W203 Lab 1: Candidate Debt EDA

Jon Mease
James De La Torre
Adam Yang

Introduction

As members of the campaign committee for the upcoming Washington state election, we are interested in the nature of the debt that candidates for office have taken on in past elections. This report contains the findings of our initial exploratory data analysis of a dataset from the 2012 election.

Research question: How can the amount of debt taken on by candidates in the 2012 Washington state election be understood in relation to all of the other available variables in this data set?

Load data set

The data set we are investigating is named `CandidateDebt.csv` and it is located in the `datasets` directory distributed alongside this report.

First, we use R to load the data set as the `DebtRaw` variable, treating the string `'#N/A'` as a missing value.

```r
DebtRaw = read.csv("../dataset/CandidateDebt.csv", na.strings='#N/A')
print(sprintf("Number of columns: %d, Number of rows: %d", ncol(DebtRaw), nrow(DebtRaw)))
```

```
## [1] "Number of columns: 28, Number of rows: 1043"
```

Align column labels

Next, we examine the first 4 rows of the dataset across all columns

```r
str(DebtRaw[1:4,])
```

```
## 'data.frame':    4 obs. of  28 variables:
##  $ reportnumber       : int  100495995 100496548 100498383 100495987
##  $ origin             : Factor w/ 1 level "B.3": 1 1 1 1
##  $ filerid            : Factor w/ 141 levels "ASHAK  359","BILLA2 203",..: 110 129 30 122
##  $ filertype          : Factor w/ 1 level "Candidate": 1 1 1 1
##  $ filername          : Factor w/ 134 levels "ASHABRANER KARIN L",..: 105 124 31 117
##  $ firstname          : Factor w/ 106 levels "ACHIYAMMA","ALLEN",..: 19 103 43 99
##  $ middleinitial      : Factor w/ 23 levels "","A","B","C",..: 19 15 4 5
##  $ lastname           : Factor w/ 129 levels "ASHABRANER","AXTHELM",..: 101 119 30 113
##  $ office             : Factor w/ 16 levels "APPEALS COURT JUDGE",..: 12 4 12 6
##  $ legislativedistrict: Factor w/ 14 levels "ATTORNEY GENERAL",..: 11 11 11 11
##  $ position           : int  1 1 1 1
##  $ party              : int  NA NA NA NA
##  $ jurisdiction       : Factor w/ 4 levels "DEMOCRAT","INDEPENDENT",..: 4 4 4 4
##  $ jurisdictioncounty : Factor w/ 51 levels "ATTORNEY GENERAL, OFFICE OF",..: 10 10 10 10
##  $ jurisdictiontype   : Factor w/ 15 levels "","BENTON","CLALLAM",..: 6 6 6 6
##  $ electionyear       : Factor w/ 4 levels "Judicial","Legislative",..: 2 2 2 2
##  $ amount             : int  2012 2012 2012 2012
##  $ recordtype         : num  283 283 283 283
##  $ fromdate           : Factor w/ 1 level "DEBT": 1 1 1 1
```

```
##  $ thrudate           : Factor w/ 37 levels "1/1/10","1/1/11",..: 30 30 30 30
##  $ debtdate           : Factor w/ 30 levels "1/31/10","1/31/11",..: 24 24 24 24
##  $ code               : Factor w/ 72 levels "1/21/11","1/22/10",..: 61 61 61 61
##  $ description        : Factor w/ 4 levels "","Fundraising",..: 1 1 1 1
##  $ vendorname         : Factor w/ 106 levels "","$750 PER MONTH THROUGH OCTOBER",..: 73 73 73 73
##  $ vendoraddress      : Factor w/ 75 levels "ABBOT TAYLOR",..: 26 26 26 26
##  $ vendorcity         : Factor w/ 80 levels "","10 SABLE COURT ",..: 68 68 68 68
##  $ vendorstate        : Factor w/ 30 levels "","BAINBRIDGE ISLAND",..: 30 30 30 30
##  $ vendorzip          : Factor w/ 5 levels "","CA","DC","TX",..: 5 5 5 5
```

Based on the column names, factor levels, and the first few values it appears that the column labels of this data set are well aligned over the first 9 columns (`reportnumber` through `office`). There seems to be an alignment error, however, beginning with the `legislativedistrict` column. The first factor level of this column, `"ATTORNEY GENERAL"`, is not sensible as a legislative district. Instead we expect legislative districts to be identified by integers, as is the case for the column immediatly after `legislativedistrict`.

After careful examination, it is clear that the column labels from `legislativedistrict` through `vendorstate` should all be shifted forward by one position. In this shift we remove the trailing `vendorzip` column label, which is appropriate since we do not have any data columns consistent with zip codes. We must also introduce a new label for the data column previously labeled by `jurisdiction`. We will choose the name `DUMMY` for the time being, and will introduce a more meaningful name once we have identified the column's role.

```r
DebtShifted = DebtRaw
colnames(DebtShifted) <- c(colnames(DebtRaw)[1:9],
                           'DUMMY',
                           colnames(DebtRaw)[10:(ncol(DebtRaw)-1)])
```

We once again examine the first 4 rows of the data set, and note that the column labels are now well aligned with the data columns. We discuss the columns in detail in the following section.

```r
str(DebtShifted[1:4,])
```

```
## 'data.frame':    4 obs. of  28 variables:
##  $ reportnumber       : int  100495995 100496548 100498383 100495987
##  $ origin             : Factor w/ 1 level "B.3": 1 1 1 1
##  $ filerid            : Factor w/ 141 levels "ASHAK  359","BILLA2 203",..: 110 129 30 122
##  $ filertype          : Factor w/ 1 level "Candidate": 1 1 1 1
##  $ filername          : Factor w/ 134 levels "ASHABRANER KARIN L",..: 105 124 31 117
##  $ firstname          : Factor w/ 106 levels "ACHIYAMMA","ALLEN",..: 19 103 43 99
##  $ middleinitial      : Factor w/ 23 levels "","A","B","C",..: 19 15 4 5
##  $ lastname           : Factor w/ 129 levels "ASHABRANER","AXTHELM",..: 101 119 30 113
##  $ office             : Factor w/ 16 levels "APPEALS COURT JUDGE",..: 12 4 12 6
##  $ DUMMY              : Factor w/ 14 levels "ATTORNEY GENERAL",..: 11 11 11 11
##  $ legislativedistrict: int  1 1 1 1
##  $ position           : int  NA NA NA NA
##  $ party              : Factor w/ 4 levels "DEMOCRAT","INDEPENDENT",..: 4 4 4 4
##  $ jurisdiction       : Factor w/ 51 levels "ATTORNEY GENERAL, OFFICE OF",..: 10 10 10 10
##  $ jurisdictioncounty : Factor w/ 15 levels "","BENTON","CLALLAM",..: 6 6 6 6
##  $ jurisdictiontype   : Factor w/ 4 levels "Judicial","Legislative",..: 2 2 2 2
##  $ electionyear       : int  2012 2012 2012 2012
##  $ amount             : num  283 283 283 283
##  $ recordtype         : Factor w/ 1 level "DEBT": 1 1 1 1
##  $ fromdate           : Factor w/ 37 levels "1/1/10","1/1/11",..: 30 30 30 30
##  $ thrudate           : Factor w/ 30 levels "1/31/10","1/31/11",..: 24 24 24 24
##  $ debtdate           : Factor w/ 72 levels "1/21/11","1/22/10",..: 61 61 61 61
##  $ code               : Factor w/ 4 levels "","Fundraising",..: 1 1 1 1
##  $ description        : Factor w/ 106 levels "","$750 PER MONTH THROUGH OCTOBER",..: 73 73 73 73
```

```
## $ vendorname         : Factor w/ 75 levels "ABBOT TAYLOR",..: 26 26 26 26
## $ vendoraddress      : Factor w/ 80 levels "","10 SABLE COURT ",..: 68 68 68 68
## $ vendorcity         : Factor w/ 30 levels "","BAINBRIDGE ISLAND",..: 30 30 30 30
## $ vendorstate        : Factor w/ 5 levels "","CA","DC","TX",..: 5 5 5 5
```

Describe variables

Next, a brief overview of each of these 28 variables. The initial description in *italics* is the description provided in the `CandidateDebt.pdf` documentation provided alongside the dataset.

- `reportnumber`: *identifier used for tracking the individual form*
  An integer with values like 100495995, 100496548, etc. Contains no duplicate or missing values.
- `origin`: *This field shows from which filed report-type the data originates.*
  A string with a single constant value, `"B.3"`, and no missing values.
- `filerid`: *The unique id assigned to a candidate*
  A unique candidate id string that is somewhat based on the candidate's name. Of the 1043 rows, there are only 141 unique `filerid` values and no missing values.
- `filertype`: *Indicates if this record is for a candidate*
  A string with a single constant value, `"Candidate"`, and no missing values.
- `filername`: *The candidate or committee name as reported on the candidates registration.*
  A string with with 134 unique values and no missing values.
- `firstname`: *This field represents the first name, as reported by the filer*
  A string with no missing values.
- `middleinitial`: *This field represents the middle initial, as reported by the filer*
  A string with no missing values.
- `lastname`: *This field represents the last name, as reported by the filer*
  A string with no missing values.
- `office`: *The office sought by the candidate*
  A string with 16 unique values (`"GOVERNOR"`, `"STATE SENATOR"`, `"COUNTY SHERIFF"`, etc.) and no missing values.
- `DUMMY`:
  This is the placeholder column label that was introduced above in order to align the subsequent column labels with their data columns. There are 14 unique `DUMMY` values present in the dataset, and these 14 values are a proper subset of the 16 unique values present in the `office` variable described above.
- `legislativedistrict`: *The Washington State legislative district*
  An integer identifiying the legislative district ranging from 1 to 48 with 354 missing values.
- `position`: *The position associated with an office*
  An integer identifying the office's position raning from 1 to 40 with 574 missing values.
- `party`: *The political party as declared by the candidate on their registration*
  A string with 4 unique values (`"DEMOCRAT"`, `"REPUBLICAN"`, `"INDEPENDENT"`, and `"NON PARTISAN"`) and 56 missing values.
- `jurisdiction`: *The political jurisdiction associated with the office of a candidate*
  A string with 51 unique values (`"LEG DISTRICT 11 - SENATE"`, `"LEG DISTRICT 41 - HOUSE"`, `"SUPREME COURT"`, etc.) and 56 missing values.
- `jurisdictioncounty`: *The county associated with the jurisdiction of a candidate*
  A string with 15 unique values (`"KING"`, `"PIERCE"`, `"SPOKANE"`, etc.). There are also 56 missing values and an additional 215 empty values.

- `jurisdictiontype`: *The type of jurisdiction this office is: Statewide, Local, etc*
  A string with 4 unique values (`"Statewide"`, `"Legislative"`, `"Local"`, and `"Judicial"`) and 56 missing values.
- `electionyear`: *The election year in the case of candidates*
  An integer with the constant value 2012 and 56 missing values.
- `amount`: *The amount of the debt incurred or order placed*
  A floating point number representing the debt incurred or spending in dollars. Values range from 3.24 to 19000.00 and there are 56 missing values.
- `recordtype`: *This field designates the item as a debt*
  A string with a single unique value, `"DEBT"` and 56 missing values.
- `fromdate`: *The start date of the period for the report on which this debt record was reported*
  A Date that ranges from 2009-10-01 to 2012-08-01 with 56 missing values.
- `thrudate`: *The end date of the period for the report on which this debt record was reported*
  A Date that range from 2009-10-31 to 2012-08-31 with 56 missing values.
- `debtdate`: *The date that the debt was incurred*
  A Date that range from 2008-10-29 to 2012-08-31 with 56 missing values.
- `code`: *The type of debt*
  A string with 3 unique values (`"Operation and Overhead"`, `"Management Services"`, and `"Fundraising"`). In addition, there are 56 missing values and 610 empty values.
- `description`: *The reported description of the transaction*
  A string with 106 unique values (`"YARD SIGNS"`, `"STAMPS`, `"AIRFARE"`, etc.), 56 missing values, and 39 empty values.
- `vendorname`: *The name of the vendor or recipient's name*
  A string with 75 unique values (`"SEATTLE MEDIUM NEWSPAPER"`, `"IMPACT SIGNS"`, `"THE CONNECTIONS GROUP"`, etc.) and 56 missing values.
- `vendoraddress`: *The street address of the vendor or recipient*
  A string with 80 unique values (e.g. `"PO BOX 650448"`, `"5810 COWAN PL NE"`, `"2600 S JACKSON ST"`, etc.), 56 missing values, and 24 empty values.
- `vendorcity`: *The city of the vendor or recipient*
  A string with 30 unique values (`"SAN JOSE"`, `"SEATTLE"`, `"TUMWATER"`, etc.), 56 missing values, and 24 empty values.
- `vendorstate`: *The state of the vendor or recipient*
  A string with 4 unique values (`"WA"`, `"DC"`, `"TX"`, and `"CA"`), 56 missing values, and 25 empty values.

Examine row alignment

Next, we examine select columns from the first row of the data set.

```
t(DebtShifted[1,c('filerid', 'filername', 'office', 'DUMMY', 'legislativedistrict',
                  'party', 'jurisdiction', 'jurisdictiontype')])
```

```
##                     1
## filerid             "RYU C  133"
## filername           "RYU CINDY S"
## office              "STATE REPRESENTATIVE"
## DUMMY               "STATE SENATOR"
## legislativedistrict "1"
## party               "REPUBLICAN"
```

```
## jurisdiction        "LEG DISTRICT 01 - SENATE"
## jurisdictiontype    "Legislative"
```

Notice that the `office` column indicates that a candidate named Cindy Ryu ran for `"STATE REPRESENTATIVE"`. The `party`, `jurisdiction`, and `legislativedistrict` columns indicate that she ran for state senate as a Republican in district 1. However, a quick wikipedia search shows that Cindy Ryu atually ran for state representative in 2012 as a Democrat in district 32.

After careful examination of many individual rows, we have concluded that the values for each row are self consistent from `reportnumber` through `office` (e.g. Candidate with `filername` of `"RYU CINDY S"` did run for the `office` of `"STATE REPRESENTATIVE"`), but they are inconsistent from `DUMMY` through `vendorstate` (Cindy Ryu did not have `party` of `"REPUBLICAN"` and did not run for an office with a `jurisdiction` of `"LEG DISTRICT 01 - SENATE"`).

The columns from `DUMMY` through `vendorstate` also exhibit peculiar repition and missing value behavior. All of these colums have 56 missing values, and they occur in the same 56 rows. In contrast, there are no missing values in any of the columns before ‘`DUMMY`. Also, there are many rows that have distinct values from `reportnumber` through `office` but identical values from `DUMMY` through `vendorstate`.

Based on these missing value and repitition observations, and the need to introduce the `DUMMY` column in the first place, we speculate that this data set is actually the combination of two separate data sets that are improperly aligned, perhaps as the result of an errant SQL join. We further conclude that there is not sufficient information available to realign these two data sets.

To move forward, we will define two new data sets

1. `DebtLeft` will include all rows for the columns from `reportnumber` through `office`.

```r
# Split off left table
DebtLeft = DebtShifted[, seq(9)]

# Print out number of rows and columns
print(sprintf("The DebtLeft data set has %d columns and %d rows.", ncol(DebtLeft), nrow(DebtLeft)))
```

```
## [1] "The DebtLeft data set has 9 columns and 1043 rows."
```

2. `Debt` will include the unique, non-missing, rows for the columns from `DUMMY` through `vendorstate`. Additionally, we will rename the `DUMMY` column to `office` since it has a nearly identical set of unique values as the `office` column in `DebtLeft`, and seems to serve the same purpose.

```r
# Split off right table
DebtRight = DebtShifted[, seq(10, ncol(DebtRaw))]

# Rename DUMMY to office
colnames(DebtRight)[[1]] <- "office"

# Keep cases with at least one non-NA entry
DebtRightValid <- DebtRight[rowSums(!is.na(DebtRight)) > 0,]

# Drop duplicates
Debt <- unique(DebtRightValid)

# Print number of rows and columns
print(sprintf("The Debt data set has %d columns and %d rows.", ncol(Debt), nrow(Debt)))
```

```
## [1] "The Debt data set has 19 columns and 194 rows."
```

While we will explore both `Debt` and `DebtLeft` in the analysis to follow, the `Debt` data set is of more utility to us because it contains `amount`, the target variable of this analysis.

Univariate Analysis of Key Variables

In the introduction, we mentioned how the dataset we received actually consists of 2 misaligned tables. We subsequently split up the two tables into `DebtLeft` and `Debt` where the latter holds the `amount` column which is the dependant variable of our research question. Although the `DebtLeft` table is not aligned with the `amount` variable, it can still give us some insight about the nature of our dataset.

Who are the candidates and how many reports did they file?

The first thing we needed to do was get a sense of what each row of the `DebtLeft` table means.

```
# Find number of unique reportnumbers
length(unique(DebtLeft$reportnumber))
```

```
## [1] 1043
```

```
# Find number of unique cadidate names
length(unique(DebtLeft$filername))
```

```
## [1] 134
```

There are 1043 rows to the `DebtLeft` table and as shown above, there are a total of 1043 unique report numbers, one for each row. Every report number is associated with a candidate, but we have found that there are only 134 unique candidates in the table. This suggests that each row of the data set represents a B.3 report filed by a candidate and some of the candidates filed multiple reports.

To visualize this, we can aggregate the data into a dataframe showing the number of reports filed by each candidate. The summary of this aggregated dataframe (shown below) tells us that the minimum number of reports filed by a candidate is 1, while the maximum filed is 38. The median of the sample is 6 while the mean is 7.784 which suggests there is a skew towards the right of the distribution.

```
# Find the number of reports filed by each candidate
RepPerCan <- aggregate(reportnumber ~ filername, DebtLeft, length)
colnames(RepPerCan)[2] <- "NumOfReports"
# Summarize data
summary(RepPerCan$NumOfReports)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   6.000   7.784  10.000  38.000
```

Before looking at the histogram, we decided to take a look at the 5 candidates that filed the most reports as shown below. Unfortunately, our dataset does not allow us to determine which candidates occured the most debt. Because of the misalignemnt with the two halves of our dataset, we can only get a sense of which candidates filed the most debt reports. Furthermore, it is very possible that our dataset is missing a lot of debt reports so these results are only reflective of the sample we received.

```
# Show top 5 candidates that filed the most reports
head(RepPerCan[order(RepPerCan$NumOfReports, decreasing = TRUE),],5)
```

```
##               filername NumOfReports
## 39     GOLDMARK PETER J           38
## 6          BROWN LISA J           34
## 94 PRENTICE MARGARITA L           30
## 76     MCINTIRE JAMES L           28
## 11        CHOPP FRANK V           23
```

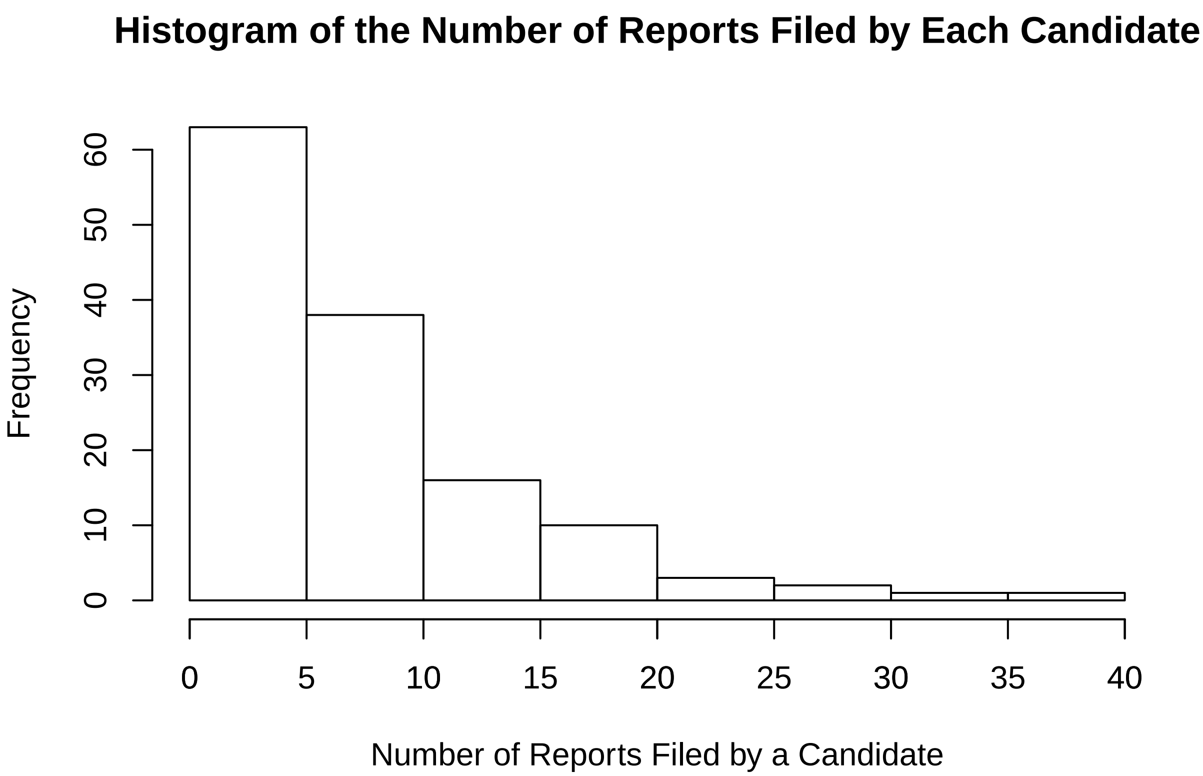
Because of the strong skew towards the right of the data set, we are expecting to see many candidates that filed only one report. Instead of listing all of these candidates, we found that there are 22 candidates that only filed one report, which is over 16% of the total reports filed in our sample (shown below).

```
# How many candidates filed only 1 report?
length(which(RepPerCan$NumOfReports == 1))
```

```
## [1] 22
```

The figure below is a histogram showing the number of reports filed by each candidate and it confirmed that there is indeed a skew towards the right of the distribution. We can conclude that most candidates filed between 1 and 5 reports and very few filed over 20 reports. This gives us a bit of an insight about the amount of debt incurred by the candidates. For the candidates in our sample, we now understand that the majority of them will not be filing many reports but there are a few that will file up to 38 reports.

```
# Show histogram showing distribution of the number of reports filed by each candidate.
hist(RepPerCan$NumOfReports, main = "Histogram of the Number of Reports Filed by Each Candidate",
     xlab = "Number of Reports Filed by a Candidate")
axis(1, at = seq(0,40, by = 5))
```



Which offices are being pursued by the candidates?

The next piece of information we can get from the `DebtLeft` dataset is how many candidates in our sample are running for each office. This can give us insight about the `Debt` table which consists of the debt `amount` variable because the office of the candidates is the only variable shared between the two datasets, even if they don't align.

To get a sense of how many candidates are applying for each office, we first have to get rid of the duplicate candidates. To do this, we can simply remove the `reportnumber` column and keep only the unique rows after that.

```
# Remove reportnumber and keep unique rows
Office <- unique(DebtLeft[-1])
```

```r
# Find length of rows.
length(Office$filerid)
```

```
## [1] 141
```

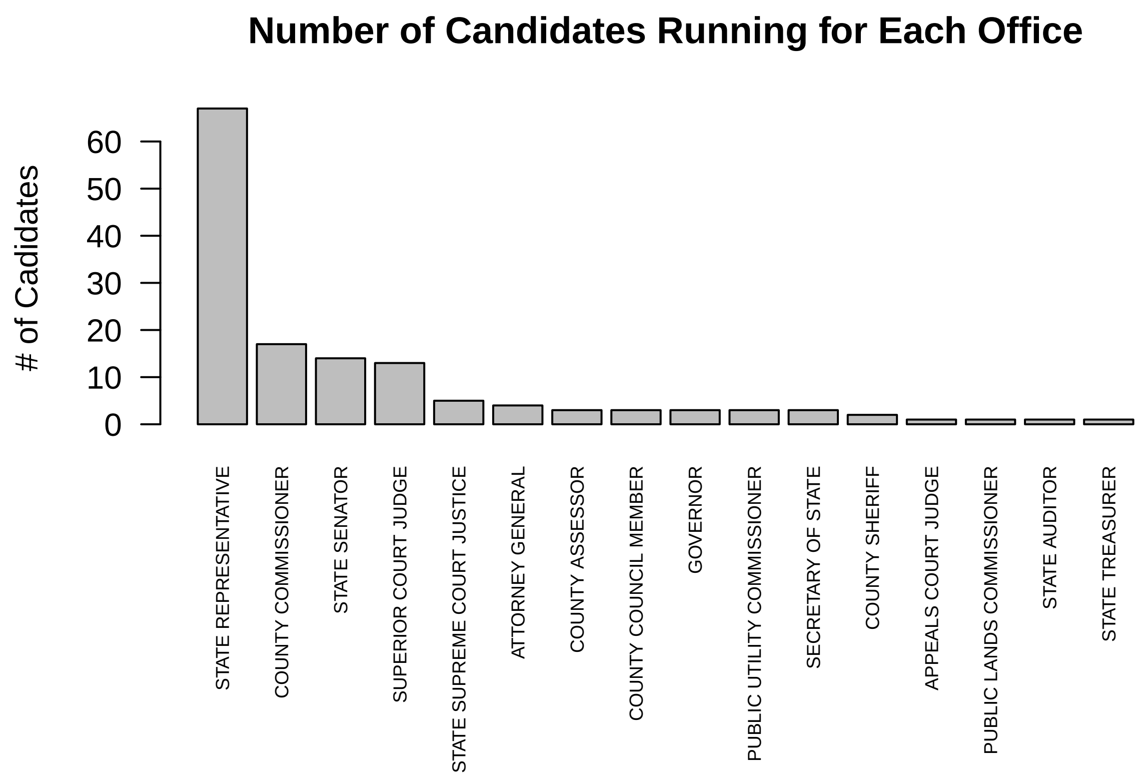
After creating the new dataframe with only the unique candidate rows, something strange occured where we ended up 141 rows instead of the expected 134. By doing a summary of `filername` variable, we can see that there are 7 candidates that have more than 1 row. Upon inspection, these 7 candidates each have 2 different `filerids` and had two different offices associated with them. An example is shown below. According to the two rows shown below, it looks like David S Frockt was running for State Representative but then switched to State Senator based on the numbering of the `filerid`. With some quick research, we found that David Frockt was a State Representative in 2010 but was elected as State Senator in the middle of 2011 after the death of Senator Scott White. This is very much a special situation, and it appears to have happened 7 times in our data set. We learned that candidates can apply for a different position within the time period of our sample.

```r
# Show exmaple of duplicate candidate.
Office[Office$filername == "FROCKT DAVID S",]
```

```
##     origin    filerid filertype      filername firstname middleinitial
## 81     B.3 FROCD2 111 Candidate FROCKT DAVID S     DAVID             S
## 355    B.3 FROCD  111 Candidate FROCKT DAVID S     DAVID             S
##     lastname                 office
## 81    FROCKT          STATE SENATOR
## 355   FROCKT STATE REPRESENTATIVE
```

Though we found these 7 instances where a candidate ran for two different positions within our sample, we did not feel the need to screen out these 7 candidates because we are only looking for the approxiate number of applicants for each position instead of exact values. We are only using the `DebtLeft` table to gain some insight on the `Debt` table so an approximate representation is all we want. Furthermore, there is no indication that screening out those candidates altogether, or only keeping one of their rows would result in a more accurate representation.

```r
# Create Dataframe showing the number of candidates running for each office
OfficePopularity <- aggregate(filername ~ office, Office, length)
colnames(OfficePopularity)[2] <- "N"
OfficePopularity <- OfficePopularity[order(OfficePopularity[,2], decreasing = TRUE),]

# Show barplot of results
op <- par(mar = c(10,4,4,2) + 0.1)
barplot(OfficePopularity$N, main = "Number of Candidates Running for Each Office", ylab = "# of Cadidat
      names.arg=OfficePopularity$office, las = 2, cex.names=0.6)
```



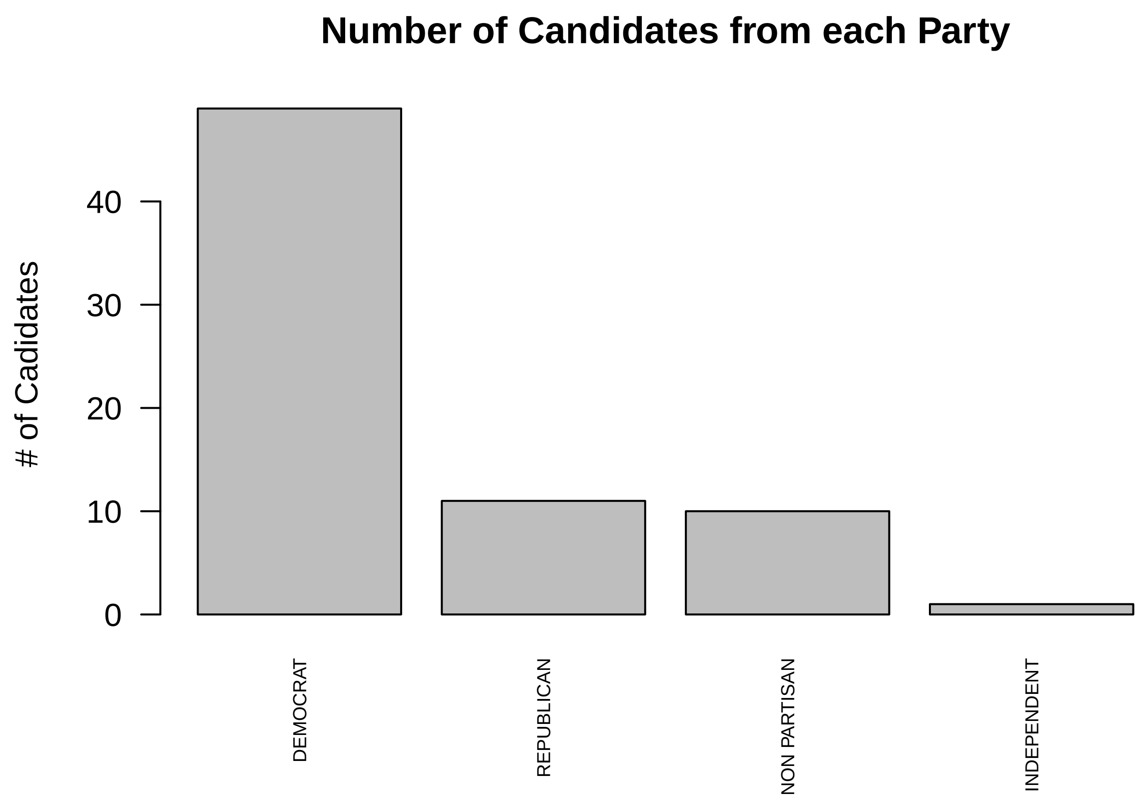

```
par(op)
```

The bar chart aboves shows that most of the candidates in our sampel are running for the State Representative position. This might be because there at 10 chairs open in the state of Washington compared to the 2 chairs open for the position of state senator. There is only 1 chair open for state auditor, but even so, it does not seem to be a highly contended position based on our sample.

Which party has the most candidates running for office?

Now we will start looking at the data in the `Debt` table (the one with vendor information and debt amount). We have no way of knowing which debt report was filed by which candidate and we know from our `DebtLeft` analysis above, that one candidate can file multiple reports. However, assuming that each candidate will have a unique combination of office, legislative dictrict, position, party, and jurisdiction, we can find the rows that are unique for all of these categories and assume each row is a candidate. Now we can plot a bar chart to get a sense of how many cadidates for each party are filing these debt reports.

```
# Only keep Office2, legislativedictrict, position, party, jurisdiction and find unique rows
DebtUnique <- unique(Debt[1:5])
# Aggregate a new dataframe to store the number of rows for each party
PartyCount <- aggregate(jurisdiction ~ party, DebtUnique, length)
colnames(PartyCount)[2] <- "N"
PartyCount <- PartyCount[order(PartyCount[,2], decreasing = TRUE),]
# Plot a bar chart showing the number of rows for each party
op <- par(mar = c(5,4,4,2) + 0.1)
barplot(PartyCount$N, main = "Number of Candidates from each Party", ylab = "# of Cadidates",
      names.arg=PartyCount$party, las = 2, cex.names=0.6)
```




```r
par(op)
```

Above, we are assuming that we have successfully filtered the dataset to one row per candidate but it is possible that two democrats are running for the exact same office in the exact same district, jurisdiction, and position, in which case they would be grouped as a single data point. However, regardless of the previous point, we think it is safe to claim that in our sample, the majority of our candidates are from the Democratic party, which makes sense since Washington is a blue state. However, since the number of republican and non-partisan candidates are very similar, we cannot claim which one has more candidates because of the uncertainty of our row manipulation.

Barplot of the debtdate by year-month

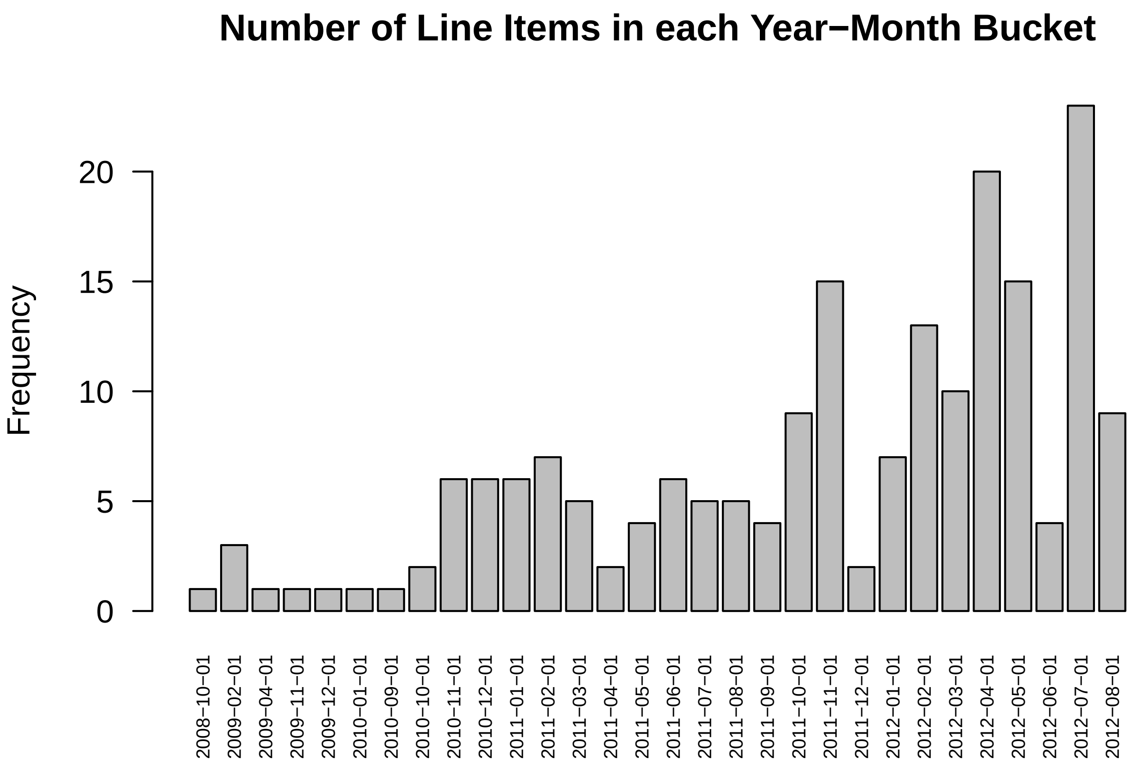
The next item we will analyze is the `debtdate` variable which is defined as the date that the debt was incurred. However, looking at the count of each date may be too discrete. Instead we can group them by month and year for easier visualization.

```r
# Copy Debt into a temporary df, but convert the debtdate variable into date object
DebtTemp <- Debt
DebtTemp$debtdate <- as.Date(DebtTemp$debtdate, format='%m/%d/%y')
# Import lubridate to round the dates to the month
library(lubridate)
```

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

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

```r
DebtTemp$debtdate <- floor_date(DebtTemp$debtdate, unit = "month")
# Aggregate the data to show the amount of line items are in each month
DateCount <- aggregate(amount ~ debtdate, DebtTemp, length)
colnames(DateCount)[2] <- "N"
op <- par(mar = c(5,4,4,2) + 0.1)
barplot(DateCount$N, main = "Number of Line Items in each Year-Month Bucket", ylab = "Frequency",
      names.arg=DateCount$debtdate, las = 2, cex.names=0.6)
```




```r
par(op)
```

From the barplot shown above, we can observe that in our sample, the number of line items are increased as we apprach the election date. This might suggest increased spending but we are not yet able to make that claim until we look at the debt amount over time in the bivariate analysis section.

Histogram of the debt amount

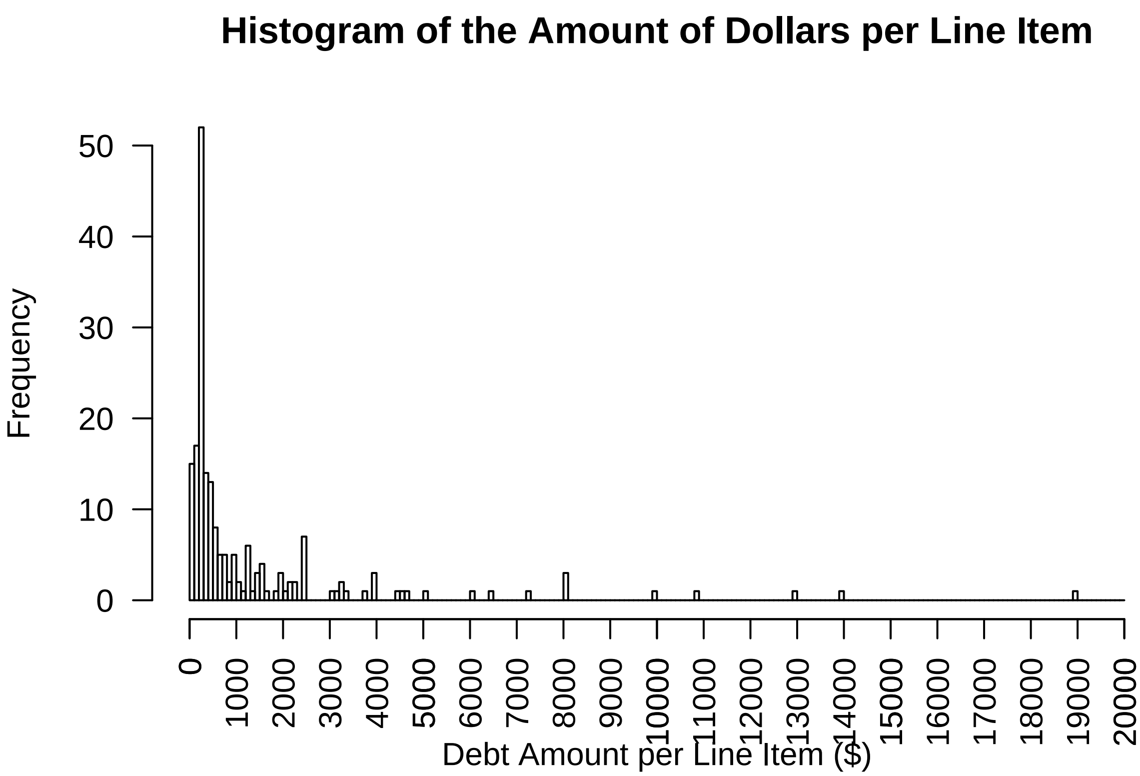
Finally we have reached the univariate analysis of our debt `amount` variable.

```r
summary(Debt$amount)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     3.24   271.88   400.00  1387.59  1328.41 19000.00
```

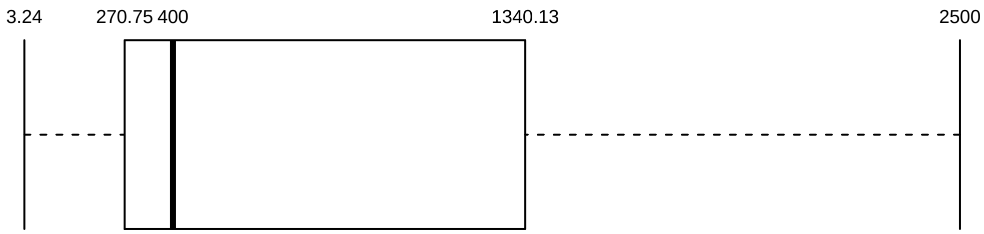
Before we plot the histogram, we took a look at the summary of this sampel. As noted above, the minimum debt reported was $3.24 while the maximum debt reported was $19,000. The median debt amount was $400 while the mean was much higher at $1387.59. This suggests a very strong skew to the right of the distribution and it is likely that the $19,000 amount is an outlier.

```r
hist(Debt$amount, breaks = seq(0, 20000, by = 100) , main = "Histogram of the Amount of Dollars per Lin
     xlab = "Debt Amount per Line Item ($)", las = 2)
axis(1, at = seq(0, 20000, by = 1000), las = 2)
```



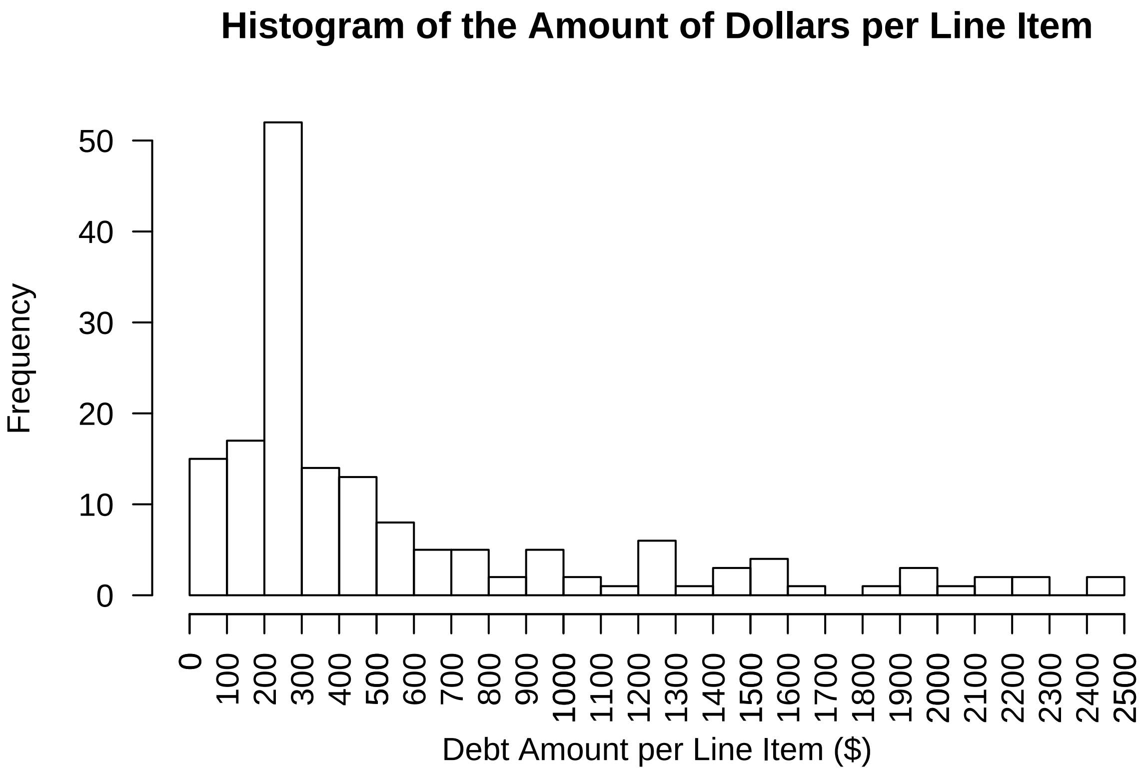
The first thing to do is to take a look at the histogram without filtering out any data. We can see that most of the data is between 0 and 2000 dollars while there are some outliers that fall above 2500 dollars.

```
boxplot(Debt$amount, horizontal = TRUE, outline = FALSE, axes = FALSE, staplewex = 1)
text(x = boxplot.stats(Debt$amount)$stats, labels = boxplot.stats(Debt$amount)$stats, y=1.25, cex = 0.6)
```



Generating a boxplot without the outliers as shown above lets us know that the bulk of the data is between 0 and 2500 dollars. Therefore, we can replot the histogram with these new bounds.

```
filteredAmount <- Debt[(Debt$amount < 2500),]
hist(filteredAmount$amount, breaks = seq(0, 2500, by = 100) , main = "Histogram of the Amount of Dollars
     xlab = "Debt Amount per Line Item ($)", las = 2)
axis(1, at = seq(0, 2500, by = 100), las = 2)
```



In this new histogram, we have a clearer view of the distribution of the bulk of our sample. Most notably, we see that most of our line items are reporting a debt between 200 and 300 dollars. We will revisit this point in the bivariate analysis to see what candidates are spending money on that costs between 200 and 300 dollars.

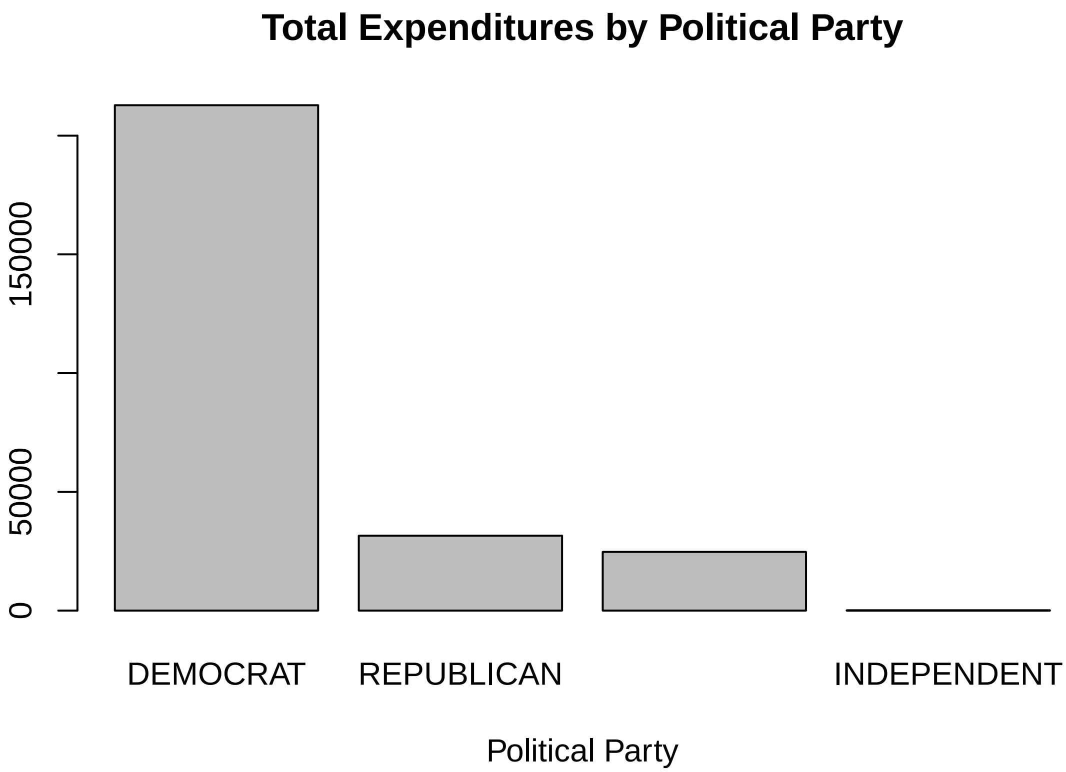
Analysis of Key Relationships

How much money is each party spending?

The first bivariate relationship explored was the total expenditure by political party. The `Debt` dataset was grouped by party, and the sum of the dollar amounts for each group was computed. Figure 8 shows a bar chart with this information. Expenditures in the dataset were overwhelmingly from Democrats, accounting for 79% of the total dollar amount.

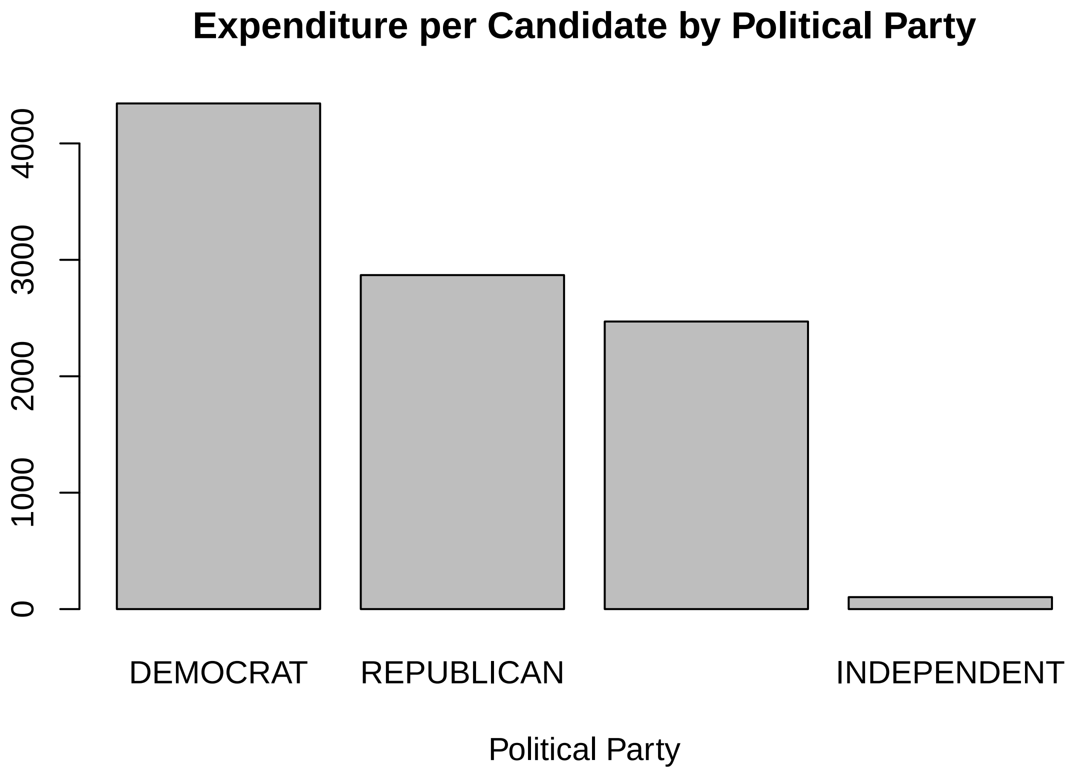
```
# sum amount by party
amountbyparty <-aggregate(Debt['amount'], by=list(party=Debt$party), FUN=sum)

# Rename
names(amountbyparty)[names(amountbyparty)=="amount"] <- "totalamountbyparty"


#format method, which is necessary for formating in a data.frame
format.money  <- function(x, ...) {
  paste0("$", formatC(as.numeric(x), format="f", digits=2, big.mark=","))
}
class(amountbyparty$totalamountbyparty) <- c("money",class(amountbyparty$totalamountbyparty))


# sort the list descending, create pareto
amountbyparty <- amountbyparty[order(amountbyparty$totalamountbyparty, decreasing=TRUE),]
```

```r
# bar plot tutorial https://www.statmethods.net/graphs/bar.html
barplot(amountbyparty$totalamountbyparty, main="Total Expenditures by Political Party",
    xlab="Political Party",
    names.arg=amountbyparty$party,
    cex.names=1)
```



The univariate analysis of number of candidates by party revealed that the number of candidates per party was heavily skewed toward Democrats, so it is reasonable that more total money was spent by Democrat candidates than by those of other parties. The amount of money spent per candidate by party can be computed by dividing the amount of money spent by party by the number of candidates of each party. This is shown in Figure 9.

```r
# Merge PartyCount into amountbyparty
amountbyparty <- merge(amountbyparty, PartyCount)
amountbyparty$amountpercandidate <- amountbyparty$totalamountbyparty / amountbyparty$N

# sort the list descending, create pareto
amountbyparty <- amountbyparty[order(amountbyparty$amountpercandidate, decreasing=TRUE),]


# bar plot tutorial https://www.statmethods.net/graphs/bar.html
barplot(amountbyparty$amountpercandidate, main="Expenditure per Candidate by Political Party",
    xlab="Political Party",
    names.arg=amountbyparty$party,
    cex.names=1)
```



Interestingly, the amounts of money spent per candidate by party have the same relative order as the total amounts spent by party, though the differences in magnitude are greatly reduced.

Both Figures 8 and 9 hint that Washington state politics are dominated by Democrats.

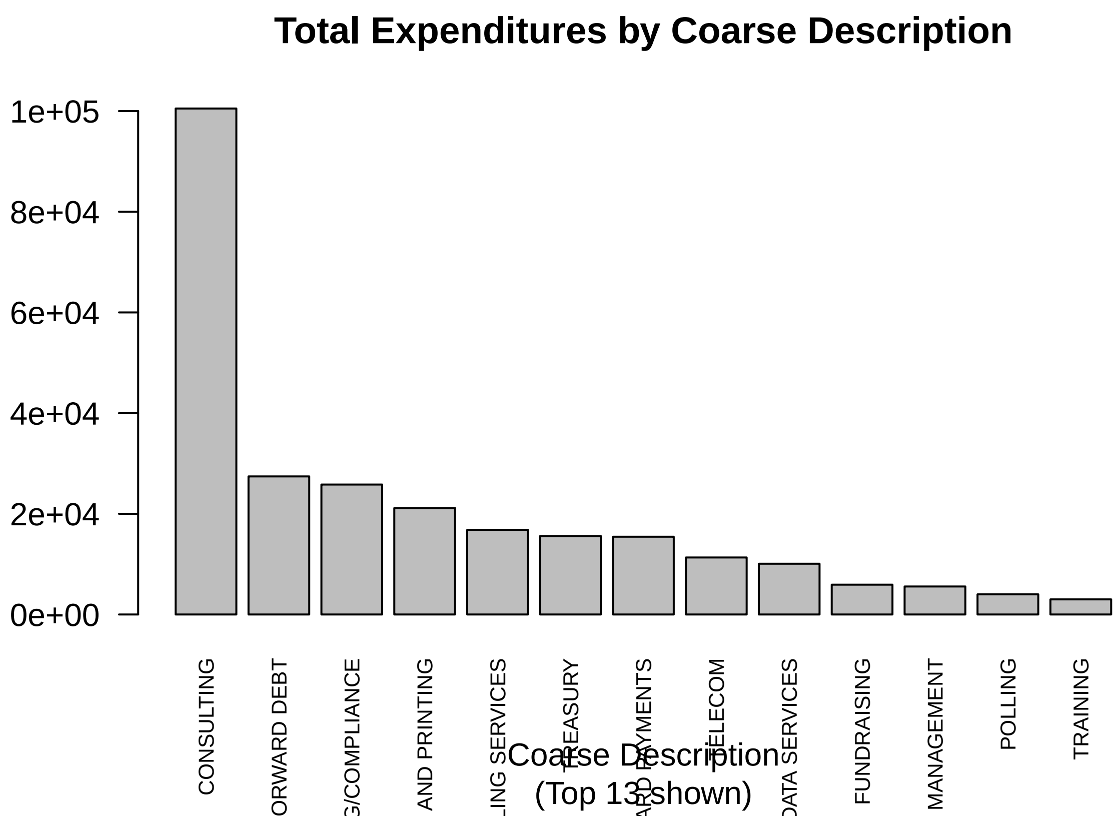
Where is the money going?

The next relationship explored was expenditure by expense category. The dataset's `description` column provided some categorization of expenses, but since this field was populated by different people for different reports, there was not much consistency of values. Out of 194 records, there were 105 unique values in the `description` column. Another column in the dataset, `code`, was supposed to provide a more generalized categorization of expenses. However, this field was populated in only 94 of 194 records and only encompassed 20% of the grand total of all expenses in the dataset ($55,326.09 out of $269,191.70).

Since the `description` data were too granular, and since the `code` data were too sparse, a lookup table was created containing a mpping from `description` to a new column, `coarsedescription`. This table was created by manually examining the unique values in the original `description` column and assigning more general, coarse labels to them. For example, there were 23 `description` values of the form "TREASURY" + month. These were all assigned into `coarsecategory` "TREASURY". Similarly, there were several `description` values that indicated they were related to consulting, so these were assigned a `coarsedescription` of "CONSULTING". In some cases where the nature of an expense was not immediately clear, a quick bit of web searching for the type of business in the `vendorname` column helped to determine a reasonable value for `coarsedescription`. In those cases, an explanation of the reasoning for selecting the value of `coarsedescription` is provided in a `comments` column. There is the possibility that some `description` values did not get mapped to an optimal `coarsedescription` value, but the author is reasonably confident that the mapping is sensible. The 106 unique `description` values were mapped to 23 `coarsedescription` values. For a complete listing of the translation from `description` to `coarsedescription`, see Appendix A.

A pareto of total expenditure by `coarsedescription` is shown in Figure 10. The total for a given `coarsedescription` is computed simply by summing the values in the `amount` column for all records having that `coarsedescription`. The pareto includes only those `coarsedescription` values whose sum of `amount` values exceeds 1% of grand total expenditures, so 13 of the 23 `coarsedescription` values are shown.

```
desc_to_coarsedesc = read.csv("../dataset/description_to_coarsedescription.csv", na.strings='#N/A')
# we need to merge coarsedescription into Debt by description
Debt2=merge(Debt,desc_to_coarsedesc[c('description','coarsedescription')])

# Put 0 for position when there is no value
Debt2$position[is.na(Debt2$position)] <- 0
Debt2$legislativedistrict[is.na(Debt2$legislativedistrict)] <- 0

# sum amount by coarsedescription
amountbycoarsedesc <-aggregate(Debt2['amount'], by=list(coarsedescription=Debt2$coarsedescription), FUN=

#format method, which is necessary for formating in a data.frame
format.money  <- function(x, ...) {
  paste0("$", formatC(as.numeric(x), format="f", digits=2, big.mark=","))
}
class(amountbycoarsedesc$amount) <- c("money",class(amountbycoarsedesc$amount))


# sort the list descending, create pareto
amountbycoarsedesc <- amountbycoarsedesc[order(amountbycoarsedesc$amount, decreasing=TRUE),]
#print(amountbycoarsedesc)

nbars=13
topn=amountbycoarsedesc[1:nbars,]
# bar plot tutorial https://www.statmethods.net/graphs/bar.html
barplot(topn$amount, main="Total Expenditures by Coarse Description",
        sub=paste(c("(Top", nbars, "shown)"), collapse = " "),
    xlab="Coarse Description",
    names.arg=topn$coarsedescription,
    cex.names=0.65, las=2)
```




```
# Maybe I should make a function for bar charts, since I will be making many of them.
```

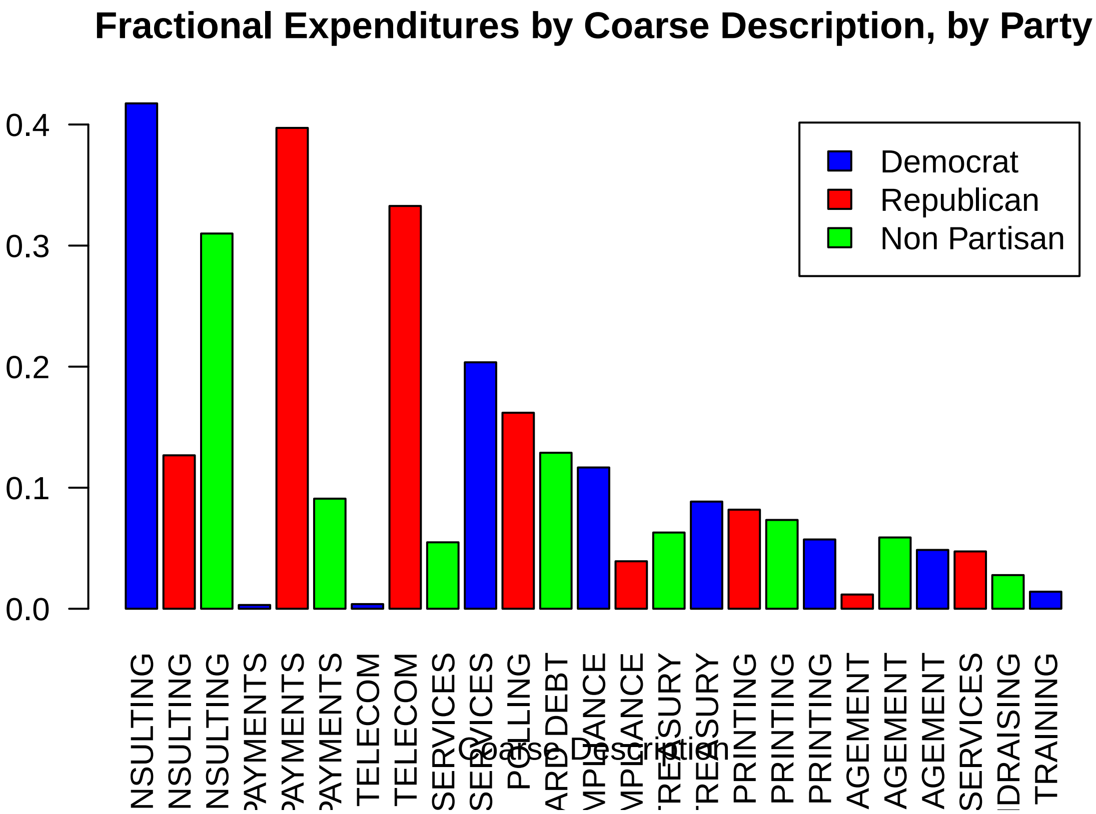
Do the parties spend their money differently?

The expenditures by coarse description can be broken down by political party to see differences in how Republicans and Democrats allocate their expenses. Since the total expenditures were so different between the parties, rather than look at the raw dollar amounts, the data were normalized to fraction of total expenditure for a given party. Since there were only $102.88 in expenditures for Independent candidates (0.04% of total), which was all spent in a single coarse description (Mailing Services), Independent party data were excluded from this specific analysis.

Figure 11 shows a pareto of fraction of total expenditure by coarse description, by party. The same `coarsedescription` used to generate Figure 11 were used. The data reveal that Democrats and Non-partisan candidates spent a greater fraction of money on consulting than the Republicans, while Republicans spent a greater fraction on credit card payments and telecom expenses. Unfortunately, there is no visibility into what purchases were made using the credit cards (the raw `description` column just named the credit card (e.g., AMEX, VISA), so it provides no additional information, either.)

```
# Rename "subject" column to "N"
names(amountbyparty)[names(amountbyparty)=="amount"] <- "totalamountbyparty"

#rollup debt2 by coarsedesc and party
expense_by_coarsedesc_by_party <-aggregate(Debt2['amount'],                              by=


#merge totalamount into expense_by_coarsedesc_by_party
frac_by_coarsedesc_by_party <- merge(expense_by_coarsedesc_by_party,amountbyparty)
# compute fraction
```

```r
frac_by_coarsedesc_by_party$fractionofpartytotal <- frac_by_coarsedesc_by_party$amount / frac_by_coarsed

# drop independents
frac_by_coarsedesc_by_party <- frac_by_coarsedesc_by_party[frac_by_coarsedesc_by_party$party != "INDEPEN

# get max frac by coarsedescription (across parties)
maxfrac_by_coarsedesc <-aggregate(frac_by_coarsedesc_by_party['fractionofpartytotal'],
                                  by=list(coarsedescription=frac_by_coarsedesc_by_party$coarsed

# Rename "fractionofpartytotal" column to "maxfracofpartytotal"
names(maxfrac_by_coarsedesc)[names(maxfrac_by_coarsedesc)=="fractionofpartytotal"] <- "maxfracofpartytot

#merge maxfrac into frac_by_coarsedesc_by_party
frac_by_coarsedesc_by_party2=merge(frac_by_coarsedesc_by_party,maxfrac_by_coarsedesc)


# assign part number to put nonpartisan last
frac_by_coarsedesc_by_party2$partynumber<-ifelse(frac_by_coarsedesc_by_party2$party=="DEMOCRAT",1,
        ifelse(frac_by_coarsedesc_by_party2$party=="REPUBLICAN",2,3
        ))

#Now, sort the df by maxfracofpartytotal desc, then by partynumber
frac_by_coarsedesc_by_party2 <- frac_by_coarsedesc_by_party2[order(-frac_by_coarsedesc_by_party2$maxfrac
                                                                    frac_by_coarsedesc_by_party2$partynum

# Now we can finally make a grouped pareto!
frac_by_coarsedesc_by_party2$barlabels <- ""
frac_by_coarsedesc_by_party2$barlabels <- frac_by_coarsedesc_by_party2$barlabels <-


# Trick Make labels which are non blank only when row mod 3 =2
x<-1:nrow(frac_by_coarsedesc_by_party2)
y <- x%%3==2

cd=frac_by_coarsedesc_by_party2$coarsedescription
frac_by_coarsedesc_by_party2$barlabels <- with(frac_by_coarsedesc_by_party2, ifelse(y,cd,"" ) )

cd <- frac_by_coarsedesc_by_party2$coarsedescription
frac_by_coarsedesc_by_party2$barlabels <- ifelse(y,frac_by_coarsedesc_by_party2$coarsedescription,"" )

# get the same coarsedescriptions used in previous graph
topnparty<-frac_by_coarsedesc_by_party2[frac_by_coarsedesc_by_party2$coarsedescription %in% topn$coarsed

# now make a sidebyside bar chart
barplot(topnparty$fractionofpartytotal, main="Fractional Expenditures by Coarse Description, by Party",
        xlab="Coarse Description", col=c("blue","red","green"),
        legend = c("Democrat","Republican","Non Partisan"), beside=TRUE, names.arg=topnparty$coarsedescr
```



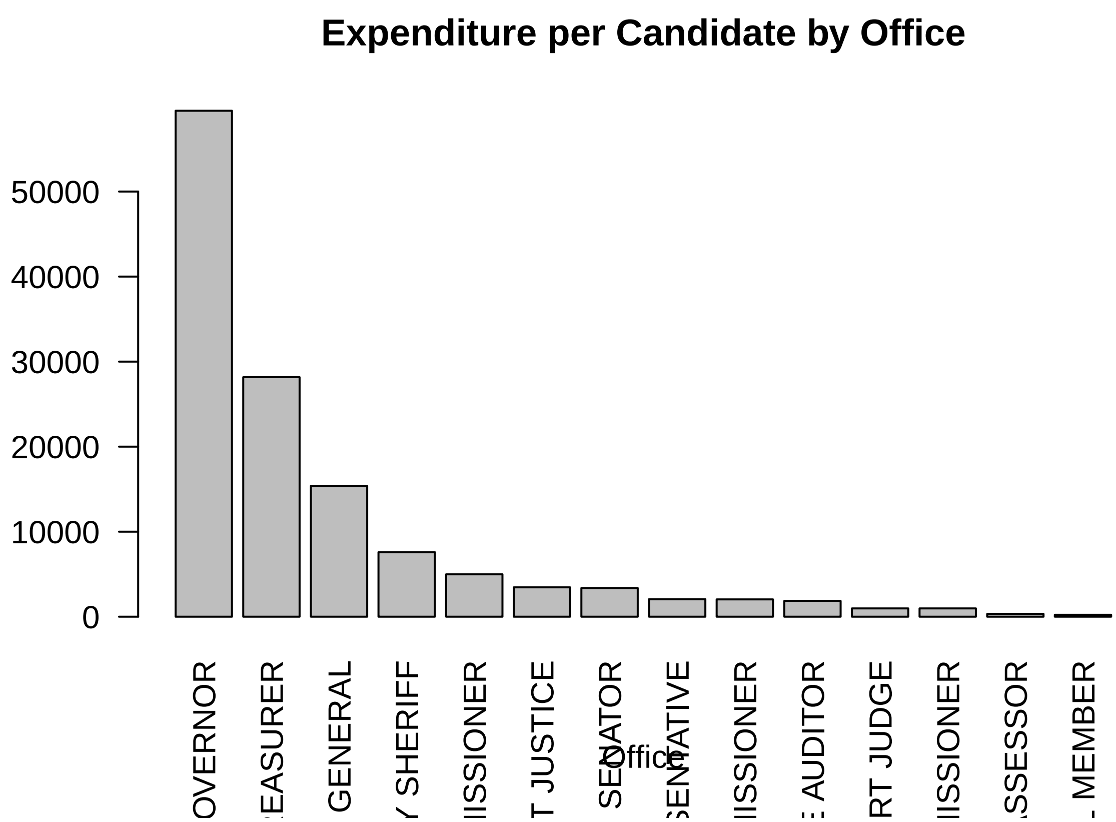
For Which Types of Offices is the Most Money Spent?

An analysis of amount of expenditure by office was performed by summing the amounts spent by unique value of the office2 column (the office column from the right half of the dataset). However, since the number of offices of a given type varied (there are more state representatives than state senators or governors), and since the number of candidates varied by office type, the amounts were divided by the number of unique combinations of `legislativedistrict`, `position`, `party`, and `jurisdiction` within each `office` value. The intent of this was to normalize the amounts to a "per campaign" value. For example, there were 39 unique candidates for state representative. In Washington, there are two state representatives per legislative district, and the `position` column designates which of these two slots a row belongs to. The `party` column provides another level of granularity for a given line item. A pareto of expenditures per campaign by office is shown in Figure 12. Note that since the dataset does not have a unique candidate identifier, such as a name or registration ID, multiple candidates of the same party with expenditures for the same office and position would be treated as a single candidate.

The pareto shows that the race for governor had the greatest expenditure per candidate. Two other statewide offices, treasurer and attorney general, showed the next highest expenditures per candidate. This seems reasonable, since candidates for these offices must campaign in the entire state, vs. within a single legislative district.

```
# get sum of amount by DUMMY
#sumbyoffice <- aggregate(Debt2['amount'],
#by=list(office=Debt2$DUMMY), FUN=sum)

# get sum(amount) and n rows by unique value of DUMMY, legislativedistrict, position, and party,
# jurisdiction, and jurisdictiontype (in case I summarize by that later).
sumbycand <- aggregate(Debt2['amount'],
by=list(office=Debt2$office, legislativedistrict=Debt2$legislativedistrict, position=Debt2$position, pa
```

```r
# now sum the amounts in sumbycand and count rows by office
sumbyoffice <- aggregate(sumbycand['amount'], by=list(office=sumbycand$office), FUN=sum)
countbyoffice <- aggregate(sumbycand['position'], by=list(office=sumbycand$office), FUN=length)

# Rename cols
names(sumbyoffice)[names(sumbyoffice)=="amount"] <- "totalamount"
names(countbyoffice)[names(countbyoffice)=="position"] <- "ncandidates"

# merge the sum and count data
amountperjob <- merge(sumbyoffice,countbyoffice)
amountperjob$mean_amount <- amountperjob$totalamount / amountperjob$ncandidates

# sort by mean_amount desc
# sort the list descending, create pareto
amountperjob <- amountperjob[order(amountperjob$mean_amount, decreasing=TRUE),]

# bar plot tutorial https://www.statmethods.net/graphs/bar.html
barplot(amountperjob$mean_amount, main="Expenditure per Candidate by Office",
    xlab="Office",
    names.arg=amountperjob$office,
    cex.names=1, las=2)
```

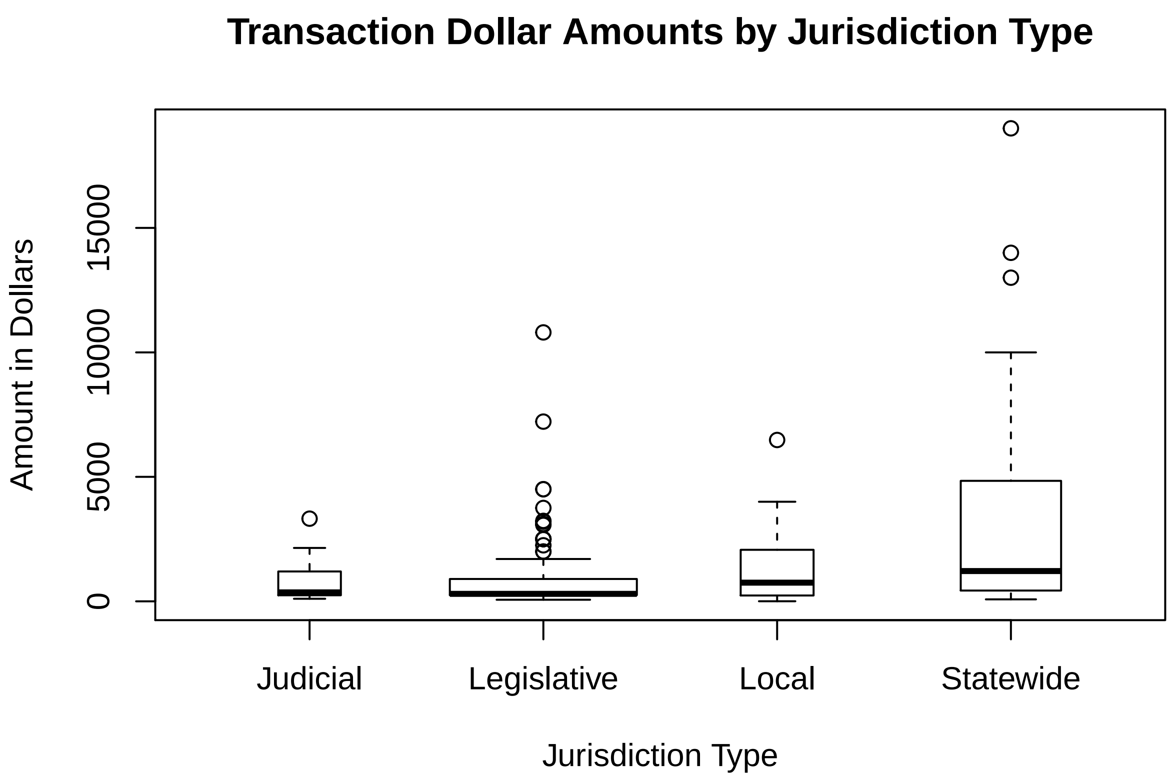


Do Different Jurisdiction Types Have Different Transaction Sizes?

Campaigns for offices of different jurisdiction types are likely conducted differently. The dataset contains four unique values in its jurisdictiontypes column–judicial, legislative, local, and statewide. Box plots of transaction amounts by jurisdictiontypes are shown in Figure 13. To provide insight into the relative number of data points comprising each jurisdition type, the width of the boxes are drawn proportionally to the square root of the number of points. The plot shows that while there were several outlier points for legislative offices, the highest amounts were in the outliers in the statewide offices. The three outlier expenditure amounts for statewide offices were all for consulting expenses, whereas the outliers for legislative offices spanned many categories, including mailing services, graphic design and printing, telecom, consulting, and web and data services. The single outlier for judicial offices was also consulting. The single outlier expense for local offices was for a credit card payment (to Cabela's, a hunting, fishing, and camping supply store, oddly enough).

The small dataset size of 194 total points, and the disparate number of points by jurisdiction type (Judicial: 14; Legislative: 125; Local: 19; and Statewide: 36) may limit the insight obtained by the box plots, but examination of the outlier points give some indication that consulting fees are some of the most expensive transactions in political campaigns for most types of political offices in Washington.

```r
# Expenditure amounts by jurisdictiontype
boxplot(amount~jurisdictiontype,data=Debt2, main="Transaction Dollar Amounts by Jurisdiction Type",
    xlab="Jurisdiction Type", ylab="Amount in Dollars", varwidth=T)
```



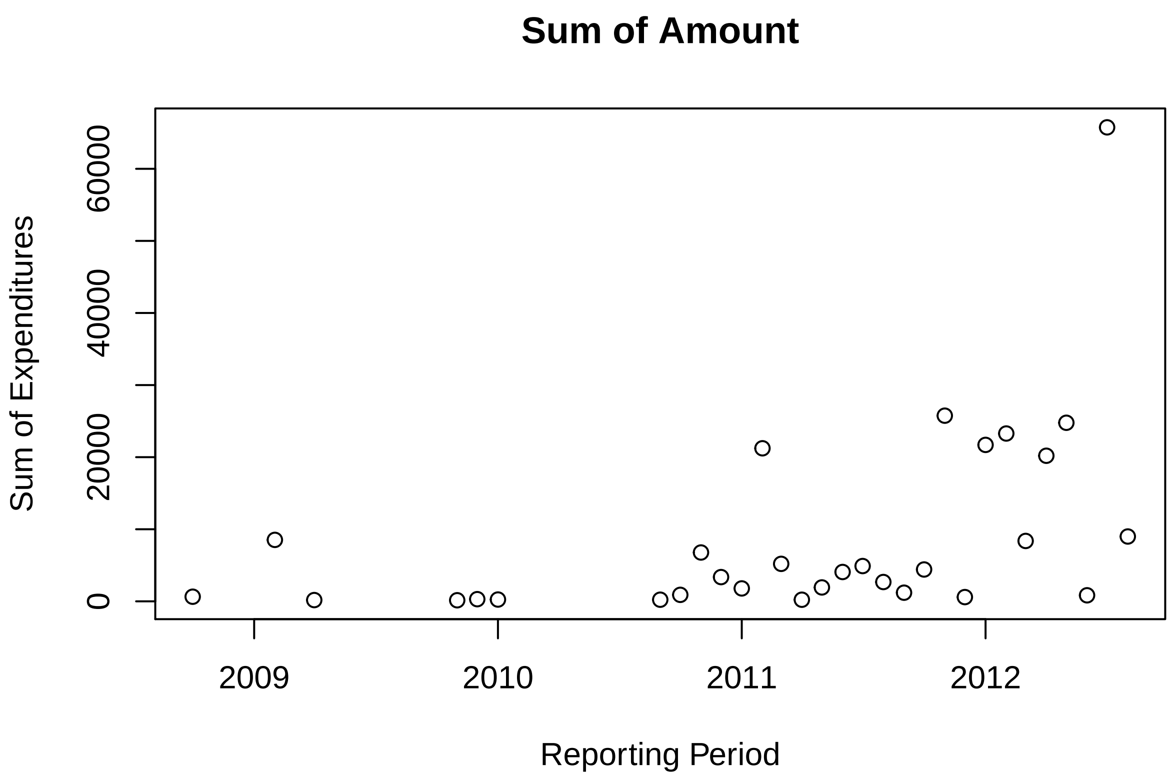
Do Expenditures Change with Time?

As a final investigation, the change in spending amounts over time was examined. The data were grouped by the month in which the expenditure occurred, based upon the value of `debtdate`. Trend plots of the sum, count, and average expenditure amount vs. thrudate are shown in Figures 15-17, respectively. The trends show that both the number of expenditures and the dollar amount of individual expenditures generally increased in the months leading up to the election.

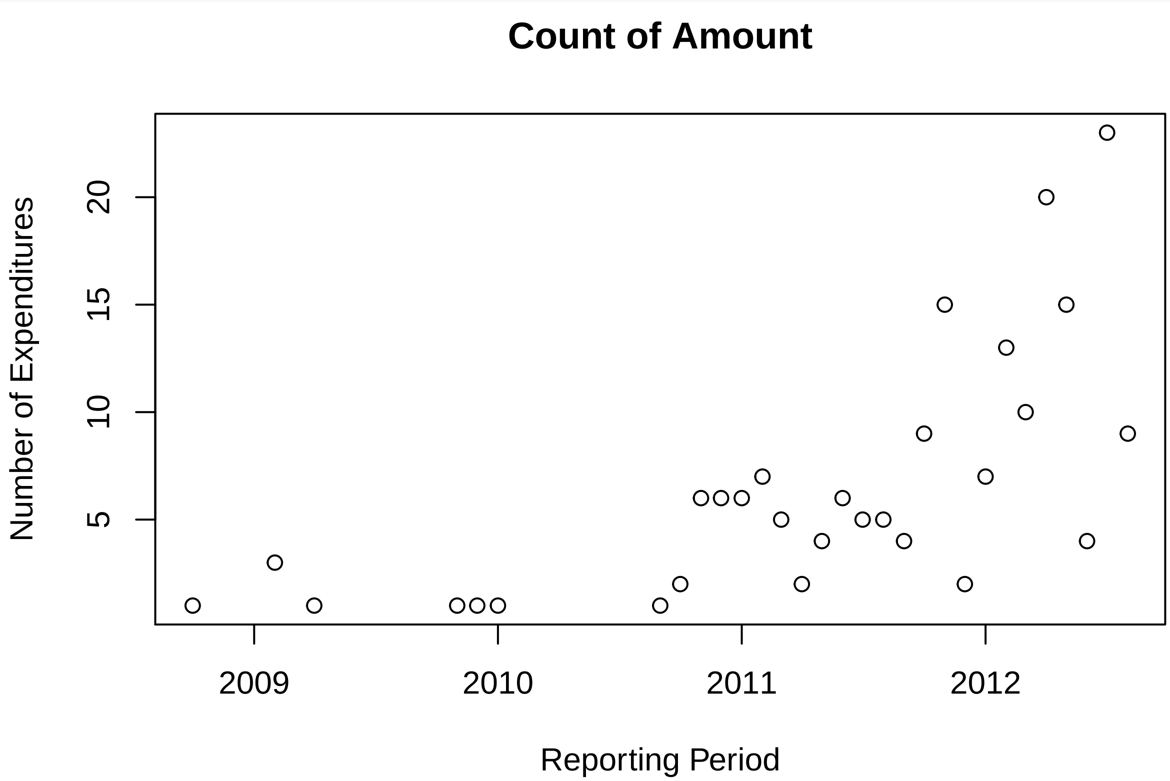
```r
# Time component
# Copy Debt2 to Debt3
Debt3 <- Debt2
# Let's make a debtmonth column
library(lubridate)
# Get just the month
Debt3$truedebtdate<-mdy(Debt3$debtdate)
Debt3$debtmonth<-floor_date(Debt3$truedebtdate, unit="month")
# Make str version so it will be a factor
Debt3$debtmonthstr<- as.character(Debt3$debtmonth)
# Create dummy variable, TREASURY
Debt3$Treasury<-ifelse(Debt3$coarsedescription=="TREASURY",1,0)


##need plyr package to use ddply
library(plyr)
```

```
##
## Attaching package: 'plyr'

## The following object is masked from 'package:lubridate':
##
##     here
```

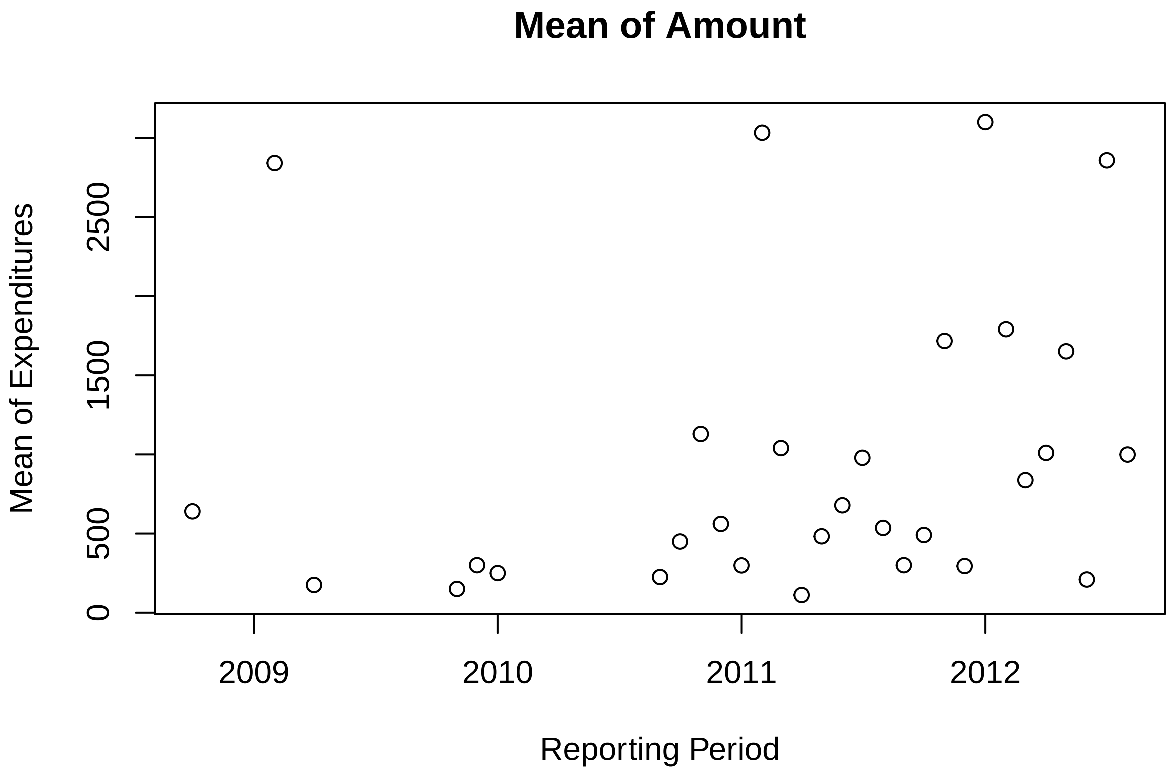
```r
# Now get sum, count, and mean of amount and mean of Treasury by debtmonthstr
##summarize data. Because R categorized Day as a factor we can go ahead and use this
timestats<- ddply(Debt3,~debtmonthstr,summarise,Sum_amount=sum(amount), Count_amount = length(amount),
      Mean_amount = mean(amount), Mean_Treasury = mean(Treasury))


#Create debtmonth in timestats, as a date (this works but format is not good for graphing)
timestats$debtmonth<- strptime(timestats$debtmonthstr,"%Y-%m-%d")

# format it to MM/YYYY
# I believe this sets it back to char!
#timestats$debtmonth<- format(timestats$debtmonth, format="%Y/%m")


# sort by it
timestats<- timestats[order(timestats$debtmonth),]


xvector=timestats$debtmonth
yvector=timestats$Sum_amount
yvector2=timestats$Count_amount
yvector3=timestats$Mean_amount
plot(xvector, yvector,xlab="Reporting Period",  ylab="Sum of Expenditures", main="Sum of Amount")
```




```
plot(xvector, yvector2,xlab="Reporting Period",  ylab="Number of Expenditures", main="Count of Amount")
```



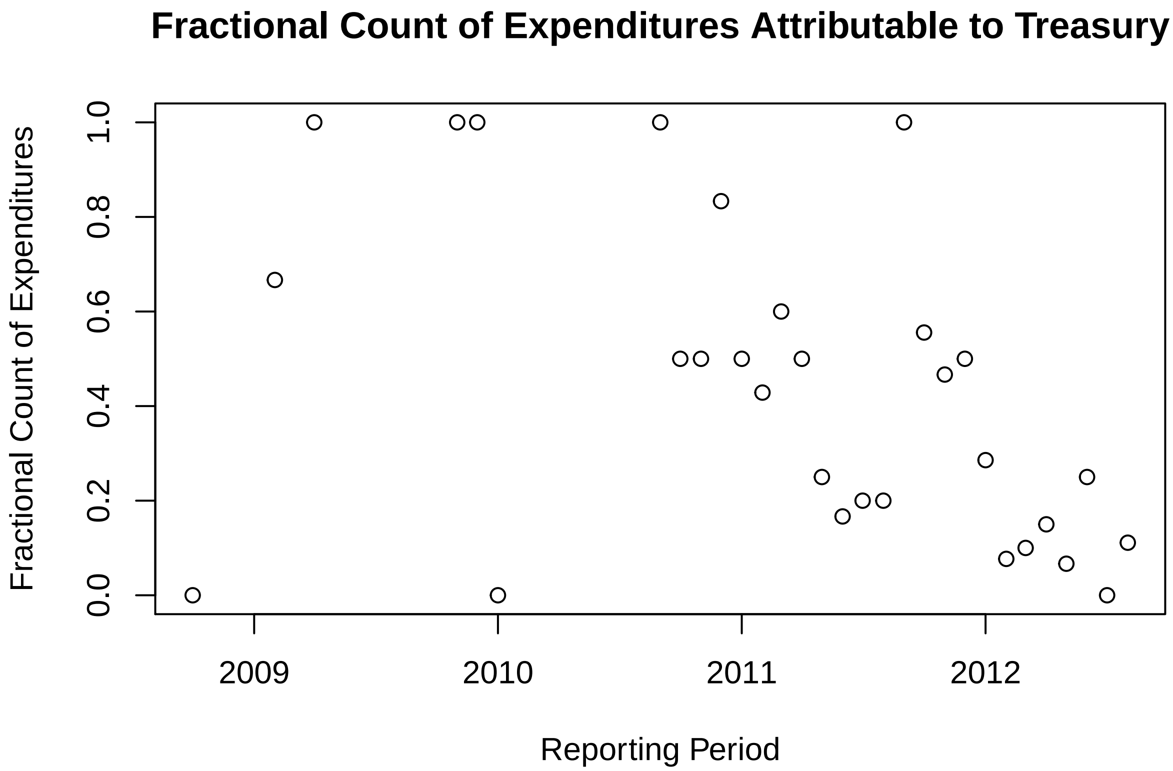

```
plot(xvector, yvector3,xlab="Reporting Period",  ylab="Mean of Expenditures", main="Mean of Amount")
```



Just what are "Treasury" Expenditures?

Figures 15-17 show expenditures attributable to the 2012 election as far back as 2009. These may or may not be erroneous data. Investigation of the descriptions of expenditures in the earliest months indicate that the majority of expenditures through February of 2011 were described as Treasury expenditures. To understand the significance of these expenditures, a dummy variable, `Treasury`, was created which was set to 1 for treasury expenses and 0 for all other categories. The mean of this dummy variable by thrudate (the average fractional count of individual expenditures in a month attributable to Treasury) was plotted vs. `debtmonth` in Figure 18. The plot shows that after February of 2011, the fractional count of expenditures with this coarse description decreases significantly, as other expense categories become more numerous. There are 57 records of Treasury expenses in the dataset, of which 54 have the value "Operations and Overhead" in the `code` column. The average amount of treasury expenses is $278. Without further domain knowledge, one cannot know the exact meaning of these expenses, but perhaps these are monthly fees paid for the services of each campaign's treasurer which are incurred even during periods of relative inactivity in the campaign.

```
# plot treasury mean by debtmonth
xvector=timestats$debtmonth
yvector=timestats$Mean_Treasury
plot(xvector, yvector, xlab="Reporting Period",  ylab="Fractional Count of Expenditures", main="Fractio
```



Analysis of Secondary Effects

We now extend a few of the bivariate relationships discussed above with a secondary variable that deepens our understanding of the primary relationship.

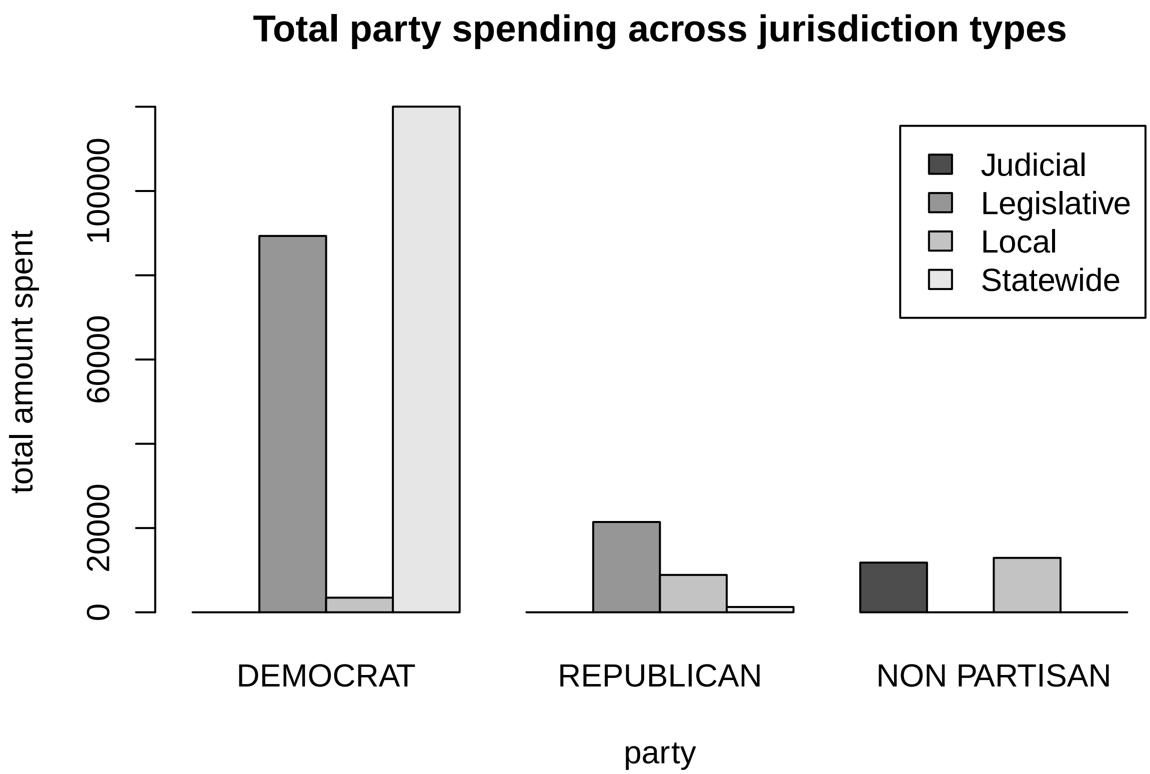
What jurisdiction types do the parties spend their money on?

We saw above that candidates from the Democratic party far outspend the candidates from all of the other parties. When we examine the secondary effect of the jurisdiction type (`jurisdictiontype`) on this relationship, the story is even more interesting. Note: We again omit the Independent candidates from this summary due to their overall low spending amount.

To examine the spending priorities of the parties by jurisdiction type, a bar plot of debt by jurisdiction type by party was generated, with a group of bars for each party and a single bar within each group for a single jurisdiction type. Grouping the data in this way gives insight into each party's relative allocation of expenditures by jurisdiction type, and it allows comparison of how each party prioritizes different jurisdiction types. Figure 14 shows Democrats spend the most money on statewide campaigns, followed closely by legislative campaigns, while Republicans spend most of their money on legislative campaigns, followed by local campaigns. Judicial offices and many local offices are non-partisan, which is why non-partisan candidate spending was highest for these categories.

```
party_jurtype_sum <- aggregate(Debt['amount'],
                               by=list(party=Debt$party,
                                       jurisdictiontype=Debt$jurisdictiontype),
                               FUN=sum)

# Order by total spending
xtab_sum <- xtabs(amount ~ jurisdictiontype + party, data=party_jurtype_sum)
```

```r
xtab_sum <- xtab_sum[,order(colSums(xtab_sum), decreasing = TRUE)]

# Remove INDEPENDENT
xtab_sum <- xtab_sum[,colnames(xtab_sum) != 'INDEPENDENT']

xtab_sum_mat <- data.matrix(xtab_sum)
barplot(xtab_sum_mat,
        beside = TRUE,
        legend.text = rownames(xtab_sum_mat),
        main = "Total party spending across jurisdiction types",
        xlab = "party",
        ylab = "total amount spent")
```



Now we can see that although the Democratic party spends more money than the other parties overall, they spend nearly all of it on legislative and statewide elections. The Republican party, on the other hand, focuses their resources on legislative and local elections and spends almost nothing on statewide elections. This may suggest that Washington state, as a whole, is sufficiently Democratic that the Republican party does not attempt to compete for statewide offices, and instead focuses its resources on competing for geographically localized offices in more conservative districts.

Conclusion

Debt amount is highly skewed in the positive direction.

Statewide amounts have the largest skew of the jurisdiction types

Total expendatures for democrats far exceed those for all other parties combined

Conclusion…