Prediction on Canadian City Polls

Wenya Cai

2023-04-09

Overall Goal

With the comprehensive understanding of the poll data (reference https://open.toronto.ca/dataset/polls-conducted-by-the-city/), we can help stakeholders make informed decisions that better reflect the needs and the opinions of the community. Also, we are able to predict the "in favor" rate by identifying trends and patterns to help policy makers generate polls and shift focus on more important topics.

Data Description

Background Knowledge

Some background about the poll data:
The City Clerk's Office is responsible for conducting polls on behalf of City divisions. Polls are conducted to establish the opinions of residents and businesses on various topics covered by a City by-law (Chp190) .

Poll engagement is the primary data provided in this set. Daily data updates occur upon the closing and certification of each poll. The collection of data starts from April 1, 2015.

Data Evaluation

No response rate table

APPLICATION_FOR	Total	No_Responce_Rate
Front Yard Parking	449	0.03
Traffic Calming	220	0.55
Appeal - Front Yard Parking	199	0.03
Permit Parking	131	0.13
Boulevard Cafe	113	0.35
Commercial Boulevard Parking	16	0.44
Traffic Calming Safety Zone	9	0.11
Proposed Business Improvement Area	8	0.38
Traffic Calming – Island	6	0.33
Business Improvement Area	5	0.20

From the above table, we can see that people tends to care more about front yard parking where front yard parking results in the largest total number of observations. Meanwhile, Traffic Calming is the second application that people will focus on, and Appeal - Front Yard Parking is the third highest.

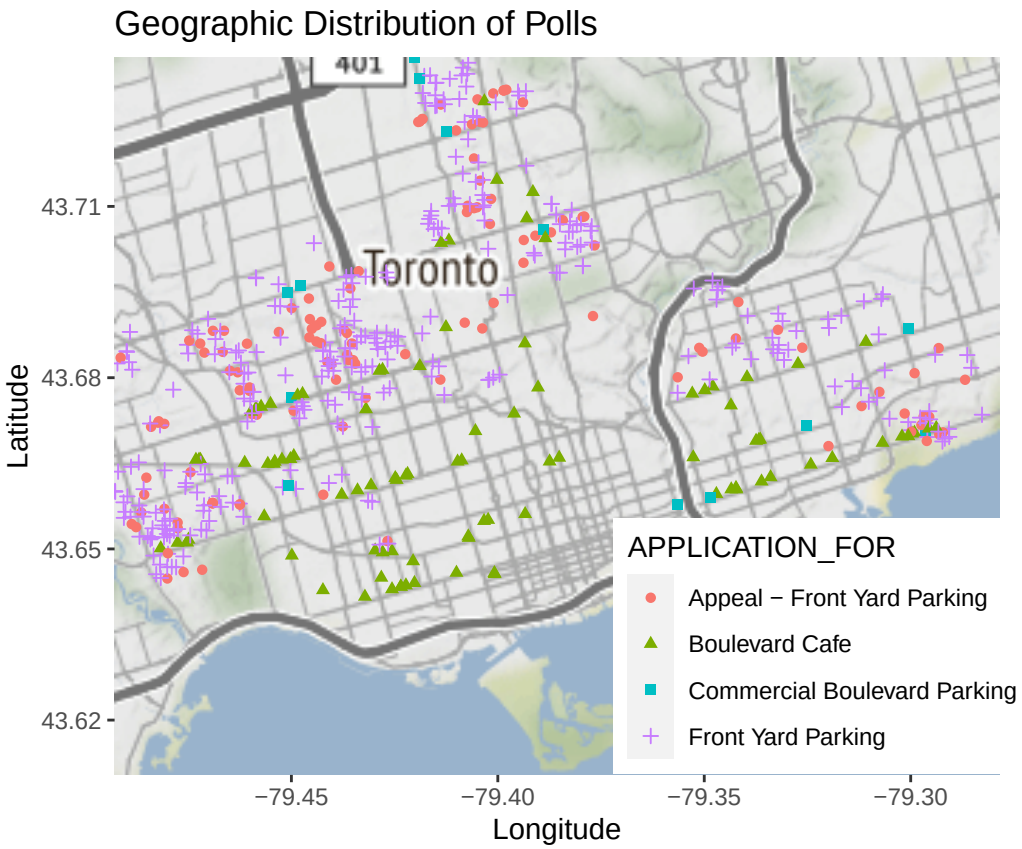
What's more, as for the No_Response_Rate, we see that Front Yard Parking has a very low no response rate, of 3%. However, Traffic Calming's performance is not so good as the no_response_rate is about 55%.

In favour Poll rate summary table

APPLICATION_FOR	Total	In_Favour_Rate
Front Yard Parking	437	0.97
Appeal - Front Yard Parking	194	0.85
Permit Parking	114	0.47
Traffic Calming	99	0.72
Boulevard Cafe	73	0.62
Commercial Boulevard Parking	9	0.33
Traffic Calming Safety Zone	8	1.00
Proposed Business Improvement Area	5	1.00
Business Improvement Area	4	1.00
Traffic Calming – Island	4	0.25

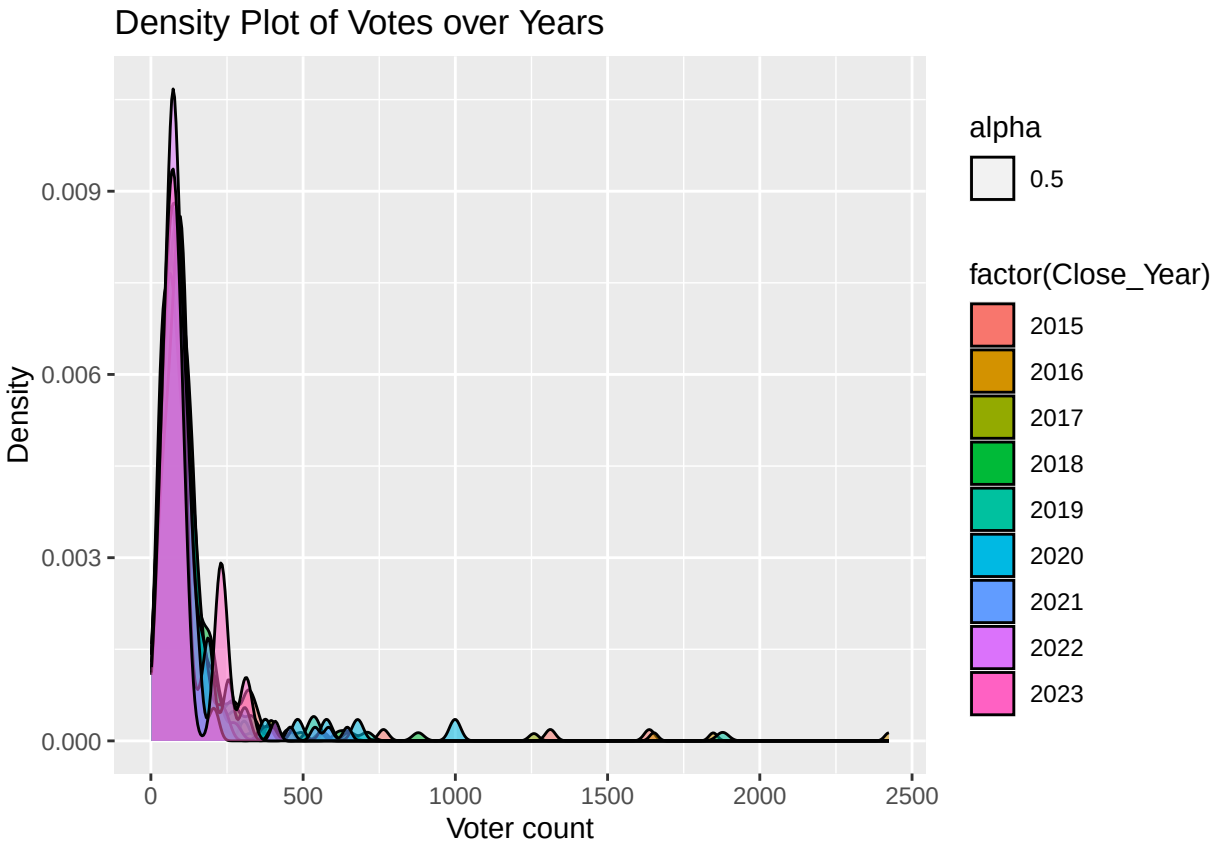
Similarly for no_response_rate, we are interested in the "in favor" poll rate. The above table gives us the results. Here are some findings: The application for Front Yard Parking stills goes the top place, resulting in the highest number of observations with a high in favor rate (97%). Similar to Front Yard Parking, Appeal - Front Yard Parking goes the second place, having total number of 194 and a favor rate of 85%.

Geographic distribution of polls



From the above geographic distribution of the polls, we are able to get a sense of how different applications are distributed around the Toronto area. One interesting thing is, at the middle-south part of Toronto, there is only application for Boulevard Cafe and we don't have any other applications in that area.

Distribution of polls theme in each year



From the density plot of votes over years, we are able to see that most of the voters count for a poll range from 0-400. What's more, the density curve is highly skewed to right.

Average votes per roll

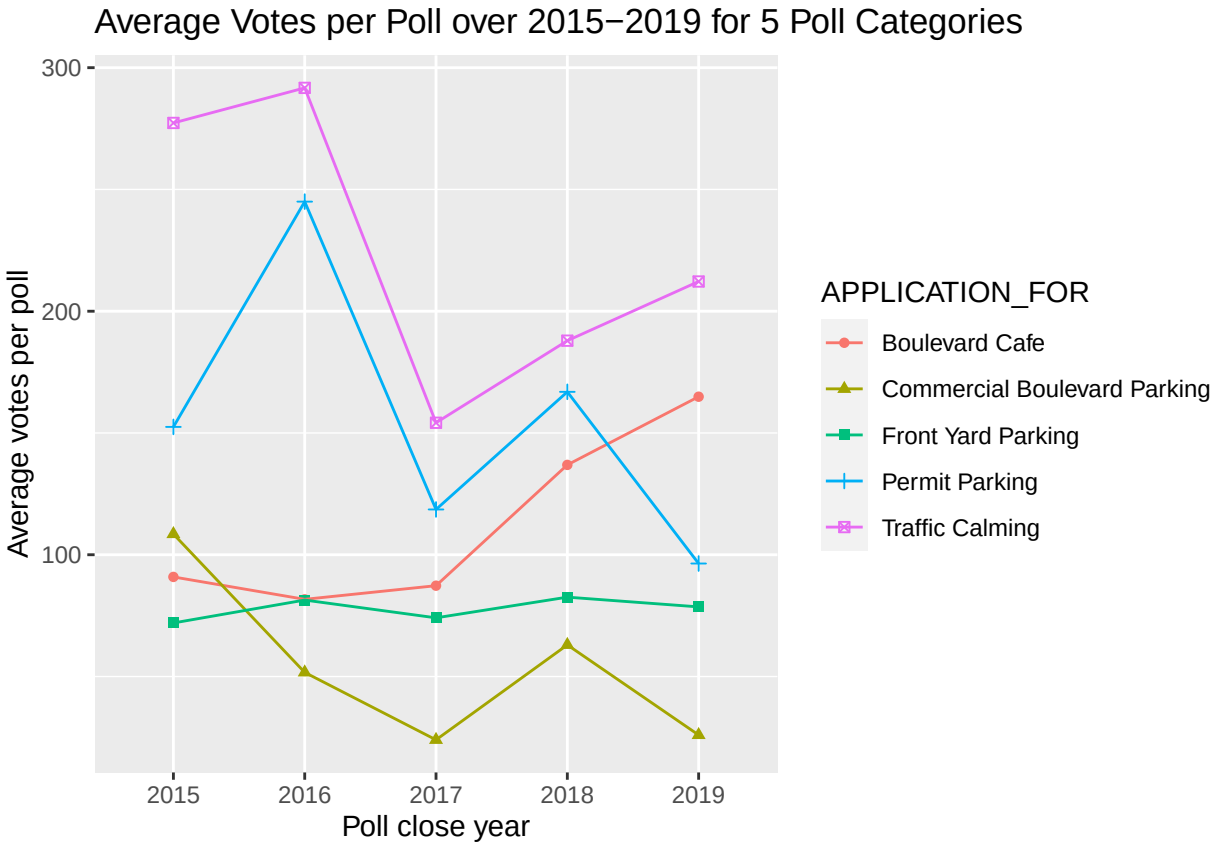
Close_Year	Total_Polls	Total_Votes	Avg_Votes
2016	220	26808	121.9
2017	210	22040	105.0
2018	158	18320	115.9
2019	145	18283	126.1
2015	128	16272	127.1
2021	126	14296	113.5
2022	89	7647	85.9
2020	61	8358	137.0
2023	19	2055	108.2

We will choose to analyze 2015-2019 in later parts since these consecutive years have sufficiently large total votes and total polls, stable average votes compared to what we have after 2019.

Next, let's take a look at what happened in detail during 2015-2019:

First, let's determine what categories actually appeared consecutively from 2015 to 2019:

By now, we have all the poll categories appeared consecutively from 2015 to 2019. Thus, we have everything we need to plot:



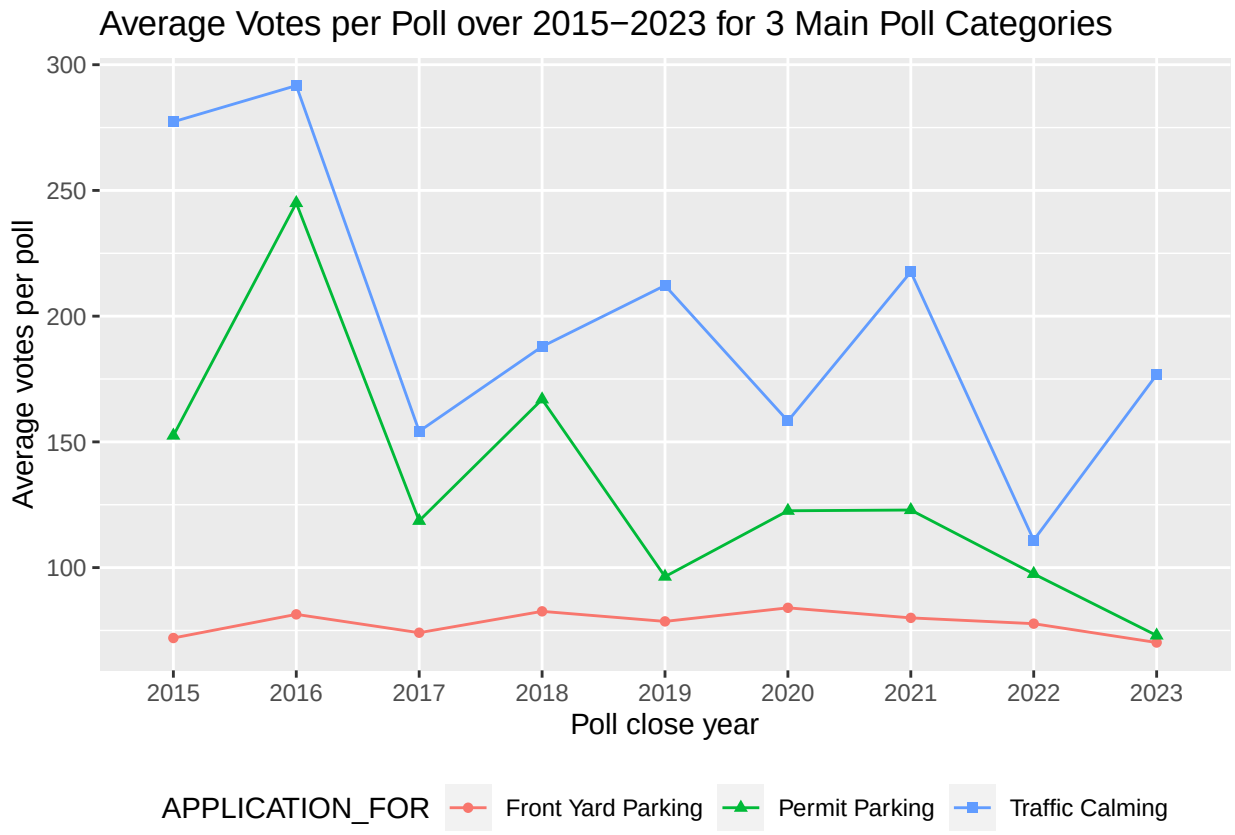
From the graph, we conclude that:

1. all those 5 application categories appeared every consecutive years during 2015-2019.
2. Traffic Calming and Permit Parking seems to have fluctuating average votes per poll within 2015-2019, while Boulevard Cafe has an upward trending among those years, Commercial Boulevard Parking has a downward trending among those years and Front Yard Parking seems to be stable at all years.

Besides that, we can also take a look at those who appeared in all years from 2015-2023. To do that, we can do similar procedures:

```
## [1] "Front Yard Parking" "Traffic Calming"    "Permit Parking"
```

Here, we see that this time we only have 3 categories which appearcd every consecutive years from 2015-2023. So we are just using those 3 categories to draw the graph:



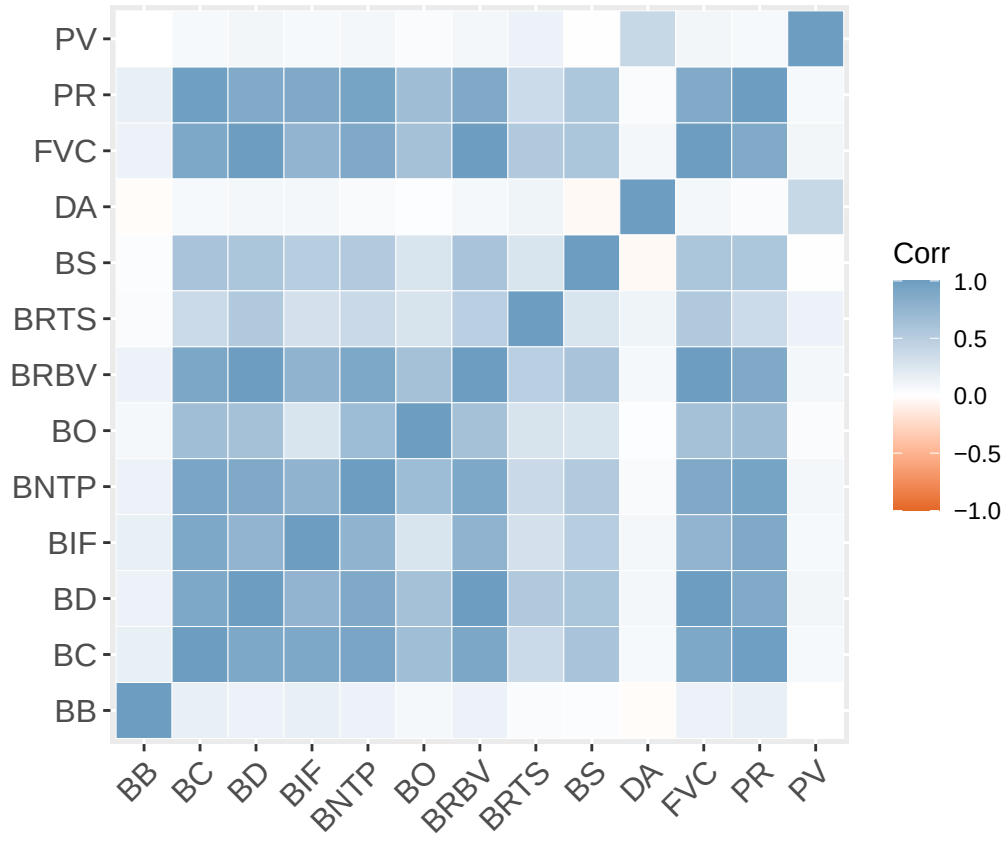
The results are quite similar:

1. Among 2015-2023, Traffic Calming and Permit Parking seems to have a downward trending with regards to average votes per poll with small fluctuations between years.
2. Front Yard Parking still has a stable average votes per poll every year. However, the number of votes per poll is relatively limited compared to other categories.

We begin the next part of the report with the above background knowledge that emerged from the poll responses. Through these analysis, we hope to gain a deeper understanding of the needs of Toronto residents and how they differ across different regions of the city. Finally, we will discuss the implication of these findings for Toronto residents and policy makers.

Multiple Statistical Analysis

PCA




```
##            Comp.1        Comp.2      Comp.3       Comp.4      Comp.5
## BB     0.05127036  0.7158537774  0.05893432  0.079589536  0.34183464
## BC    -0.35259699  0.0005054838  0.18746051  0.048283783  0.04024678
## BD    -0.33984345 -0.0913921275 -0.09837451 -0.015656983  0.11176109
## BIF   -0.30566230  0.0104624042  0.37102869 -0.288127638  0.33284652
## BNTP  -0.34277321 -0.0341896725  0.11229933  0.161764531  0.04140445
## BO    -0.24178594 -0.0365043653 -0.18885123  0.725572704 -0.28327563
## BRBV  -0.34636749 -0.0749422702 -0.02710302  0.007515836  0.08370451
## BRTS  -0.11123798 -0.2118639282 -0.76342250 -0.239737603  0.33844667
## BS    -0.23889174  0.0251004144  0.04757710 -0.530496303 -0.60684780
## DA     0.17297375 -0.4688884322  0.29368485  0.077353529  0.40328733
## FVC   -0.33984345 -0.0913921275 -0.09837451 -0.015656983  0.11176109
## PR    -0.35335163  0.0098292766  0.19219889  0.082215179  0.03057006
## PV     0.16877601 -0.4440173784  0.21699939  0.059186267 -0.10202910
```

By PCA, we gained statistical significance among all features, and for later analysis, we gonna choose top five features to do a logistic regression which explains best in variations among all data. They are:

1. comp1: PASS_RATE
2. comp2: POTENTIAL_VOTERS
3. comp3: BALLOTS_RETURNED_TO_SENDER

4. comp4: BALLOTS_OPPOSED
5. comp5: BALLOTS_SPOILED

Next, we will use those 5 top features to construct a linear logistic regression and a non-linear logistic regression and compare their R^2 to determine which model might actually performs better than the other.

Fit a linear logistic regression and calculate its R^2

```
## [1] 0.8727206
```

By using the top 5 features that we determined by PCA, we are able to fit a linear logistic regression model with many useful information from the summary table. Besides that, the R^2 in the summary table tells us how good is this model. In this case, R^2 = 0.8727206 = 87.27%, which represents that 87.27% of the variation in the data can be explained by this linear regression model, which is pretty good.

Similarly, we can fit a non-linear logistic regression and calculate its corresponding R^2.

Fit a non-linear logistic regression and calculate its R^2

```
## [1] 0.8759611
```

From the model given above together with the summary table and the R^2 we want, we can see that in the non-linear logistic regression model, $R^24 is slightly larger than what we have in the linear regression model, which means this data set fits better in the non-linear case.

By having those two different models in hand, we are interested in another way of looking at the performance of different models. Next, we will perform cross validation for each model and compare which model gives a lower MSE since a lower MSE represents the model fits better and have smaller errors.

Cross Validation for linear model

We will conduct a cross validation using randomly 80% of the data as the training group and 20% of them to be the test data. What's more, we will use the same linear model gained from the PCA above as well. Also, we are going to see what is the value of MSE:

```
## [1] 0.039801
```

From the above results, we see that if we use linear logistic regression model to do cross validation, we will result in a MSE of 0.039801.

Cross Validation for nonlinear model

Similarly, we will perform the same cross validation algorithm by using the nonlinear logistic regression model and see the value of MSE this time.

```
## [1] 0.05319149
```

Doing similar procedures, but this time we are using the non-linear logistic regression model to do cross validation. The MSE result turns out to be 0.0532. Comparing to what we got in the previous part, we see that time we have a larger MSE.

To sum up, MSE of linear model is smaller than nonlinear model, but the R^2 of the nonlinear is larger. Therefore we can conclude that there may exist overfitting in nonlinear model.

Last but not least, we will focusing on constructing a 95% confidence interval to see whether the poll is more likely to be "in favor" or not. This time, we will not only do the single 95% confidence interval to illustrate that idea, but also conduct a 95% bootstrap confidence interval. Hopefully, these two methods will receive the same result.

95% confidence interval of poll result is "In Favour" (population proportion)

```
##
##  1-sample proportions test with continuity correction
##
## data:  sum(s) out of length(s), null probability 0.5
## X-squared = 0.3, df = 1, p-value = 0.5839
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.3766139 0.7402456
## sample estimates:
##         p
## 0.5666667
```

Based on 95% CI for population proportion, we see that confidence interval starts from 0.3766129 to 0.7402456. Since our null hypothesis is p = 0.5, and it falls in the 95% CI, thus we have statistically confidence to say that we fail to reject the null hypothesis.

Lastly, let's perform a 95% bootstrap CI of poll results "In Favour".

95% prediction bootstrap CI of poll result is "In Favour"

```
##      2.5%     97.5%
## 0.5666667 0.8666667
```

Prediction confidence Interval

```
##  2.5% 97.5%
##   0.6   0.9
```

```
##      2.5%     97.5%
## 0.5000000 0.8333333
```

_____CI for linear model is closer to the real CI_____

Based on the 2.5% and 97.5% quantile result, we are able to conclude that our null hypothesis falls in the middle 95% region, which indicates that we fail to reject the null hypothesis as well.

However, one thing to note here is that if we compare the CI with the previous one, we could see that the CI using bootstrap is wider than the normal CI, mainly because we are simulating each trial instead of using actual data.

Final Summary

Since our goal is to identify significant characteristics of the poll data and help stakeholders make informed decisions that could represent the needs for the whole community. By looking at poll data itself, we construct the "no response rate" table and "in favor rate" table plot. Here are some findings: people tends to care more about front yard parking where front yard parking results in the largest total number of observations. Meanwhile, Traffic Calming is the second. Similar results hold for "in favor rate" table plot. Thus, we conclude that people are more interested in the topic of front yard parking rather than other applications.

Besides that, from the density plot of votes over years, we are able to see that most of the voters count for a poll range from 0-400. Also, statistics about total polls, total votes, and average rolls for each year, we could have the following findings: 2016 received most polls; 2020 received most votes on average. To see this in a whole, we will mainly choose to analyze 2015-2019 in later parts since these consecutive years have sufficiently large total votes and total polls, stable average votes compared to what we have after 2019. However, for completeness, we also attach those statistics for all years. The result is also consistent to what we have before: Front Yard Parking still has a stable average votes per poll every year.

Next, for detailed analysis regarding to this poll data, we try to find the best model which can best predict the "in favor" rate. Before fitting a model, we first determine the top 5 features that influence the prediction result the most. Therefore, we use Principle Component Analysis (PCA) to find those features. Next, we fit the linear logistic regression model as well as a non-linear logistic regression model and test their performance. To test the performance, we mainly use two criteria: R^2 and the MSE given by the cross validation method. The result turns out to be: MSE of linear model is smaller than nonlinear model, but the R^2 of the nonlinear is larger. Therefore we can conclude that there may exist overfitting in nonlinear model.

Last but not least, we constructed three 95% CI for predicting the "in favor" rate. The first CI is based on linear logistic regression, the second one is based on non-linear logistic regression, and the last one is according to real data. Based on our findings, we can conclude that the linear model is more closer to the true CI, which represents that liner model is preferred and appreciated among the choices. This linear logistic regression model can be used by researchers and experts to predict the future "in favor" rate as well as other further analysis.

Appendix

```
# install and run all required library
library(tidyverse)
library(ggmap)
library(osmdata)
library(ggplot2)
library(knitr)
library(dplyr, warn.conflicts = FALSE)
library(tidygeocoder)
library(corrr)
library(ggcorrplot)
```

Data Description

Background Knowledge

```
# load the data
data=read.csv("Polls Data.csv")
```

Data Evaluation

No response rate table

```
#No response rate table
d1=data
d1$NoRes_Count=0
d1$NoRes_Count[d1$POLL_RESULT=="Response Rate Not Met"]=1
No_Res_table=d1 %>% select(APPLICATION_FOR,NoRes_Count) %>%
  group_by(APPLICATION_FOR) %>%
  summarise(Total=n(),No_Responce_Rate=round(mean(NoRes_Count),2)) %>%
  arrange(desc(Total))
kable(No_Res_table)
```

In favour Poll rate summary table

```
#only count valid polls
d = data %>% filter(POLL_RESULT != "Response Rate Not Met")
d$InFavour_Count=0
d$InFavour_Count[d$POLL_RESULT=="In Favour"]=1
```

```
#In favour Poll rate summary table
Poll_rate_table = d %>%
  select(APPLICATION_FOR,InFavour_Count) %>%
  group_by(APPLICATION_FOR) %>%
  summarise(Total=n(),In_Favour_Rate=round(mean(InFavour_Count),2)) %>%
  arrange(desc(Total))
kable(Poll_rate_table)
```

Geographic distribution of polls

```
#filter out blank address
addresses=data%>%select(APPLICATION_FOR,ADDRESS)
addresses = addresses %>% filter(ADDRESS != "")
```

```
#get latitude and longtitude
lat_longs <- addresses %>%
  geocode(ADDRESS, method = 'osm', lat = latitude , long = longitude)
#reference:https://cran.r-project.org/web/packages/tidygeocoder/readme/README.html
```

```
#plot
p = get_map(location = getbb("toronto"),zoom=11,source = "stamen")
```

```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
map <- ggmap(p)
df=lat_longs%>%filter(!is.na(longitude))
map +
  geom_point(df, mapping=aes(x=longitude,y=latitude,
                             shape=APPLICATION_FOR,color=APPLICATION_FOR)) +
  ggtitle("Geographic Distribution of Polls") +
  xlab("Longitude") + ylab("Latitude") +
  theme(legend.position = c(0.8,0.18)) +
  labs(fill="Application For")
```

Distribution of polls theme in each year

```
#density plot
d2 = data %>%
  mutate(Close_Year=format(as.Date(CLOSE_DATE),format="%Y"))
g <- ggplot(d2,aes(FINAL_VOTER_COUNT))
g + geom_density(aes(fill=factor(Close_Year),alpha=0.5)) +
  ggtitle("Density Plot of Votes over Years") +
  xlab("Voter count") + ylab("Density")
```

Average votes per roll

```
#polls count table
Polls_Count_table=d2 %>%
  select(APPLICATION_FOR,FINAL_VOTER_COUNT,Close_Year) %>%
  group_by(Close_Year) %>%
  summarise(Total_Polls=n(),Total_Votes=sum(FINAL_VOTER_COUNT),
            Avg_Votes=round(mean(FINAL_VOTER_COUNT),1)) %>%
  arrange(desc(Total_Polls))
kable(Polls_Count_table)
```

```
#preprocess the data
d3 = d2 %>% filter(Close_Year %in% c(2015:2019))
t = d3 %>%
  select(APPLICATION_FOR,FINAL_VOTER_COUNT,Close_Year) %>%
  group_by(Close_Year,APPLICATION_FOR) %>%
  summarise(Total_Polls=n(),
            Total_Votes=sum(FINAL_VOTER_COUNT),
            Avg_Votes=round(mean(FINAL_VOTER_COUNT),1))
```

```
#checking appearance
app_for = d3 %>% distinct(APPLICATION_FOR)
app_for = app_for[['APPLICATION_FOR']]
checker=rep(0,5)
names=c()
for (i in 1:9) {
  for (j in 2015:2019) {
```

```
    temp=t%>%filter(Close_Year==j)
    if(app_for[i] %in% temp$APPLICATION_FOR){
      checker[j-2014]=1
    }
  }
  if(identical(checker,rep(1,5))){
    names[i]=app_for[i]
  }
  checker=rep(0,5)
}
names=names[!is.na(names)]
```

```
#avg votes plot
t1 = t %>% filter(APPLICATION_FOR %in% names)
ggplot(t1,
       aes(x=Close_Year, y=Avg_Votes,group=APPLICATION_FOR,
           shape=APPLICATION_FOR,color=APPLICATION_FOR)) +
  geom_point() + geom_line() +
  ggtitle("Average Votes per Poll over 2015-2019 for 5 Poll Categories") +
  xlab("Poll close year") +
  ylab("Average votes per poll")
```

```
#preprocess the data
t2=d2 %>%
  select(APPLICATION_FOR,FINAL_VOTER_COUNT,Close_Year) %>%
  group_by(Close_Year,APPLICATION_FOR) %>%
  summarise(Total_Polls=n(),
            Total_Votes=sum(FINAL_VOTER_COUNT),
            Avg_Votes=round(mean(FINAL_VOTER_COUNT),1))
```

```
#checking appearance
app_for2 = d2 %>% distinct(APPLICATION_FOR)
app_for2 = app_for2[['APPLICATION_FOR']]
checker2=rep(0,9)
names2=c()
for (i in 1:10) {
  for (j in 2015:2023) {
    temp=t2%>%filter(Close_Year==j)
    if(app_for2[i] %in% temp$APPLICATION_FOR){
      checker2[j-2014]=1
    }
  }
  if(identical(checker2,rep(1,9))){
    names2[i]=app_for2[i]
  }
  checker2=rep(0,9)
}
names2=names2[!is.na(names2)]
```

```
t3=t2 %>% filter(APPLICATION_FOR %in% names2)
ggplot(t3,
       aes(x=Close_Year, y=Avg_Votes, group=APPLICATION_FOR,
           shape=APPLICATION_FOR,color=APPLICATION_FOR)) +
```

```
  geom_point() + geom_line() +
  ggtitle("Average Votes per Poll over 2015-2023 for 3 Main Poll Categories") +
  xlab("Poll close year") +
  ylab("Average votes per poll") +
  theme(legend.position = "bottom")
```

Multiple Statistical Analysis

PCA

```
glmdata = data %>% filter(POLL_RESULT != "Response Rate Not Met")%>%
  mutate(result = case_when(POLL_RESULT == "In Favour"~1,
                            TRUE~0), .after = POLL_RESULT)
glmdataNumeric = select_if(glmdata, is.numeric) %>%
  select(!starts_with(c("X_id", "result", "POLL_ID")))

# normalization of data
data_normalized <- scale(glmdataNumeric)
colnames(data_normalized) = c("BB", "BC", "BD", "BIF", "BNTP", "BO",
                              "BRBV", "BRTS", "BS", "DA", "FVC", "PR", "PV")

corr_matrix <- cor(data_normalized)
ggcorrplot(corr_matrix,
           colors = c("#E46726", "white", "#6D9EC1"),
           outline.color = "white",
           ggtheme = ggplot2::theme_gray)
```

```
#PCA
data.pca <- princomp(corr_matrix)
comps = data.pca$loadings[, 1:5]
comps
```

Fit a linear logistic regression and calculate its R^2

```
#Linear
glmModel = glm(result~ PASS_RATE+POTENTIAL_VOTERS+BALLOTS_RETURNED_TO_SENDER+
                BALLOTS_OPPOSED+BALLOTS_SPOILED+
                 APPLICATION_FOR, family = binomial, data = glmdata)
#R square
with(summary(glmModel), 1 - deviance/null.deviance)
```

Fit a non-linear logistic regression and calculate its R^2

```
#Nonlinear
glmModel1 = glm(result~ PASS_RATE+POTENTIAL_VOTERS*BALLOTS_RETURNED_TO_SENDER+
                BALLOTS_OPPOSED*BALLOTS_SPOILED+
                 APPLICATION_FOR, family = binomial, data = glmdata)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#R square
with(summary(glmModel1), 1 - deviance/null.deviance)
```

Cross Validation for linear model

```
set.seed(3)
d=glmdata %>% mutate(group_ind = sample(c("train","test"),
size=nrow(glmdata),
prob = c(0.8,0.2),
replace = T))

glmModelVa = glm(result~ PASS_RATE+POTENTIAL_VOTERS+BALLOTS_RETURNED_TO_SENDER+
                BALLOTS_OPPOSED+BALLOTS_SPOILED+
                 APPLICATION_FOR, family = binomial, data = d%>% filter(group_ind=="train"))

y.hat = predict(glmModelVa, newdata = d %>% filter(group_ind=="test"))
y.pass =y.hat>=0.5
y.hat[y.pass] = 1
y.hat[!y.pass] = 0
mean((d$result[d$group_ind=="test"] - y.hat)^2)
```

Cross Validation for nonlinear model

```
d=glmdata %>% mutate(group_ind = sample(c("train","test"),
size=nrow(glmdata),
prob = c(0.8,0.2),
replace = T))

glmModelVa1 = glm(result~ PASS_RATE+POTENTIAL_VOTERS*BALLOTS_RETURNED_TO_SENDER+
                BALLOTS_OPPOSED*BALLOTS_SPOILED+
                 APPLICATION_FOR, family = binomial, data = d%>% filter(group_ind=="train"))

y.hat = predict(glmModelVa1, newdata = d %>% filter(group_ind=="test"))
y.pass =y.hat>=0.5
y.hat[y.pass] = 1
y.hat[!y.pass] = 0
mean((d$result[d$group_ind=="test"] - y.hat)^2)
```

95% confidence interval of poll result is “In Favour” (population proportion)

```
Favour_result = as.numeric(data$POLL_RESULT == "In Favour")
set.seed(0)
s = sample(Favour_result, size = 30)

prop.test( x = sum(s), n = length(s) )
```

95% bootstrap CI of poll result is "In Favour"

```
set.seed(2)
obs.sample = sample(Favour_result, size = 30)

boot_function=function(){
  boot_s = sample(obs.sample, size=30, replace=TRUE)
  return(mean(boot_s))
}

boot_X_bar = replicate(10000,boot_function())
quantile(boot_X_bar, c(0.025,0.975))
```

Prediction confidence Interval

```
set.seed(0)
obs.sample = sample(y.hat, size = 30)

boot_function=function(){
  boot_s = sample(obs.sample, size=30, replace=TRUE)
  return(mean(boot_s))
}

boot_X_bar = replicate(10000,boot_function())
quantile(boot_X_bar, c(0.025,0.975))
```

```
set.seed(0)
obs.sample = sample(y.hat1, size = 30)

boot_function=function(){
  boot_s = sample(obs.sample, size=30, replace=TRUE)
  return(mean(boot_s))
}

boot_X_bar = replicate(10000,boot_function())
quantile(boot_X_bar, c(0.025,0.975))
```