = "", ylab = "", yl
legend("topleft",legend = c("Other", "Davita", "Fresenius", "Average of all"), col=c("red", "green", "bl
```



We found that there was a consistent difference between the average salaries of independent clinics compared to chain clinics from 1998 to 2010. Our first graph illustrates this difference, with independent clinics paying higher salaries on average than chain clinics on average each year, even with increases in salaries across the board. Our second graph shows how the chain clinics differ in their average salaries. The gap is further emphasized, with Davita and Fresenius consistently paying at or below the average calculated salary compared to other chains. Overall, this aligns with the findings reported in the research paper and podcast, as researchers concluded that chain clinics were cutting costs by moving towards more low-skill technicians and less highly-trained nurses. Typically, nurses are more qualified and provide better care for patients than technicians, which translates into higher salaries. From aggregate salary-collecting websites like Glassdoor (founded 2007) and Ziprecruiter (founded 2010), dialysis nurse salaries in the US have been around $60-75k over the past decade, while dialysis technician salaries have ranged from $40-50k. These external statistics, in tandem with our findings, corroborate the research paper's own conclusions.

Research Question 3: Non-Medicare Sessions

```
print("Linear Regression on Non Medicare Sessions for Chains")
```

```
## [1] "Linear Regression on Non Medicare Sessions for Chains"
```

```
summary(glm(non_medicare_sessions ~ total_costs_hd_drugs + total_treatments_hd +
            total_costs_hd_labs + dialyser_reuse_times + total_costs_hd_benefits
            + total_costs_hd_housekeeping + total_costs_hd_machines
            + total_costs_hd_other + total_costs_hd_salaries +
              total_costs_hd_supplies, data = chains))
```

```
##
## Call:
## glm(formula = non_medicare_sessions ~ total_costs_hd_drugs +
##     total_treatments_hd + total_costs_hd_labs + dialyser_reuse_times +
##     total_costs_hd_benefits + total_costs_hd_housekeeping + total_costs_hd_machines +
##     total_costs_hd_other + total_costs_hd_salaries + total_costs_hd_supplies,
##     data = chains)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -13157    -819    -196     458   88079
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -2.270e+02  3.658e+01  -6.207 5.56e-10 ***
## total_costs_hd_drugs        2.106e-03  7.482e-04   2.815  0.00489 **
## total_treatments_hd         1.103e-02  1.868e-03   5.904 3.62e-09 ***
## total_costs_hd_labs         1.694e-02  2.284e-03   7.420 1.24e-13 ***
## dialyser_reuse_times       -4.199e-06  2.209e-06  -1.901  0.05734 .
## total_costs_hd_benefits     1.863e-04  4.096e-04   0.455  0.64921
## total_costs_hd_housekeeping 1.953e-04  1.751e-04   1.116  0.26462
## total_costs_hd_machines    -4.462e-03  4.218e-04 -10.579  < 2e-16 ***
## total_costs_hd_other        1.235e-03  1.102e-04  11.210  < 2e-16 ***
## total_costs_hd_salaries     5.265e-03  1.421e-04  37.046  < 2e-16 ***
## total_costs_hd_supplies    -2.549e-04  3.047e-04  -0.836  0.40292
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 4246822)
##
##     Null deviance: 1.3680e+11  on 15029  degrees of freedom
## Residual deviance: 6.3783e+10  on 15019  degrees of freedom
##   (25828 observations deleted due to missingness)
## AIC: 272049
##
## Number of Fisher Scoring iterations: 2
```

```
print("Linear Regression on Non Medicare Sessions for Independents")
```

```
## [1] "Linear Regression on Non Medicare Sessions for Independents"
```

```
summary(glm(non_medicare_sessions ~ total_costs_hd_drugs + total_treatments_hd
            + total_costs_hd_labs + dialyser_reuse_times + total_costs_hd_benefits
            + total_costs_hd_housekeeping + total_costs_hd_machines +
              total_costs_hd_other + total_costs_hd_salaries +
              total_costs_hd_supplies, data = independent))
```

```
##
## Call:
## glm(formula = non_medicare_sessions ~ total_costs_hd_drugs +
##     total_treatments_hd + total_costs_hd_labs + dialyser_reuse_times +
##     total_costs_hd_benefits + total_costs_hd_housekeeping + total_costs_hd_machines +
##     total_costs_hd_other + total_costs_hd_salaries + total_costs_hd_supplies,
##     data = independent)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -10545.9    -760.5    -111.7     551.9   21055.7
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -3.922e+02  8.788e+01  -4.463 8.59e-06 ***
## total_costs_hd_drugs        -1.871e-03  5.519e-04  -3.391 0.000712 ***
## total_treatments_hd          3.839e-01  1.871e-02  20.518  < 2e-16 ***
## total_costs_hd_labs          4.053e-02  5.202e-03   7.792 1.12e-14 ***
## dialyser_reuse_times         6.125e-06  2.950e-06   2.076 0.038043 *
## total_costs_hd_benefits      3.895e-03  6.319e-04   6.164 8.76e-10 ***
## total_costs_hd_housekeeping  1.067e-03  3.636e-04   2.933 0.003396 **
## total_costs_hd_machines     -1.314e-03  7.586e-04  -1.733 0.083311 .
## total_costs_hd_other        -5.327e-04  2.516e-04  -2.117 0.034361 *
## total_costs_hd_salaries     -1.112e-03  2.416e-04  -4.604 4.44e-06 ***
## total_costs_hd_supplies     -3.060e-03  5.535e-04  -5.527 3.74e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 3279694)
##
##     Null deviance: 1.5775e+10  on 1766  degrees of freedom
## Residual deviance: 5.7591e+09  on 1756  degrees of freedom
```

```
##   (4163 observations deleted due to missingness)
## AIC: 31538
##
## Number of Fisher Scoring iterations: 2
```

We used multiple costs variables to look at changes in the number of sessions done through private (non-Medicare) insurance. From our regressions, we can see that the total cost for hemodialysis drugs has a positive relationship with nonmedicare session for chain clinics (coefficient = 2.106e-03), while this relationship became negative for independent clinics (coefficient = -1.871e-03). The same trend can be found for the total 'other' and salary costs for hemodialysis, changing from 1.235e-03 to -5.327e-04 and 5.265e-03 to -1.112e-03, respectively between chain and independent clinics. In general, large dialysis chains have more market and bargaining power, and are thus able to negotiate higher reimbursement rates from private insurers. Chain firms would then be incentivized move towards non-medicare sessions as their costs increase, in order to receive more reimbursements. In contrast, independent chains are forced to revert to Medicare's flat rates, having less bargaining power. The research paper and podcast both touch on the role that Medicare plays in the dialysis industry, and our findings seem to align with these general economic trends.