Code ▾

FINAL PROJECT: MATCHING DONORS TO PROJECTS

Kaggle Dataset

Data Science For Good: DonorsChoose.org (https://www.kaggle.com/donorschoose/io)

Founded in 2000 by a Bronx history teacher, DonorsChoose.org has raised $685 million for America's classrooms. Teachers at three-quarters of all the public schools in the U.S. have come to DonorsChoose.org to request what their students need, making DonorsChoose.org the leading platform for supporting public education.

To date, 3 million people and partners have funded 1.1 million DonorsChoose.org projects. But teachers still spend more than a billion dollars of their own money on classroom materials. To get students what they need to learn, the team at DonorsChoose.org needs to be able to connect donors with the projects that most inspire them.

In the second Kaggle Data Science for Good challenge, DonorsChoose.org, in partnership with Google.org, is inviting the community to help them pair up donors to the classroom requests that will most motivate them to make an additional gift. To support this challenge, DonorsChoose.org has supplied anonymized data on donor giving from the past five years. The winning methods will be implemented in DonorsChoose.org email marketing campaigns.

Problem Statement

DonorsChoose.org has funded over 1.1 million classroom requests through the support of 3 million donors, the majority of whom were making their first-ever donation to a public school. If DonorsChoose.org can motivate even a fraction of those donors to make another donation, that could have a huge impact on the number of classroom requests fulfilled.

A good solution will enable DonorsChoose.org to build targeted email campaigns recommending specific classroom requests to prior donors. Part of the challenge is to assess the needs of the organization, uncover insights from the data available, and build the right solution for this problem. Submissions will be evaluated on the following criteria:

- *Performance - How well does the solution match donors to project requests to which they would be motivated to donate? DonorsChoose.org will not be able to live test every submission, so a strong entry will clearly articulate why it will be effective at motivating repeat donations.*
- *Adaptable - The DonorsChoose.org team wants to put the winning submissions to work, quickly. Therefore a good entry will be easy to implement in production.*

- *Intelligible - A good entry should be easily understood by the DonorsChoose.org team should it need to be updated in the future to accommodate a changing marketplace.*

Data Preprocessing

Hide

```
options(scipen = 999)  # Disable scientific notation
```

Hide

```
donations
donors
projects
resources
schools
teachers
```

Hide

```
# Unique column names across all datasets
columns <- NULL
for (table in list(donations, donors, projects, resources, schools, teachers)) {
  columns = c(columns, colnames(table))
}
sort(unique(columns))
```

Hide

```
# Ensure there is only one subject category per row of Projects.csv for further
# analysis.
# Take a random sample of Projects.csv with only Project ID and Project Subject
# Category Tree.
# projects.categorized is a data.table with Project ID and a separate row for
# every unique subject category for that Project ID.
set.seed(0)
n <- 50000

projects.categorized <- projects[sample(nrow(projects), n), list(`Project ID`, `Proje
ct Subject Category Tree`)]
for (row in 1:nrow(projects.categorized)) {
  row.categories <- unlist(strsplit(projects.categorized[row]$`Project Subject Catego
ry Tree`, ', '))
  if (length(row.categories) > 1) {
    duplicate <- projects.categorized[row]
    projects.categorized[row]$`Project Subject Category Tree` <- row.categories[1]
    for (category in row.categories[-1]) {
      duplicate$`Project Subject Category Tree` <- category
      projects.categorized <- rbindlist(list(projects.categorized, duplicate))
    }
  }
}

# Readd "Warmth, Care & Hunger"
projects.categorized.warmth <- projects.categorized[`Project Subject Category Tree` =
= 'Warmth']
projects.categorized.warmth$`Project Subject Category Tree` <- 'Warmth, Care & Hunger
'
projects.categorized <- projects.categorized[!(`Project Subject Category Tree` %in% c
('Warmth', 'Care & Hunger'))]
projects.categorized <- rbindlist(list(projects.categorized, projects.categorized.war
mth))
projects.categorized
```

Hide

```
# 8 unique project subject categories
categories.grouped <- projects.categorized[, .N, by=`Project Subject Category Tree`][
order(N, decreasing=TRUE)]
categories.grouped

categories <- categories.grouped$`Project Subject Category Tree`
categories
```

Scenario: Text Mining

Hide

```
texts <- merge(projects.categorized, projects[, c(1,8,9)], by='Project ID')[, -1]

for (category in categories) {
  category.projects <- texts[`Project Subject Category Tree`==category]
  words <- c(category.projects$`Project Short Description`, category.projects$`Projec
t Need Statement`)
  wordcloud(words, max.words=50)
}
```

Scenario: Seasonal pattern

Question: Is there a seasonal pattern of donations? Is the total amount of received donation related to donation date? Is the total amount of received donation related to day of the week?

Hide

```
# open the project merged
merged_projet <- read.csv("merged_projects.csv")
merged_projet['month']<- NA
month <- as.Date(merged_projet$Project.Posted.Date)
merged_projet$month <- as.numeric(strftime(month,"%m"))
ggplot(merged_projet, aes(x=month)) + geom_histogram(aes(y=..count..), stat = 'bin',b
inwidth=1,color="darkblue", fill="lightblue")+
  labs(title="Project posted in each month",x="month", y = "count")+
  theme_classic()
fit <- lm(merged_projet$Donation.Amount~merged_projet$duration)
merged_projet$Project.Fully.Funded.Date<- merged_projet[which(merged_projet$Project.F
ully.Funded.Date!=''),]
merged_projet['year']<- NA
merged_projet$year <- as.numeric(strftime(as.Date(merged_projet$Project.Posted.Date),
"%y"))
merged_recent_year <- merged_projet[which(merged_projet$year!=18),]
ggplot(merged_recent_year, aes(x=Project.Fully.Funded.Date,color=year,fill=year)) +
  geom_histogram(binwidth = 10,alpha = 0.5, aes(y=..count..), stat = 'bin', position
= 'stack',alpha=0.5)+ xlim(0, 90)+
  geom_vline(data=mu1, aes(xintercept=grp.mean, color=Donor.Is.Teacher),
             linetype="dashed")+
  scale_color_manual(values=c("#999999", "#E69F00", "#56B4E9"))+
  scale_fill_manual(values=c("#999999", "#E69F00", "#56B4E9"))+
  labs(title="Donation amount for Teacher vs Nonteacher",x="Donation amount", y = "co
unt")+
  theme_classic()
```

```
ggplot(merged_recent_year,aes(x=month))+geom_histogram(aes(y=..count..), stat = 'bin'
,binwidth=1,color="darkblue", fill="lightblue")+facet_grid(~year)+
  labs(title="Amount of project posted from 2013-2017",x="month for each year", y = "
count")+theme_bw()
summary_donation_amount <- c(mean(df_merged$Donation.Amount),median(df_merged$Donatio
n.Amount),quantile(df_merged$Donation.Amount, c(.25, .75)))
#Chi squared godness fit test
merged_2014 <- merged_projet[which(merged_projet$year==14),]
set.seed(2017)
n<- nrow(merged_2014)
uniform.sample<-runif(n,min=0,max=365)
merged_2014['exact_day']<- NA
merged_2014$exact_day <- as.numeric(strftime(as.Date(merged_2014$Project.Fully.Funded
.Date), format = "%j"))
locations<- merged_2014$exact_day
for (num.regions in c(3,4,6,12,24)) {
  expected.counts <- rep(n/num.regions, num.regions)
  observed.counts <- as.vector(table(cut(locations, breaks=seq(0,365,length.out=num.r
egions+1), include.lowest=TRUE)))
  hist(locations, breaks=num.regions, probability=TRUE, col=rgb(1,0,0,0.5), main=past
e('Locations of project posted (Original vs. Simulated) in',num.regions,'Sub-Interval
s'), xlab='Base Pair')
  hist(uniform.sample, breaks=num.regions, probability=TRUE, col=rgb(0,0,1,0.5), add=
TRUE)
  lines(density(locations, adjust=2), col=2)
  lines(density(uniform.sample, adjust=2), col=4)
  legend('topright', legend=c('Original', 'Uniform'), lty=c(1,1), col=c(rgb(1,0,0,0.5
), rgb(0,0,1,0.5)))
  print(num.regions)
  print(chisq.test(observed.counts, p=expected.counts/n))

  residuals <- (observed.counts - expected.counts) / sqrt(expected.counts)
  plot(residuals, type='h', main=paste('Standardized Residuals of Number of project p
osted for',num.regions,'Sub-Intervals'), xlab='Sub-Interval Index', ylab='Standardize
d Residuals')
}
project_teacher <- merge(Teachers,merged_projet,by="Teacher.ID")
project_teacher <- merge(project_teacher,df_merged, by="Project.ID")
project_teacher_male <- project_teacher[which(project_teacher$Teacher.Prefix == "Mr."
),]
project_teacher_female <- project_teacher[which(project_teacher$Teacher.Prefix == "Mr
s." | project_teacher$Teacher.Prefix=="Ms."),]
res <- wilcox.test(project_teacher_male$Donation.Amount, project_teacher_female$Donat
ion.Amount)
res
#seperate the merged project data into 2014 and 2015
merged_2014 <- merged_2014[which(merged_2014$Project.Current.Status=="Fully Funded"),
]
merged_2015 <- merged_recent_year[which(merged_recent_year$year==15),]
merged_2016 <- merged_recent_year[which(merged_recent_year$year==16),]
merged_2015['exact_day']<- NA
merged_2015$exact_day <- as.numeric(strftime(as.Date(merged_2015$Project.Fully.Funded
.Date), format = "%j"))
merged_2015 <- merged_2015[which(merged_2015$Project.Current.Status=="Fully Funded"),
]
merged_2014_DOY <- merged_2014[order(merged_2014$exact_day),]
merged_2015_DOY <- merged_2015[order(merged_2015$exact_day),]
merged_2014_sum <- ddply(merged_2014,c("exact_day"),summarize,B=sum(donations))
merged_2015_sum <- ddply(merged_2015,c("exact_day"),summarize,B=sum(donations))
difference_2014 <- apply(merged_2014_sum,2,diff)
difference_2015 <- apply(merged_2015_sum,2,diff)
# ks test for the difference of two years
ks.test(difference_2014, difference_2015)
res <- wilcox.test(merged_2014$donations, merged_2015$donations)
res
a <- c(mean(merged_2014$donations),mean(merged_2015$donations))
res <- wilcox.test(c(372,372,372,372,372,372), c(390,390))
res
merged_2014_sum_month <- ddply(merged_2014,c("month"),summarize,B=sum(donations))
merged_2015_sum_month <- ddply(merged_2015,c("month"),summarize,B=sum(donations))
difference_2014_m <- apply(merged_2014_sum_month,2,diff)
difference_2015_m <- apply(merged_2015_sum_month,2,diff)
res <- wilcox.test(difference_2014_m, difference_2015_m)
res

# Create the quantile-quantile data table
qqplot(x=merged_2014_sum_month$B,y=merged_2015_sum$B,asp=1,xlab="donation amount for
2014",ylab="donation amount for 2015",main="qqplot for donation amount between 2014 a
nd 2015")
qq.out <- as.data.frame(qq.out)
```

Scenario: Teacher identity

Question: Test whether the teachers' donation pattern is different from non-teachers' donation pattern (Test whether the fraction of teacher in the whole population is larger than the proportion of the donors that are non teacher).

Hide

```
teacher <- df_merged[which(df_merged$Donor.Is.Teacher== 'Yes'),]
nonteacher <- df_merged[which(df_merged$Donor.Is.Teacher=='No'),]
teacher_money_amount <- teacher$Donation.Amount
nonteacher_money_amount <- nonteacher$Donation.Amount
```

Hide

```
#2.1 Wilcoxon test- test for the difference in mean
# one-sided test- test for difference greater than or equal to 0
# null: true shift is greater than 0 teacher donates more than non-teachers
wilcox.test(teacher_money_amount, nonteacher_money_amount, paired = FALSE, correct =
FALSE)
wilcox.test(teacher_money_amount, nonteacher_money_amount, paired = FALSE, correct =
FALSE, alternative = "less")
```

Hide

```
#2.2 True fraction of teacher donors in the donor population
teacher.donor.fraction <- round(as.numeric(table(Donors[,`Donor Is Teacher`])[2])/nro
w(Donors)*100, 3)
print(paste0("The true fraction of teacher donors is ", teacher.donor.fraction, "%"))

# EDA: Check if teacher donors' donation pattern (except for the ones who only donate
1 dollar) follow normal distribution
t <- sample(teacher$Donation.Amount, 10000)
tt <- teacher[which(teacher$Donation.Amount<500 & teacher$Donation.Amount>1),6]$Donat
ion.Amount
qqplot(log(tt), rnorm(length(tt), mean(log(tt)), var(log(tt))))
abline(0, 1)
hist(log(tt))
```

Hide

```
# Set the x and y limits
xylim <- range( c(qq.out$x, qq.out$y) )

# Generate the QQ plot
ggplot(qq.out, aes( x= x, y = y)) +
  geom_point() +
  geom_abline( intercept=0, slope=1) +
  coord_fixed(ratio = 1, xlim=xylim, ylim = xylim) +
  xlab("donation amount in 2014") + ylab("donation amount in 2015")
#seperate df_merged into first donor and return donor
duplicate_Donor <- df_merged[duplicated(df_merged$Donor.ID),]
unique_Donor <- df_merged[!duplicated(df_merged$Donor.ID),]
#plot barplot for teacher and nonteacher for return donor and first donor
ggplot(duplicate_Donor, aes(x=factor(Donor.Is.Teacher)))+
  geom_bar(stat="count", width=0.7, fill="steelblue")+
  theme_minimal()+ labs(title="Teacher vs Nonteacher for returned Donor",x="Teacher v
s Nonteacher", y = "count")
ggplot(unique_Donor, aes(x=factor(Donor.Is.Teacher)))+
  geom_bar(stat="count", width=0.7, fill="steelblue")+
  theme_minimal()+ labs(title="Teacher vs Nonteacher for unique Donor",x="Teacher vs
Nonteacher", y = "count")

# donate less than 1 dollar
one_dollar_Donor <- df_merged[which(df_merged$Donation.Amount<=1),]
more_than_one_Donor <- df_merged[which(df_merged$Donation.Amount>1),]
ggplot(one_dollar_Donor, aes(x=factor(Donor.Is.Teacher)))+
  geom_bar(stat="count", width=0.7, fill="steelblue")+
  theme_minimal()+ labs(title="Donors that donate less than or equal to one dollar",x
="Teacher vs Nonteacher", y = "count")

#plot barplot for teacher and nonteacher for return donor and first donor
ggplot(more_than_one_Donor, aes(x=factor(Donor.Is.Teacher)))+
  geom_bar(stat="count", width=0.7, fill="steelblue")+
  theme_minimal()+ labs(title="Donors that donate more than one dollar",x="Teacher vs
Nonteacher", y = "count")


#ks.test
teacher_final <- df_merged[which(df_merged$Donor.Is.Teacher=='Yes'),]
nonteacher_final<-df_merged[which(df_merged$Donor.Is.Teacher=='No'),]
ks.test(teacher_final$Donation.Amount,nonteacher_final$Donation.Amount)
```

Scenario: Donor to population ratio

Question: Which state has the highest donor to population ratio?

Additional Dataset “Data” from online source: https://www.census.gov/data/tables/2017/demo/popest/state-total.html#par_textimage (https://www.census.gov/data/tables/2017/demo/popest/state-total.html#par_textimage)

Hide

```
population <- fread("io/nst-est2017-01.csv")
population$Population <- as.numeric(gsub(",","",population$Population))
donors.state <- as.data.table(table(donors$`Donor State`))
names(donors.state) <- c("State", "Donors.Count")
donors.state <- donors.state[order(donors.state$Donors.Count, decreasing = TRUE),]
donors.state <- merge(donors.state,population,by="State")
donors.state$percentage <- donors.state$Donors.Count/donors.state$Population*100
donors.state <- donors.state[order(donors.state$percentage, decreasing = TRUE),]
donors.state <- merge(donors.state, state.total, by = "State")
donors.state <- merge(donors.state, state.avg, by = "State")
a <- summary(donors.state$donations.total)
donors.state$indicator <- NULL
donors.state[which(donors.state$donations.total < a[[2]]), 'indicator'] <- 'Low State
Total Donations'
donors.state[which(donors.state$donations.total >= a[[2]]), 'indicator'] <- 'Medium S
tate Total Donations'
donors.state[which(donors.state$donations.total >= a[[5]]), 'indicator'] <- 'High Sta
te Total Donations'
```

Hide

```
# 3.1 Graphically Display the distribution of Donor Ratio per State
hist(donors.state$percentage, breaks = 15, main = "Histogram of States' Donor/ State
Population Ratio", xlab= "Donor/ State Population Ratio")
donors.percentage <- donors.state$percentage[-1]
normal.percentage <- rnorm(length(donors.percentage), mean(donors.percentage), var(do
nors.percentage))
qqPlot(donors.percentage, xlab = "Theoretical Quantiles",
ylab = "Observed Quantiles", main = "QQ-Plot: Donor/ State Population Ratio Quantile
vs Normal Quantile")
```

Hide

```
# 3.2 Scatter Plot: Indicate states that has special donating pattern
#qqplot(donors.state$donations.avg, donors.state$percentage)
qplot(percentage, donations.avg, colour = indicator, data = donors.state,main = 'Scat
terPlot of Donor/State Population Ratio Vs. State Average Donations', xlab = "Donor/S
tate Population Ratio", ylab = "State Average Donations")
```

Scenario Regression model

Predict donors’ donating behaviors to figure out how much a donor would donate for a project.

Hide

```
# 4.1 In total, find out the states that have the most donations.
# Figure out the rank of total donation amount for each state
state.total <-  aggregate(list(donations.total=df_merged$Donation.Amount), by=list(St
ate=df_merged$Donor.State), FUN= sum)
state.total <-  state.total[order(state.total$donations.total, decreasing = TRUE),]
summary(state.total$donations.total)
```

Hide

```
# 4.2 On average, find out the states that have the most donation per transaction.
# Figure out the rank of average donation amount per transaction for each state
state.avg <-  aggregate(list(donations.avg=df_merged$Donation.Amount), by=list(State=
df_merged$Donor.State), FUN= mean)
state.avg <- state.avg[order(state.avg$donations.avg, decreasing = TRUE),]
summary(state.avg$donations.avg)
```

Hide

```
# 4.3 Multiple Regression on predicting how much donations a project can receive
test <- merged[which(merged$Project.Current.Status == 'Fully Funded'),]
sample <- test[sample(nrow(merged), 10000), ]
sample$expire.duration <- as.numeric(as.Date(sample$Project.Expiration.Date)-as.Date(
sample$Project.Posted.Date))
sample$funded.duration <- as.numeric(as.Date(sample$Project.Fully.Funded.Date)-as.Dat
e(sample$Project.Posted.Date))
```

Hide

```
sample=na.omit(sample)
ols = lm(formula = donations ~ Project.Type + Project.Grade.Level.Category+  Project.
Cost + School.Percentage.Free.Lunch, data = sample)
summary(ols)
```

```
Call:
lm(formula = donations ~ Project.Type + Project.Grade.Level.Category +
    Project.Cost + School.Percentage.Free.Lunch, data = sample)

Residuals:
    Min      1Q  Median      3Q     Max
-4690.5   -67.2    23.4   116.1  7654.9

Coefficients:
                                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                               -2.634e+02  4.174e+01  -6.312 2.89e-10 ***
Project.TypeStudent-Led                    7.121e+01  5.272e+01   1.351   0.1768
Project.TypeTeacher-Led                    2.533e+02  4.040e+01   6.270 3.80e-10 ***
Project.Grade.Level.CategoryGrades 6-8     2.054e+00  1.028e+01   0.200   0.8416
Project.Grade.Level.CategoryGrades 9-12    2.353e+01  1.162e+01   2.026   0.0428 *
Project.Grade.Level.CategoryGrades PreK-2  1.995e+01  8.092e+00   2.465   0.0137 *
Project.Grade.Level.Categoryunknown       -1.223e+01  3.090e+02  -0.040   0.9684
Project.Cost                               6.134e-01  4.732e-03 129.623  < 2e-16 ***
School.Percentage.Free.Lunch              -5.148e-02  1.399e-01  -0.368   0.7128
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 308.9 on 8221 degrees of freedom
Multiple R-squared:  0.6752,    Adjusted R-squared:  0.6749
F-statistic:  2136 on 8 and 8221 DF,  p-value: < 2.2e-16
```

Hide

```
# 4.4 Ridge Regression on predicting how much a donor will donate
ridge <- lm.ridge (donations ~ Project.Type + Project.Grade.Level.Category+ Project.R
esource.Category + Project.Cost+ School.Metro.Type + School.Percentage.Free.Lunch, da
ta = sample, lambda = seq(0, 50, .001))
plot(ridge)
select(ridge)
```

Hide

```
set.seed(2018)
#sample <- sample[,-15]
x=model.matrix(donations~.-1,data=sample)
y=sample$donations
fit.ridge=glmnet(x,y,alpha=0)
plot(fit.ridge,xvar="lambda",label=TRUE)
cv.ridge=cv.glmnet(x,y,alpha=0)
plot(cv.ridge)
```

Hide

```
# 4.5 Lasso Regression on predicting how much a donor will donate
fit.lasso=glmnet(x,y,alpha=1)
plot(fit.lasso,xvar="lambda",label=TRUE)
cv.lasso=cv.glmnet(x,y)
plot(cv.lasso)
coef(cv.lasso)
n <- nrow(x)
train <- sample(1:n, size = floor(0.6*n), replace = FALSE) # use 60% data for trainin
g
lasso.tr=glmnet(x[train,],y[train], alpha = 1)
lasso.tr
pred=predict(lasso.tr,x[-train,])
# calculate the mse
rmse= sqrt(apply((y[-train]-pred)^2,2,mean))
plot(log(lasso.tr$lambda),rmse,type="b",xlab="Log(lambda)")
lam.best=lasso.tr$lambda[order(rmse)[1]]
lam.best
coef(lasso.tr,s=lam.best)
```

Scenario Donation Prediction

Hide

```
# 6.1 Test of Independence Checking if the funding duration time is independent of pr
oject cost
# Clean out the outliers
test <- merged[which(merged$Project.Current.Status == 'Fully Funded'),c(4,12,13,14,16
)]
test$expire.duration <- as.numeric(as.Date(test$Project.Expiration.Date)-as.Date(test
$Project.Posted.Date))
test$funded.duration <- as.numeric(as.Date(test$Project.Fully.Funded.Date)-as.Date(te
st$Project.Posted.Date))
test <- test[,-c(3,4,5)]
outlier.upper <- as.numeric(apply(test, FUN = mean, MARGIN = 2)+ 3* apply(test, FUN =
sd, MARGIN = 2))
outlier.lower <- as.numeric(apply(test, FUN = mean, MARGIN = 2)- 3* apply(test, FUN =
sd, MARGIN = 2))

test <- test[which(donations<outlier.upper[1]),]
test <- test[which(donations>outlier.lower[1]),]
test <- test[which(Project.Cost<outlier.upper[2]),]
test <- test[which(Project.Cost>outlier.lower[2]),]
test <- test[which(expire.duration<outlier.upper[3]),]
test <- test[which(expire.duration>outlier.lower[3]),]
test <- test[which(funded.duration<outlier.upper[4]),]
test <- test[which(funded.duration>outlier.lower[4]),]

# Attempt for a for loop but failed
# for (i in c(1:length(outlier.lower))){
#   test <- test[which(test[,i]>outlier.lower[i]),]
#   test <- test[which(test[,i]<outlier.upper[i]),]
# }
```

Hide

```
par(mfrow=c(1,2))
boxplot(test[,c(1,2)])
boxplot(test[,c(3,4)])

par(mfrow=c(2,3))
hist(test$expire.duration, main = "Histogram of projects' posted to expire duration",
xlab = "time duration from post to proposed expiration date", ylab = "frequency")
hist(test$funded.duration, main = "Histogram of projects' posted to fund duration",xl
ab = "time duration from post to fully funded", ylab = "frequency")
hist(test$donations, main = "Histogram of Total Donations collected for projects",xla
b = "total donations collected for project", ylab = "frequency")
hist(test$Project.Cost, main = "Histogram of Total Cost for projects",xlab = "Project
Cost", ylab = "frequency")
hist(log(test$Project.Cost), main = "Histogram of log(Cost) for projects",xlab = "log
(Project Cost)", ylab = "frequency" )
hist(log(test$donations), main = "Histogram of log(Donations) for projects",xlab = "l
og(Total Donations)", ylab = "frequency")

par(mfrow= c(2,2))
qqplot(test$donations,test$Project.Cost,main = "Quantile-Quantile Plot of donations v
s. cost",xlab = "Total Donations", ylab = "Project Cost")
qqplot(test$donations,test$expire.duration,main = "Quantile-Quantile Plot of donation
s vs. time duration to expire",xlab = "Total Donations", ylab = "time duration to exp
iration")
qqplot(test$donations, test$funded.duration, main = "Quantile-Quantile Plot of donati
ons vs. time duration fully funded",xlab = "Total Donations", ylab = "time duration t
o fully funded")
qqplot(test$Project.Cost,test$funded.duration,main = "Quantile-Quantile Plot of proje
ct cost vs funded duration",xlab = "project cost)", ylab = "time duration to fully fu
nded")

par(mfrow = c(1,1))
test_avg <- aggregate(list(donations=test$donations), by=list(Project.Cost=round(test
$Project.Cost)), FUN=mean)
test_avg <- test_avg[which(test_avg<=1590),]
least.squares <- lm(donations~Project.Cost, data=test_avg)
lad <- lad(donations~Project.Cost, data=test_avg)
quant <- rq(donations~Project.Cost, tau=.5, data=test_avg)

plot(test_avg$Project.Cost, test_avg$donations, main = "Least Squares Fitting Regress
ion Line", xlab = "Project.Cost", ylab = "donations")
abline(least.squares, col='red')
abline(lad, col='blue')
abline(quant, col='green')
summary(least.squares)
legend('topright', legend=c('Least Squares Regression Line', 'Least Absolute Deviatio
ns Regression Line', '50% Quantile Regression Line'), col=c('red', 'blue', 'green'),
lty=1)

least.squares.residuals <- data.frame(test_avg['Project.Cost'], test_avg['donations']
- rep(predict(least.squares)))

title.residuals1 <- 'Residuals of Least Squares Regression Line'
plot(least.squares.residuals, main=title.residuals1, xlim = c(0,1600) ,ylim = c(-1000
,1000))
abline(0, 0, col='red')
summary(least.squares.residuals)
```

Scenario: Recommender System

Hide

```
# Merge datasets
df <- merge(projects.categorized, projects[, c(1,2,12,14)], by='Project ID')
df <- merge(df, donations[, c(1,3,5)], by='Project ID')[, -1]
df <- merge(df, donors[, c(1,4)], by='Donor ID')[, -1]
df <- merge(df, schools[, c(1,3,4)], by='School ID')[, -1]

# Omit invalid values
df <- na.omit(df)
df <- df[`Project Grade Level Category` != 'unknown'][`School Metro Type` != 'unknown
']

# Arrange columns by category, numerical attribute, categorical attribute
df <- df[, c(1,3,4,7,5,6,2)]

# Convert some categorical columns to logical
df$`Donor Is Teacher` <- df$`Donor Is Teacher` == 'Yes'
df$`School Metro Type` <- df$`School Metro Type` == 'suburban'

# Rename columns
colnames(df) <- c('Subject.Category', 'Project.Cost', 'Donation.Amount', 'School.Perc
entage.Free.Lunch', 'Donor.Is.Teacher', 'School.Is.Suburban', 'Grade.Level')

# Convert categorical columns to factor
df$Subject.Category = as.factor(df$Subject.Category)
df$Grade.Level = as.factor(df$Grade.Level)
df
```

Hide

```
set.seed(0)
training.size <- 50000
num.trees <- 20

# Randomly partition into training and testing set
df <- df[sample(nrow(df))]
df.training <- df[1:training.size]
df.testing <- df[(training.size+1):nrow(df)]

predictions <- data.table(Subject.Category=df.testing$Subject.Category)

# Produce a random forest (of regression trees) for each subject category.
# Outputs probability of a project being that subject category given project and
# donor attributes.
for (category in categories) {
  df.category <- df.training
  df.category$Subject.Category <- df.category$Subject.Category == category

  model <- randomForest(Subject.Category ~ ., data=df.category, ntree=num.trees)

  title <- paste('Random Forest (of Regression Trees) for', category)

  print(getTree(model, labelVar=TRUE))
  plot(model, main=title, cex.main=.9)
  varImpPlot(model, main=title, cex.main=.9)

  predictions[, category] <- predict(model, df.testing)
}
```

Hide

```
# Predict subject category by selecting the category with the highest random
# forest (of regression trees) probability of being in that category.
predictions$Predicted.Category <- colnames(predictions[, -1])[max.col(predictions[, -
1])]
predictions <- predictions[, c(1,10,2:9)]
predictions
```

Hide

```
# Cross-Validation
correct <- sum(predictions$Subject.Category == predictions$Predicted.Category)
correct.column <- correct
total.column <- nrow(predictions)

for (category in categories) {
  predictions.category <- predictions[Subject.Category == category]

  correct <- sum(predictions.category$Subject.Category == predictions.category$Predic
ted.Category)
  correct.column <- c(correct.column, correct)
  total.column <- c(total.column, nrow(predictions.category))
}

categories.accuracy <- data.table(Subject.Category=c('All Categories', categories),
                                  Correct.Predictions=correct.column,
                                  Total=total.column)
categories.accuracy$Accuracy <- categories.accuracy$Correct.Predictions / categories.
accuracy$Total
categories.accuracy

par(mai=c(1,2,.5,.5))
barplot(categories.accuracy$Accuracy,
        names.arg=categories.accuracy$Subject.Category, horiz=TRUE, las=1,
        main='Cross-Validation Accuracy of Recommender System',
        xlab='Proportion of Correct Classifications', axis.lty=1)
```

Scenario: EDA

Hide

```
state.total <- state.total[order(state.total$donations.total, decreasing = TRUE),]
ggplot(data=state.total, aes(x=state.total$State, y=state.total$donations.total)) +ge
om_bar(stat="identity")+ coord_flip()
merged[order(merged$donations, decreasing = TRUE),][1,]
```