Building microtargeting strategy

through predicting Walz's approval rating

Introduction

Based off the 2020 elections, we sought to predict the value of future voters to determine the recipients of the Tim Walz campaign mailers. The data used in the project consists of eight characteristics of respondents, along with answers to three questions asked by 1,000 people since the 2020 U.S. general elections. We were asked to focus on and decided to analyze the characteristics of the respondents who answered the approval of Government Tim Walz's performance. In order to determine Tim Walz's performance, voters chose from the following five responses: strongly approve, strongly disapprove, approve, and neither approve nor disapprove. We considered three analytical methods that all predict categorical response values: the ordinal logistic regression model, the decision tree, and the random forest. The biggest difference between the three methods was that the ordinal logistic regression was the ordinal classifier and that the decision tree and random forest were the non-linear classifier. Additionally, due to the difference in the features of each method, the accuracy and interpretation resulted can differ. We obtained the analysis results, compared their error rates, and determined which method is more accurate. then we utilized the most appropriate method to develop the microtargeting strategy of the campaign. Also, the significant results we obtained are expected to help us strategically identify voters to consider when sending mail. This will help promote Walz's campaign and increase participation in the campaign's new events or speeches.

Methods and Materials

As mentioned earlier, the ordinal logistic regression we used to analyze the data is a regression model that many experts consider. Fullerton said, "The ''traditional'' approach to ordered logit, the ''proportional odds'' model (McCullagh 1980), is the primary tool that many sociologists and other social scientists utilize to examine the determinants of ordinal-level outcomes" (Fullerton, 2009, p.307). The ordinal logistic regression uses logit link based on cumulative probability. $\gamma_{ik}=P(Y_i \leq k)$ is the form of cumulative probabilities where i index the group or the individual and k indexes the categories in that $1 \leq \ldots. \leq K$. The logit link of the proportional odds model is $\log\frac{\gamma_{ik}}{1-\gamma_{ik}}$, and proportional odds model for i and i' is $(\frac{\gamma_{i\prime k}}{1-\gamma_{i\prime k}})/(\frac{\gamma_{i\prime}}{1-\gamma_{i\prime k}})$ which does not depend on k; from above the formula about i and i', denominator represents odds for the i^{th} individual and numerator represents odds for the i^{th} individual. This is why we call this model a proportional-odds model (Algeri, 2021).

When considering the models for categorical response variables within the data of this project, the multinomial logistic regression model was also an option, but the reason for choosing the ordinal logistic regression was because of the great advantage of the ordinal logistic regression. Even though we are essentially not considering the information on the order of the response variable when fitting logit multinomial models, the proportional odds model takes into account the order of the response variable across the categories. Since we need to check the statistical results considering the order of the response variable in this project, we anticipate that the proportional odds model will be able to make more accurate predictions than the multinomial logit model in building the model.

Another method we used to analyze this data was the decision tree, which is a statistical method currently utilized by many data scientists. The method in utilizing this is not

complicated. Yurong Zhong (2016) said that like that of a flow chart, the decision tree uses an "top-down recursive method". Through the predictor values, the "internal nodes" of the flow chart, or tree, are compared and then the branches that branch out from those nodes are then created according to those different predictor values. The result and conclusion can finally be obtained through these "leaf nodes". There are specifically two types of decision trees, one regression tree and one classification tree; the regression tree can be used when response variable is continuous, and the classification tree can be used when response variable is categorical. Hence, I used the classification tree in this project. The most important difference between a classification tree and a regression tree is that the classification tree does not focus on minimizing rss, but on minimizing Gini-index or Cross-entropy index when they grow the tree. It plays important role to split the data using a cut point at each node.

Decision trees have different advantages from the ordinal logistic regression, and these advantages are also attractive when analyzing data. One of the advantages of the decision tree is that it takes complex relationships between input variables and simplifies them, and target variables by taking the original input variables and separating them into significant subgroups (Song and Lu, 2015). Another advantage mentioned by Song and Lu (2015) was that they can be easily understood and interpreted, and missing values can be easily handled. Additionally, this allows a non-parametric approach without distributional assumptions (Song and Lu, 2015). The advantages of the decision tree described by experts above can be checked that this method can efficiently statistically analyze data. However, according to the data I referred to, there are some limits to using the decision tree. One of the disadvantages of the decision tree is that it can be overfitting and underfitting when we use small data. It can limit the final model's generalizability and robustness (Song and Lu, 2015). Another disadvantage is the strong

correlation between input variables; even though the strong correlation helps to enhance prediction accuracy of the model. it may create a high probability in selecting variables that are not concerned of our interest (Song and Lu, 2015).

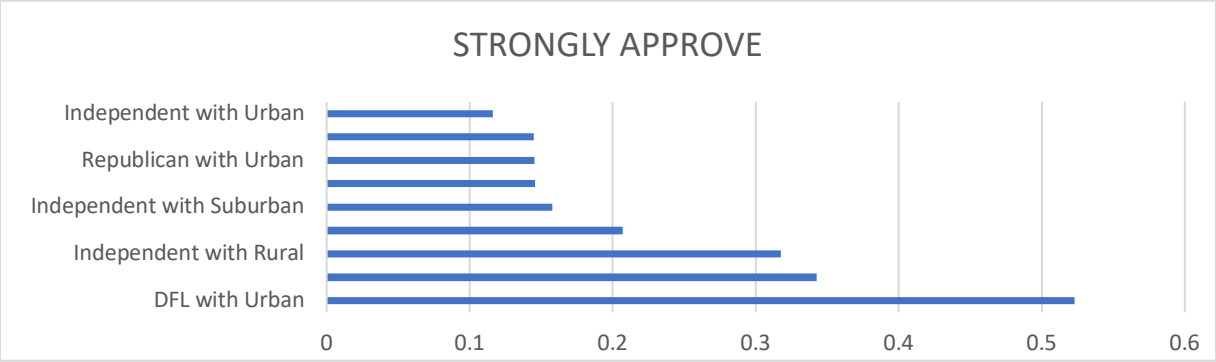
As I mentioned earlier, to overcome the critical drawbacks of the decision tree, we came up with the random forest method. Random forest helps to reduce the overfitting problem and strong correlation; it also helps to improve the accuracy rate. The random forest collects the several trees through boot strap sampling from a training set. The random forest determines the most common prediction from their prediction results as the final prediction result based on several trees. One big difference from the decision tree is that when the random forest grows a tree, all predictors are not used in random forest method; they typically select some variables at random among all predictors. When the random forest selects variables, they usually pick $\sqrt{n}$ or $\frac{n}{3}$ if n is the number of all predictors. This procedure reduced the variance which reduces the risk of overfitting and the correlation which improves the accuracy rate.

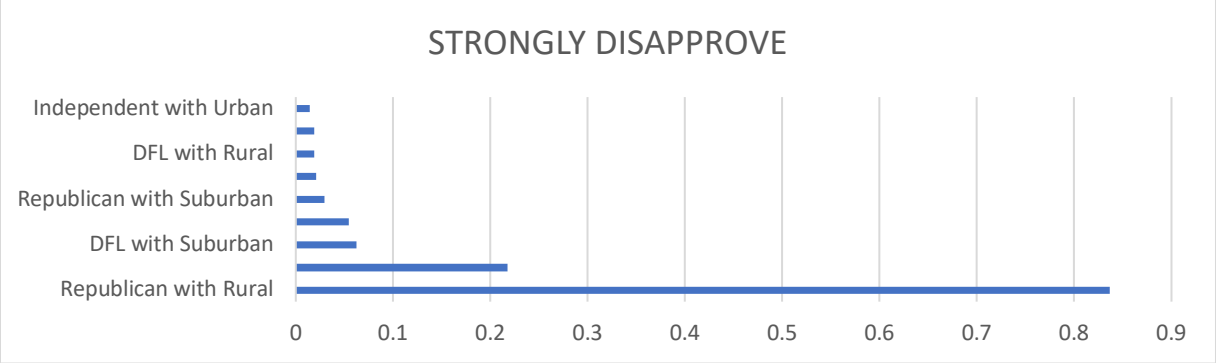
Results

As mentioned earlier, we decided to first create an ordinal logistic regression model and analyze the data. In order to get a prediction error rate, we split the data into a training set and a validation set with a 50% ratio respectively at the beginning of the fitting model, and we fitted the model based on the training data. To find meaningful variables in the predictor variables in the data, we compared the nested models using the F-test. Using the anova function, we compared the reduced model (i.e. approve (Walz's performance) ~ 1) and full model (i.e. approve (Walz's performance) ~.), and we found that the most proper model contained voter_party and voter_urban (Appendix A). Also, we found that the model with the interaction of these two

variables was the best fit. Hence our final model is “approval” (Walz’s performance) ~ voter_party * voter_urban (Appendix B). Information about this final model can be obtained by using the summary function, and we expressed the final model as follows (Appendix C). We checked this in various ways, whether the final model we found is suitable for producing meaningful results. The first way to check was by the Pearson's chi-squared test. (Appendix D). The following results show that the p-value is significant, so it is sufficiently appropriate to produce meaningful results.

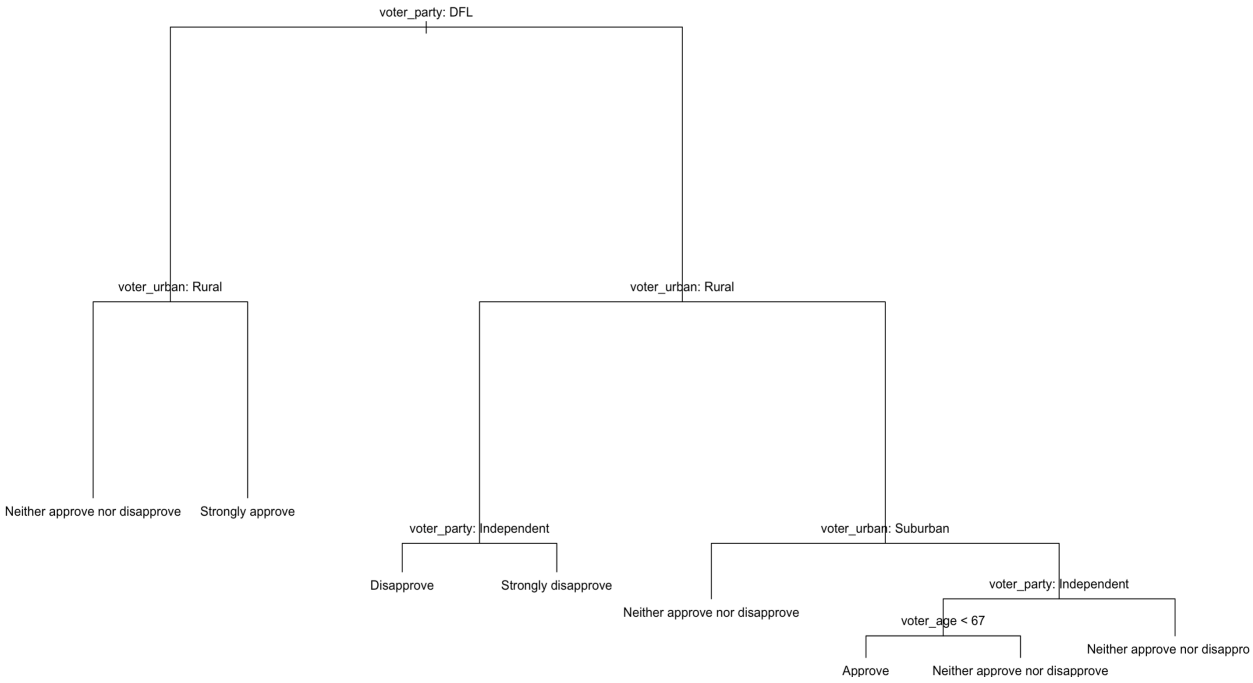
Finally, we were able to predict the probability of what people would rate Tim Walz's performance through this model. Since this is a probability predicted from historical data, we thought that in order to effectively achieve the ultimate goal, we should consider the probability of strongly approve and strongly disapprove among the five possible answers. We obtained the result value by utilizing the final model we set through the predict function and the probability we predicted is as follows.





In the case of the strongly approve probability, people in urban with DFL tendencies strongly approve of Tim Walz's performance with a 52.27% chance; those people had the highest probability to support Tim Walz's performance. In the case of the strongly disapprove probability, people with Republican tendencies in rural strongly opposed Tim Walz's performance with a 83.63% probability.

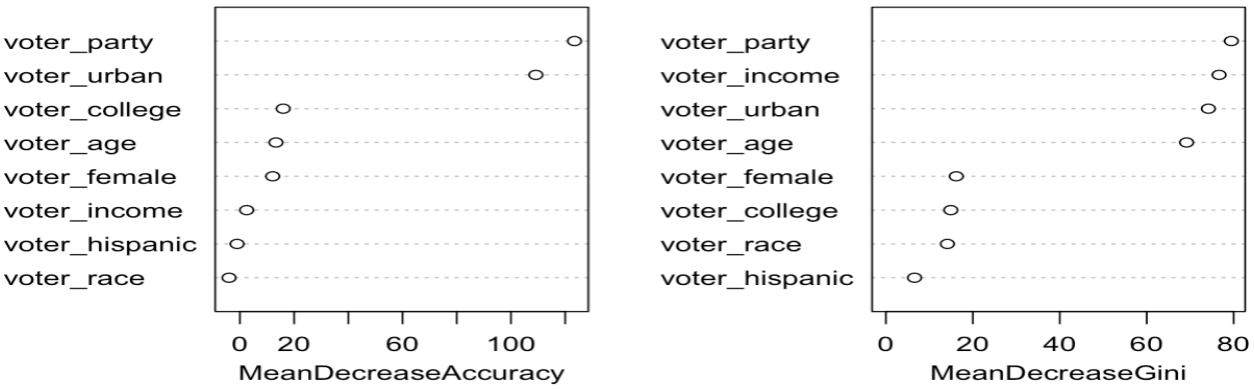
Next, we decided to analyze the data through the second consideration of the decision tree. First, the data was divided into training set for model fitting and validation set for predicting model accuracy with a 50% ratio respectively (Appendix E). After dividing the data, the tree function allowed the model to be constructed based on the training set, and the plot of the decision tree to be checked (Appendix I). As shown in the plot from Appendix, a data overfit may become a concern if the tree is too large. Therefore, a small tree with a small variance can help avoid the issues regarding a data overfit.; to reduce the tree, we pruned it. To efficiently prune this tree, we used cross-validation; the cv.tree function allows to determine what the optimal number of leaf nodes are and cross validation errors (Appendix J). Through the pruning process, we learned that the optimal number of leaf numbers was 8. With the optimal number of leaf numbers we checked, through the prune.misclass function, we made a new tree that was pruned.



In addition, the information in the model we obtained helped us interpret the tree specifically (Appendix F). Through the tree above that we finally considered, we were able to predict the probability of how people would rate Tim Walz's performance. Based on the above tree, people who support the DFL party in urban and suburban have a 77.77% probability of strongly approving Walz's performance. Then, those who do not support the Republican party and live in the rural area with a chance of 81.25% to show that they strongly disapprove of Walz's performance.

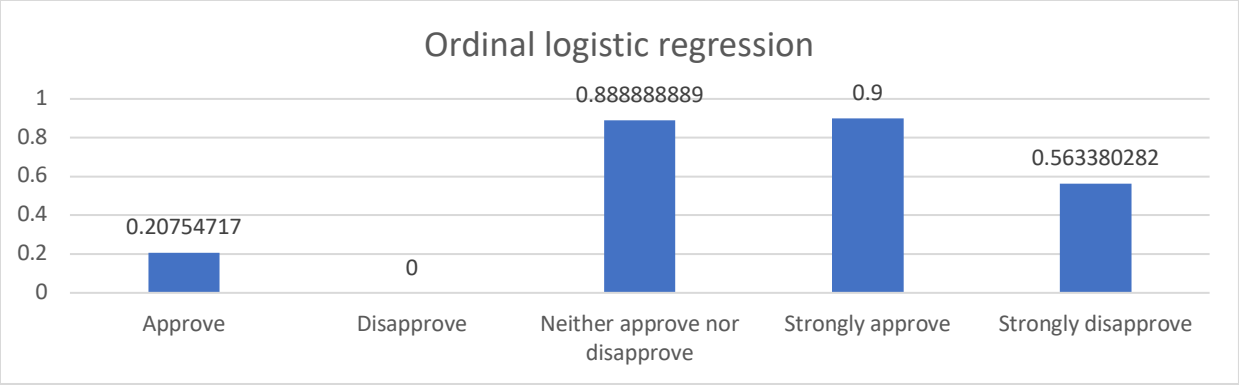
Lastly, we decided to analyze the data through the third consideration of the random forest. Like that of the prior two methods, the data was divided into training set for model fitting and validation set for predicting model accuracy with a 50% ratio respectively. After dividing the data, the random forest function allowed the model to be constructed based on the training set

(Appendix G). In the model, we could check the variable importance like shown below.



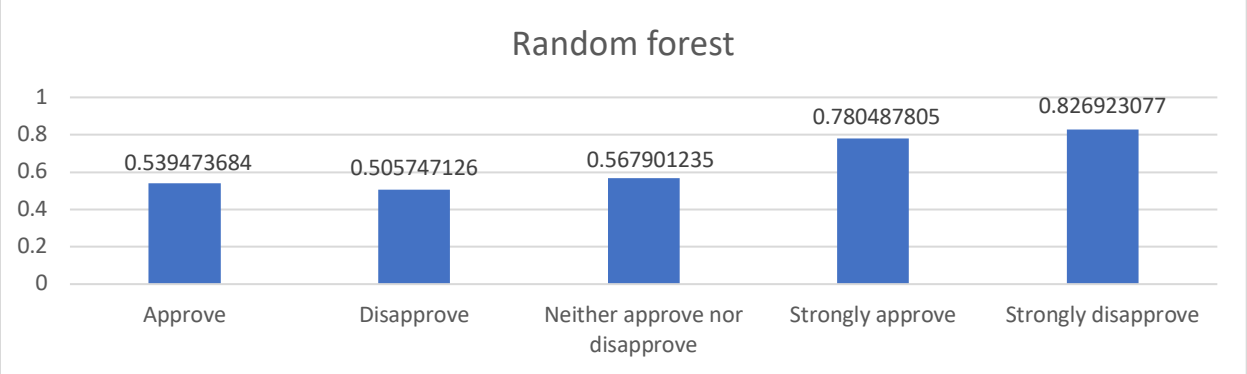
The left side represents the degree of contribution to prediction accuracy, and the right side represents the degree of contribution to improving the purity of nodes; the left graph can be interpreted that the prediction accuracy is reduced by the number corresponding to the x-axis if the variable is not selected. Hence, based on the mean decrease accuracy graphs, the voter_party and voter_urban are very important variables for prediction accuracy. It was also proved that the variables we chose were proper for the data in the ordinal logistic regression.

Finally, we compared the error rates of the ordinal logistic regression, the decision tree models, and the random forest. As I mentioned earlier, we already fitted the two models based on the training set. To check each model's accuracy rates in detail, we needed to examine the confusion matrices (Appendix H). A misclassification error rate and a classification rate were obtained using the Confusion matrix which uses the fitted model and validation dataset. Based on the confusion matrix, we could estimate the prediction of the accuracy rate for the response variables for each level. To predict the accuracy rate of the specific level based on the validation set observation, we calculated the number of matched levels divided by the whole trial number.



In the case of the ordinal logistic regression, it had the highest probability to predict the levels 'strongly approve' (72.79%) and 'strongly disapprove' (83.33%). The only thing we were worried about was that it had a zero percentage to predict 'disapprove'.

In the case of the decision tree, it had highest probability to predict the levels 'Neither approve nor disapprove' (84.92%) and 'strongly approve' (90%); it also predicted 'strongly disapprove' in 56.34 percentage, and it was not a bad probability.



This confirms that all levels are accurately predicted with a probability of over 50%. Although the random forest is slightly less likely to predict ‘Neither approve nor disapprove’ and ‘strongly approve’ than the proportional odds model, it shows significantly better predictive performance overall than the previously identified models. In particular, it was surprising that the probability of extreme values such as ‘strongly approve’ and ‘strongly disapprove’ had a predictive probability of more than 78%. According to above three plots, we could know that the random forest had better accuracy for predicting each level than the other two methods. These results supported the entire prediction accuracy rate. We then evaluated the models’ accuracy rates by comparing the fitted model and validation set through the confusion matrices; a 54.6% accuracy rate was identified for the ordinal logistic regression. For the decision tree, a 59% accuracy rate was identified, and the random forest identified a 63.2% accuracy rate. Even though the accuracy rate did not have a large difference, we could know that the random forest was more accurate than the ordinal logistic regression by 8.6% difference and the decision tree by 4.2% difference. Through the results from above, we confirmed that the results of the analysis through the random forest was more accurate than the results of the analysis through the ordinal logistic regression and decision tree. Hence, for creating project strategies, the random forest should be used to achieve more efficient results.

Discussion

Our final choice of analysis method (random forest) enabled a concise and accurate prediction of the response variable (Walz's performance) we wanted to analyze. We expected that the random forest was an efficient analysis method, because the prediction accuracy was higher than the ordinal logistic regression and the decision tree. We were also able to verify the results

objectively because we checked the prediction accuracies of the two models through using the confusion matrix after dividing the two data randomly. We believed that the prediction accuracy obtained from these statistical results was a sufficiently reliable result. Although the accuracy rate of the random forest has been the highest among the three methods we had considered so far, it is impossible to conclude that the random forest is perfect. The random forest had the disadvantage of being difficult to interpret the outcome, because it uses several trees; therefore, we need to be careful about interpretation. In conclusion, we had 1000 observations in the dataset. If we had more observations, we could have gotten more an accurate result and prediction accuracy rates. In improving the accuracy of the analysis, we believe that it will also be extremely helpful to consider the shortcomings of the random forest for this project. I am confident that if we supplement the analysis results, we can design specific strategies to meet the needs for Walz's campaign.

CITATION

Algeri, Sara. (2021). Introduction to Statistical Learning. Retrieved from University of Minnesota - Twin Cities Stat 4052

Faraway, J. J. (2016). *Extending the linear model with R: Generalized linear, mixed effects and nonparametric regression models*. Boca Raton: CRC Press, Taylor & Francis Group.

Fullerton, A. S. (2009). A Conceptual Framework for Ordered Logistic Regression Models. Sociological Methods & Research, 38(2), 306–347. https://doi.org/10.1177/0049124109346162

Song, Y. Y., & Lu, Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, *27*(2), 130–135. https://doi.org/10.11919/j.issn.1002-0829.215044

Yurong Zhong, "The analysis of cases based on decision tree," 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 2016, pp. 142-147, doi: 10.1109/ICSESS.2016.7883035

APPENDIX

Appendix A:

```
#from small model to large
model1 = polr(Walz_approval ~ voter_party,data=Phase2.dat, Hess=TRUE)
anova(reduce.model,model1)

## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##         Model Resid. df Resid. Dev   Test    Df LR stat.      Pr(Chi)
## 1           1       996   3163.140
## 2 voter_party       994   3132.444 1 vs 2     2 30.69556 2.160444e-07

model2 = polr(Walz_approval ~ voter_party+voter_urban,data=Phase2.dat, Hess=TRUE) #Best
model
anova(model1,model2)

## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                   Model Resid. df Resid. Dev   Test    Df LR stat.
## 1             voter_party       994   3132.444
## 2 voter_party + voter_urban       992   3094.891 1 vs 2     2 37.55386
```

```
##        Pr(Chi)
## 1
## 2 7.002974e-09
```

```
model3 = polr(Walz_approval ~ voter_party+voter_urban+voter_age,data=Phase2.dat,
Hess=TRUE)
anova(model2,model3)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                                Model Resid. df Resid. Dev   Test    Df
## 1             voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_age       991   3094.854 1 vs 2     1
##     LR stat.   Pr(Chi)
## 1
## 2 0.03669359 0.8480902
```

```
model4 = polr(Walz_approval ~ voter_party+voter_urban+voter_college,data=Phase2.dat,
Hess=TRUE)
anova(model2,model4)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                                Model Resid. df Resid. Dev   Test    Df
```

```
## 1                voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_college       991   3094.751 1 vs 2     1
##   LR stat.   Pr(Chi)
## 1
## 2 0.139693 0.7085864

model5 = polr(Walz_approval ~ voter_party+voter_urban+voter_income,data=Phase2.dat,
Hess=TRUE)
anova(model2,model5)

## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                                  Model Resid. df Resid. Dev   Test    Df
## 1               voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_income       991   3094.268 1 vs 2     1
##    LR stat.   Pr(Chi)
## 1
## 2 0.6220857 0.4302733

model6 = polr(Walz_approval ~ voter_party+voter_urban+voter_female,data=Phase2.dat,
Hess=TRUE)
anova(model2,model6)

## Likelihood ratio tests of ordinal regression models
##
```

```
## Response: Walz_approval
##                                 Model Resid. df Resid. Dev   Test    Df
## 1               voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_female       991   3092.103 1 vs 2     1
##   LR stat.    Pr(Chi)
## 1
## 2 2.787104 0.09502579

model6 = polr(Walz_approval ~ voter_party+voter_urban+voter_race,data=Phase2.dat,
Hess=TRUE)
anova(model2,model6)

## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                                 Model Resid. df Resid. Dev   Test    Df
## 1               voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_race       987   3093.165 1 vs 2     5
##   LR stat.   Pr(Chi)
## 1
## 2 1.725975 0.8856113

model7 = polr(Walz_approval ~ voter_party+voter_urban+voter_hispanic,data=Phase2.dat,
Hess=TRUE)
anova(model2,model7)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                                 Model Resid. df Resid. Dev   Test    Df
## 1                voter_party + voter_urban       992   3094.891
## 2 voter_party + voter_urban + voter_hispanic       991   3094.849 1 vs 2     1
##     LR stat.   Pr(Chi)
## 1
## 2 0.04198988 0.8376391

model8 = polr(Walz_approval ~ voter_party*voter_urban,data=Phase2.dat, Hess=TRUE) #Best
model
anova(model2,model8)

## Likelihood ratio tests of ordinal regression models
##
## Response: Walz_approval
##                    Model Resid. df Resid. Dev   Test    Df LR stat. Pr(Chi)
## 1 voter_party + voter_urban       992   3094.891
## 2 voter_party * voter_urban       988   2659.379 1 vs 2     4 435.5115       0
```

Appendix B:

```
summary(model8)
```

```
## Call:
## polr(formula = Walz_approval ~ voter_party * voter_urban, data = Phase2.dat,
##     Hess = TRUE)
##
## Coefficients:
##                                          Value Std. Error t value
## voter_partyIndependent                   1.100     0.2687   4.093
## voter_partyRepublican                    5.582     0.3691  15.124
## voter_urbanSuburban                      1.238     0.2437   5.079
## voter_urbanUrban                         2.671     0.2368  11.278
## voter_partyIndependent:voter_urbanSuburban -2.241     0.3695  -6.064
## voter_partyRepublican:voter_urbanSuburban  -6.371     0.4508 -14.131
## voter_partyIndependent:voter_urbanUrban    -4.044     0.3934 -10.280
## voter_partyRepublican:voter_urbanUrban     -8.256     0.4642 -17.787
##
## Intercepts:
##                                                  Value    Std. Error t value
## Approve|Disapprove                               -0.7347   0.1692    -4.3432
## Disapprove|Neither approve nor disapprove         0.1807   0.1645     1.0983
## Neither approve nor disapprove|Strongly approve   1.6236   0.1757     9.2397
## Strongly approve|Strongly disapprove              3.9511   0.2265    17.4426
##
```

```
## Residual Deviance: 2659.379

## AIC: 2683.379
```

Appendix C:

$$\text{logit}(\hat{P}(Y \leq 1)) = -0.7347 + 1.1\ \text{voter\_partyIndependent} + 5.582\ \text{voter\_partyRepublican} + 1.238\ \text{voter\_urbanSuburban} + 2.671\ \text{voter\_urbanUrban} - 2.241\ \text{voter\_partyIndependent:voter\_urbanSuburban} - 6.371\ \text{voter\_partyRepublican:voter\_urbanSuburban} - 4.044\ \text{voter\_partyIndependent:voter\_urbanUrban} - 8.256\ \text{voter\_partyRepublican:voter\_urbanUrban}$$

$$\text{logit}(\hat{P}(Y \leq 2)) = 0.1807 + 1.1\ \text{voter\_partyIndependent} + 5.582\ \text{voter\_partyRepublican} + 1.238\ \text{voter\_urbanSuburban} + 2.671\ \text{voter\_urbanUrban} - 2.241\ \text{voter\_partyIndependent:voter\_urbanSuburban} - 6.371\ \text{voter\_partyRepublican:voter\_urbanSuburban} - 4.044\ \text{voter\_partyIndependent:voter\_urbanUrban} - 8.256\ \text{voter\_partyRepublican:voter\_urbanUrban}$$

Appendix D:

```
chisq.test(Phase2.dat$Walz_approval,predict(model8))


##

##  Pearson's Chi-squared test

##

## data:  Phase2.dat$Walz_approval and predict(model8)

## X-squared = 1336.3, df = 12, p-value < 2.2e-16
```

Appendix E:

```
#arranging data set

#setting tranning and valid set

Phase2.dat=read.csv("2020 DFL Data v2.csv")

set.seed(4893)

Phase2.dat[,c("voter_party","voter_age","voter_college","voter_income","voter_female","voter_

race","voter_hispanic","voter_urban","Walz_approval")] n=nrow(Phase2.dat)

train_id=sample(1:n,n/2)
```

```
train=Phase2.dat[train_id,] #50 % of trainning data

val=Phase2.dat[-train_id,] #50 % of testing data
```

Appendix F:

```
model.3=prune.misclass(model.1,best=8)

model.3

## node), split, n, deviance, yval, (yprob)

##       * denotes terminal node

##

##  1) root 500 1571.00 Neither approve nor disapprove ( 0.194000 0.162000 0.284000 0.234000

0.126000 )

##    2) voter_party: DFL 193  442.80 Strongly approve ( 0.212435 0.046632 0.191710 0.544041

0.005181 )

##      4) voter_urban: Rural 58  117.80 Neither approve nor disapprove ( 0.241379 0.155172

0.586207 0.000000 0.017241 ) *

##      5) voter_urban: Suburban,Urban 135  162.50 Strongly approve ( 0.200000 0.000000

0.022222 0.777778 0.000000 ) *

##    3) voter_party: Independent,Republican 307  900.90 Neither approve nor disapprove

( 0.182410 0.234528 0.342020 0.039088 0.201954 )

##      6) voter_urban: Rural 92  145.70 Strongly disapprove ( 0.000000 0.380435 0.032609

0.000000 0.586957 )

##       12) voter_party: Independent 44   75.75 Disapprove ( 0.000000 0.590909 0.068182

0.000000 0.340909 ) *

##       13) voter_party: Republican 48   46.33 Strongly disapprove ( 0.000000 0.187500
```

```
0.000000 0.000000 0.812500 ) *

##      7) voter_urban: Suburban,Urban 215  554.90 Neither approve nor disapprove ( 0.260465

0.172093 0.474419 0.055814 0.037209 )

##       14) voter_urban: Suburban 105  252.80 Neither approve nor disapprove ( 0.133333

0.266667 0.514286 0.009524 0.076190 ) *

##       15) voter_urban: Urban 110  256.20 Neither approve nor disapprove ( 0.381818 0.081818

0.436364 0.100000 0.000000 )

##         30) voter_party: Independent 43   88.65 Approve ( 0.511628 0.000000 0.255814

0.232558 0.000000 )

##           60) voter_age < 67 26   38.12 Approve ( 0.692308 0.000000 0.038462 0.269231

0.000000 ) *

##           61) voter_age > 67 17   32.60 Neither approve nor disapprove ( 0.235294 0.000000

0.588235 0.176471 0.000000 ) *

##         31) voter_party: Republican 67  136.80 Neither approve nor disapprove ( 0.298507

0.134328 0.552239 0.014925 0.000000 ) *
```

Appendix G:

```
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

model_1=randomForest(walz_approval~.,data=train,mtry=3,importance=TRUE)
model_1

##
## Call:
##  randomForest(formula = walz_approval ~ ., data = train, mtry = 3,
importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
```

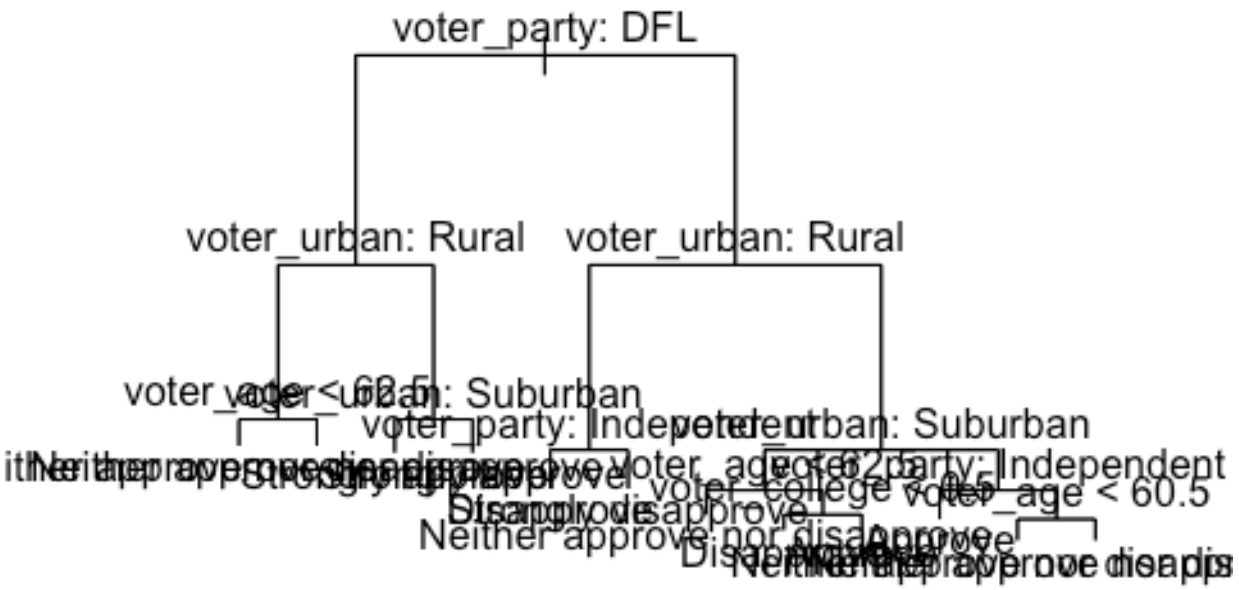
```
##
##         OOB estimate of  error rate: 36.4%
## Confusion matrix:
##                                 Approve Disapprove
## Approve                              36          1
## Disapprove                            4         32
## Neither approve nor disapprove       26         14
## Strongly approve                     17          0
## Strongly disapprove                   0         14
##                                 Neither approve nor disapprove Strongly
approve
## Approve                                                     42
16
## Disapprove                                                  33
0
## Neither approve nor disapprove                              92
2
## Strongly approve                                             1
111
## Strongly disapprove                                          1
0
##                                 Strongly disapprove class.error
## Approve                                           0   0.6210526
## Disapprove                                       10   0.5949367
## Neither approve nor disapprove                    1   0.3185185
## Strongly approve                                  0   0.1395349
## Strongly disapprove                              47   0.2419355
```

Appendix H:

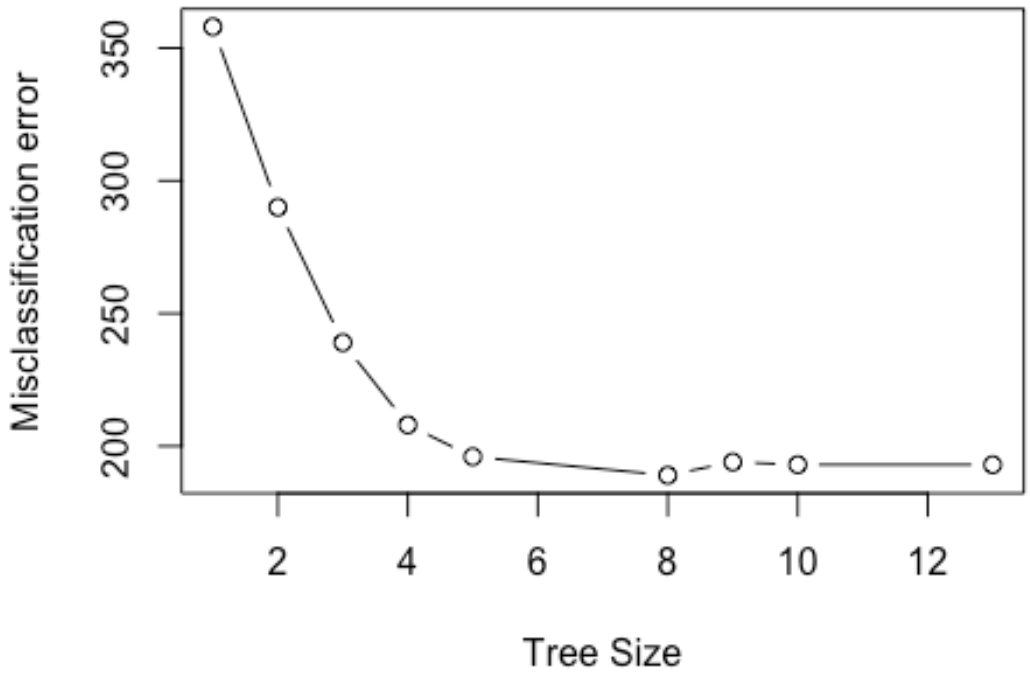
```
table(pred,val$walz_approval)

##
## pred                              Approve Disapprove
##   Approve                              44          2
##   Disapprove                            1         44
##   Neither approve nor disapprove       37         32
##   Strongly approve                     24          0
##   Strongly disapprove                   0          9
##
## pred                              Neither approve nor disapprove
##   Approve                                                    21
##   Disapprove                                                 11
##   Neither approve nor disapprove                             93
##   Strongly approve                                            1
##   Strongly disapprove                                         0
##
## pred                              Strongly approve Strongly disapprove
##   Approve                                       13                   0
##   Disapprove                                     0                  28
```

```
##   Neither approve nor disapprove                  0                   0
##   Strongly approve                               97                   0
##   Strongly disapprove                             0                  43
```

Appendix I:



Appendix J:



Appendix K:

#All R codes

#Ordinal logistic regression

```
library(MASS)
#Pick proper model
reduce.model = polr(walz_approval~1, data=train, Hess=TRUE)
full.model = polr(walz_approval ~
voter_party+voter_age+voter_college+voter_income+voter_female
          +voter_race+voter_hispanic+voter_urban,data=train, Hess=TRUE)
anova(reduce.model,full.model)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##
Model
## 1
1
## 2 voter_party + voter_age + voter_college + voter_income + voter_female +
voter_race + voter_hispanic + voter_urban
##   Resid. df Resid. Dev   Test    Df LR stat.      Pr(Chi)
## 1       496   1568.978
## 2       482   1520.416 1 vs 2    14 48.56276 1.060421e-05
```

```
chisq.test(train$walz_approval,predict(full.model))
```

```
##
##  Pearson's Chi-squared test
##
## data:  train$walz_approval and predict(full.model)
## X-squared = 77.286, df = 8, p-value = 1.717e-13
```

```
#from small model to large
model1 = polr(walz_approval ~ voter_party,data=train, Hess=TRUE)
anova(reduce.model,model1)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##         Model Resid. df Resid. Dev   Test    Df LR stat.      Pr(Chi)
## 1           1       496   1568.978
## 2 voter_party       494   1540.912 1 vs 2     2 28.06597 8.045475e-07
```

```
model2 = polr(walz_approval ~ voter_party+voter_urban,data=train, Hess=TRUE)
#Best model
anova(model1,model2)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                       Model Resid. df Resid. Dev   Test    Df LR stat.
## 1               voter_party       494   1540.912
## 2 voter_party + voter_urban       492   1530.854 1 vs 2     2 10.05865
##       Pr(Chi)
## 1
## 2 0.006543226
```

```
model3 = polr(walz_approval ~ voter_party+voter_urban+voter_age,data=train,
Hess=TRUE)
anova(model2,model3)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                   Model Resid. df Resid. Dev   Test    Df
## 1             voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_age       491   1530.793 1 vs 2     1
##     LR stat.   Pr(Chi)
## 1
## 2 0.06033679 0.8059644
```

```
model4 = polr(walz_approval ~
voter_party+voter_urban+voter_college,data=train, Hess=TRUE)
anova(model2,model4)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                     Model Resid. df Resid. Dev   Test
Df
## 1                 voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_college       491   1528.871 1 vs 2
1
##   LR stat.   Pr(Chi)
## 1
## 2 1.983032 0.1590713
```

```
model5 = polr(walz_approval ~
voter_party+voter_urban+voter_income,data=train, Hess=TRUE)
anova(model2,model5)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                    Model Resid. df Resid. Dev   Test
Df
## 1                voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_income       491   1529.684 1 vs 2
```

```
1
##   LR stat.   Pr(Chi)
## 1
## 2 1.169648 0.2794735
```

```
model6 = polr(walz_approval ~
voter_party+voter_urban+voter_female,data=train, Hess=TRUE)
anova(model2,model6)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                     Model Resid. df Resid. Dev   Test
Df
## 1                voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_female       491   1529.684 1 vs 2
1
##   LR stat.  Pr(Chi)
## 1
## 2 1.169806 0.279441
```

```
model6 = polr(walz_approval ~ voter_party+voter_urban+voter_race,data=train,
Hess=TRUE)
anova(model2,model6)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                   Model Resid. df Resid. Dev   Test    Df
## 1              voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_race       487   1528.858 1 vs 2     5
##   LR stat.   Pr(Chi)
## 1
## 2 1.995769 0.8497301
```

```
model7 = polr(walz_approval ~
voter_party+voter_urban+voter_hispanic,data=train, Hess=TRUE)
anova(model2,model7)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                                       Model Resid. df Resid. Dev   Test
Df
## 1                  voter_party + voter_urban       492   1530.854
## 2 voter_party + voter_urban + voter_hispanic       491   1530.368 1 vs 2
1
##    LR stat.   Pr(Chi)
## 1
## 2 0.4852479 0.4860548
```

```r
model8 = polr(walz_approval ~ voter_party*voter_urban,data=train, Hess=TRUE)
#Best model
anova(model2,model8)
```

```
## Likelihood ratio tests of ordinal regression models
##
## Response: walz_approval
##                       Model Resid. df Resid. Dev   Test    Df LR stat.
Pr(Chi)
## 1 voter_party + voter_urban       492   1530.854
## 2 voter_party * voter_urban       488   1261.222 1 vs 2     4 269.6314
0
```

```r
# Predict probability
predict(model8,data.frame(voter_party="DFL",
voter_urban="Rural"),type="probs")
```

```
##                       Approve                    Disapprove
##                    0.27625435                    0.21096905
## Neither approve nor disapprove               Strongly approve
##                    0.34151778                    0.15991633
##            Strongly disapprove
##                    0.01134249
```

```r
predict(model8,data.frame(voter_party="DFL",
voter_urban="Suburban"),type="probs")
```

```
##                       Approve                    Disapprove
##                    0.07787784                    0.09583543
## Neither approve nor disapprove               Strongly approve
##                    0.34335851                    0.43363289
##            Strongly disapprove
##                    0.04929533
```

```r
predict(model8,data.frame(voter_party="DFL",
voter_urban="Urban"),type="probs")
```

```
##                       Approve                    Disapprove
##                    0.01889471                    0.02685252
## Neither approve nor disapprove               Strongly approve
##                    0.15049502                    0.61849831
##            Strongly disapprove
##                    0.18525945
```

```r
predict(model8,data.frame(voter_party="Independent",
voter_urban="Rural"),type="probs")
```

```
##                       Approve                    Disapprove
##                    0.23918039                    0.19983038
## Neither approve nor disapprove               Strongly approve
##                    0.36040897                    0.18684193
```

```
##            Strongly disapprove
##                     0.01373833
```

```
predict(model8,data.frame(voter_party="Independent",
voter_urban="Suburban"),type="probs")
```

```
##                        Approve                     Disapprove
##                     0.26342600                     0.20754750
## Neither approve nor disapprove                Strongly approve
##                     0.34832659                     0.16860346
##            Strongly disapprove
##                     0.01209645
```

```
predict(model8,data.frame(voter_party="Independent",
voter_urban="Urban"),type="probs")
```

```
##                        Approve                     Disapprove
##                    0.337344420                    0.221592705
## Neither approve nor disapprove                Strongly approve
##                    0.306906589                    0.125627641
##            Strongly disapprove
##                    0.008528645
```

```
predict(model8,data.frame(voter_party="Republican",
voter_urban="Rural"),type="probs")
```

```
##                        Approve                     Disapprove
##                   0.0006714365                   0.0009982990
## Neither approve nor disapprove                Strongly approve
##                   0.0067763565                   0.1245747708
##            Strongly disapprove
##                   0.8669791372
```

```
predict(model8,data.frame(voter_party="Republican",
voter_urban="Suburban"),type="probs")
```

```
##                        Approve                     Disapprove
##                     0.23538658                     0.19846821
## Neither approve nor disapprove                Strongly approve
##                     0.36218248                     0.18993745
##            Strongly disapprove
##                     0.01402528
```

```
predict(model8,data.frame(voter_party="Republican",
voter_urban="Urban"),type="probs")
```

```
##                        Approve                     Disapprove
##                    0.312513599                    0.218348724
## Neither approve nor disapprove                Strongly approve
##                    0.321274012                    0.138322157
##            Strongly disapprove
##                    0.009541508
```

```
#Comparing both models through accuracy
predict_model8 <-  predict(model8,val)
table(val$walz_approval, predict_model8)
```

```
##                                  predict_model8
##                                   Approve Disapprove
##   Approve                              22          0
##   Disapprove                            1          0
##   Neither approve nor disapprove        9          0
##   Strongly approve                      9          0
##   Strongly disapprove                   0          0
##                                  predict_model8
##                                   Neither approve nor disapprove
##   Approve                                                     52
##   Disapprove                                                  78
##   Neither approve nor disapprove                             112
##   Strongly approve                                             2
##   Strongly disapprove                                         31
##                                  predict_model8
##                                   Strongly approve Strongly disapprove
##   Approve                                       32                   0
##   Disapprove                                     0                   8
##   Neither approve nor disapprove                 5                   0
##   Strongly approve                              99                   0
##   Strongly disapprove                            0                  40
```

```
mean(as.character(val$walz_approval) != as.character(predict_model8))
```

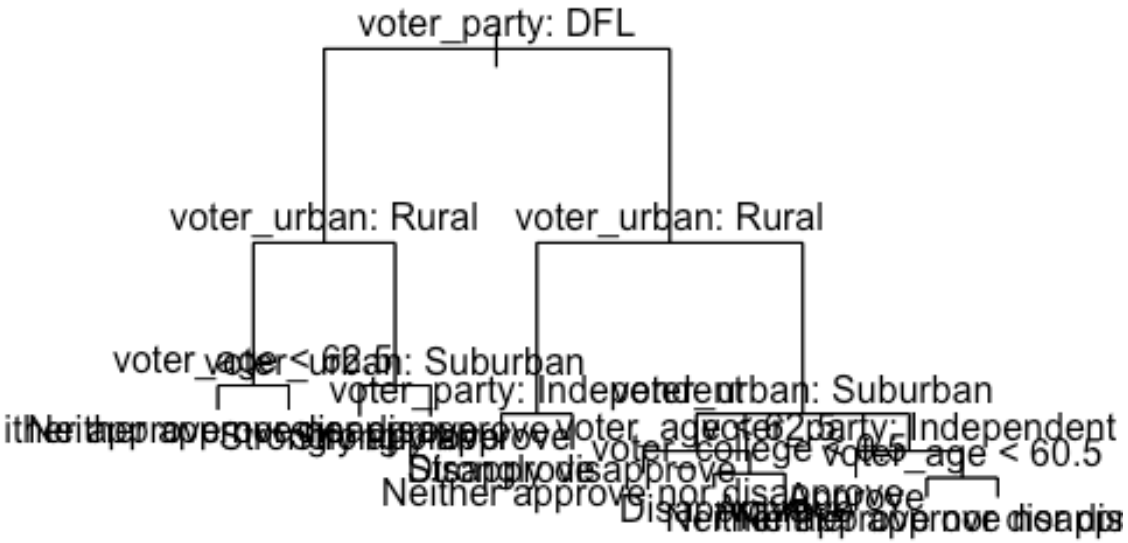
```
## [1] 0.454
```

#Decision tree

```
#arranging data set
#setting tranning and valid set
Phase2.dat=read.csv("2020 DFL Data v2.csv")
set.seed(5000)
new.dat=Phase2.dat[,c("voter_party","voter_age","voter_college","voter_income
","voter_female","voter_race","voter_hispanic","voter_urban","walz_approval")
]
n=nrow(new.dat)
train_id=sample(1:n,n/2)
train=new.dat[train_id,] #trainning data
val=new.dat[-train_id,] #testing data

# Two way to express decision tree
library(tree)
model.1=tree(walz_approval~.,data=train)
summary(model.1)
```

```
##
## Classification tree:
## tree(formula = walz_approval ~ ., data = train)
## Variables actually used in tree construction:
## [1] "voter_party"   "voter_urban"   "voter_age"     "voter_college"
## Number of terminal nodes:  12
## Residual mean deviance:  1.472 = 718.3 / 488
## Misclassification error rate: 0.336 = 168 / 500
```

```
plot(model.1)
text(model.1,pretty=0)
```




```
#library(rpart)
#library(rpart.plot)
#model.1=rpart(walz_approval~.,data=train, cp=.03)
#rpart.plot(model.1, box.palette="RdBu", shadow.col="gray", nn=TRUE)

predtion1=predict(model.1,val,type="class")
table(predtion1,val$walz_approval)

##
## predtion1                       Approve Disapprove
##   Approve                            22          2
##   Disapprove                          2         39
```

```
##   Neither approve nor disapprove      50         38
##   Strongly approve                    32          0
##   Strongly disapprove                  0          8
##
## predtion1                          Neither approve nor disapprove
##   Approve                                                    14
##   Disapprove                                                 14
##   Neither approve nor disapprove                             93
##   Strongly approve                                            5
##   Strongly disapprove                                         0
##
## predtion1                          Strongly approve Strongly disapprove
##   Approve                                         9                   0
##   Disapprove                                      0                  31
##   Neither approve nor disapprove                  2                   0
##   Strongly approve                               99                   0
##   Strongly disapprove                             0                  40
```

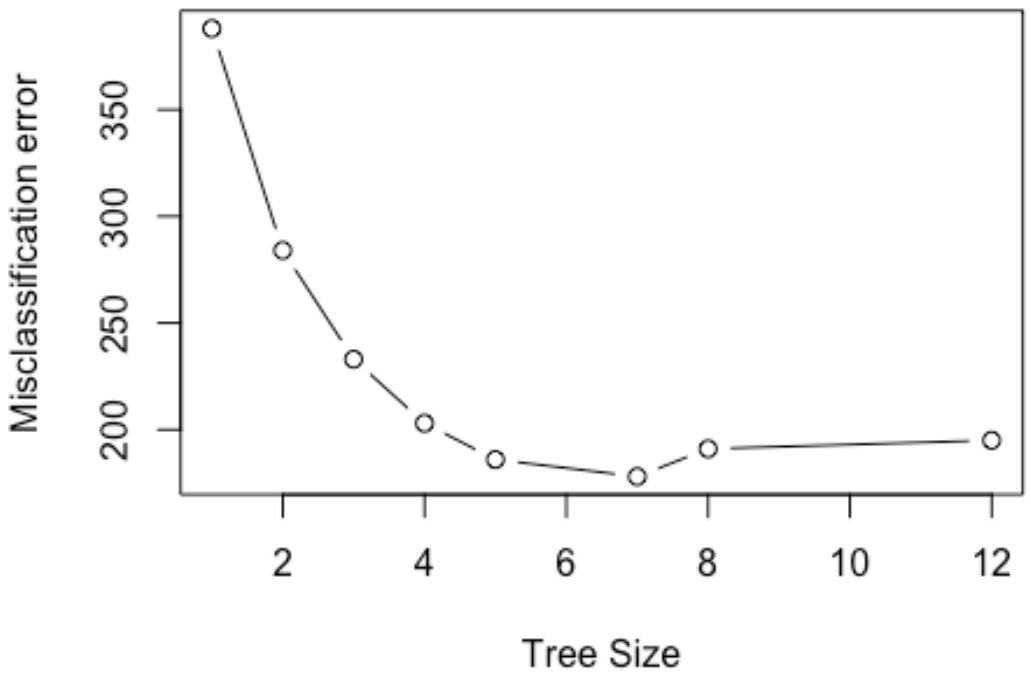
```
mean(predtion1!=val$walz_approval)
```

```
## [1] 0.414
```

```
model.2=cv.tree(model.1,FUN=prune.misclass)
model.2
```

```
## $size
## [1] 12  8  7  5  4  3  2  1
##
## $dev
## [1] 195 191 178 186 203 233 284 388
##
## $k
## [1] -Inf  0.0  4.0  5.5 18.0 32.0 51.0 81.0
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(model.2$size, model.2$dev, type="b",xlab="Tree
Size",ylab="Misclassification error")
```




```
model.3=prune.misclass(model.1,best=7)
model.3
```

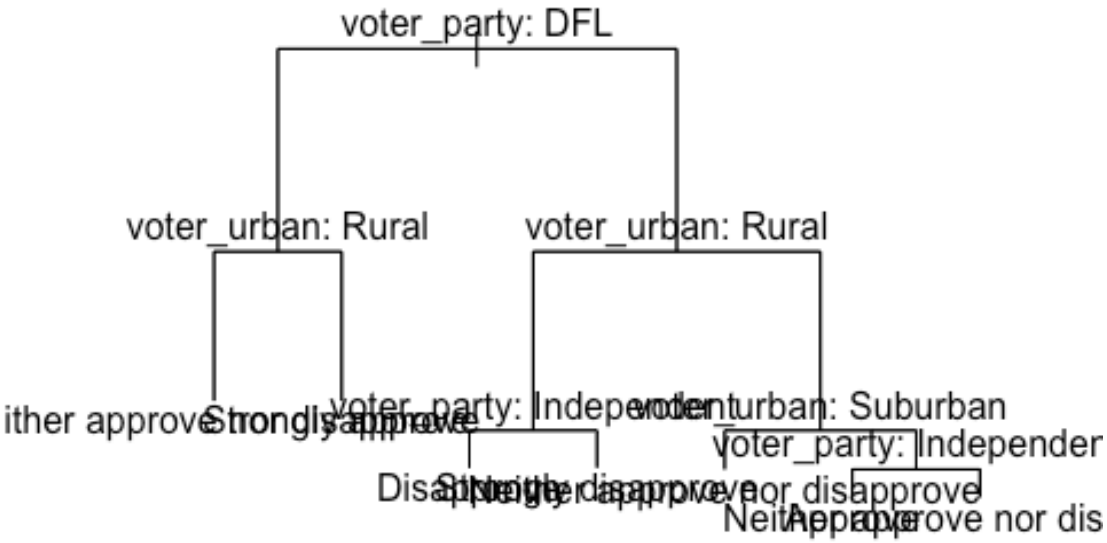
```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 500 1569.00 Neither approve nor disapprove ( 0.19000 0.15800
0.27000 0.25800 0.12400 )
##    2) voter_party: DFL 201  454.90 Strongly approve ( 0.18905 0.04975
0.17413 0.57711 0.00995 )
##      4) voter_urban: Rural 61  140.70 Neither approve nor disapprove
( 0.24590 0.16393 0.54098 0.01639 0.03279 ) *
##      5) voter_urban: Suburban,Urban 140  145.30 Strongly approve ( 0.16429
0.00000 0.01429 0.82143 0.00000 ) *
##    3) voter_party: Independent,Republican 299  884.60 Neither approve nor
disapprove ( 0.19064 0.23077 0.33445 0.04348 0.20067 )
##      6) voter_urban: Rural 87  131.60 Strongly disapprove ( 0.00000
0.36782 0.02299 0.00000 0.60920 )
##       12) voter_party: Independent 36   52.55 Disapprove ( 0.00000 0.72222
0.05556 0.00000 0.22222 ) *
##       13) voter_party: Republican 51   36.95 Strongly disapprove ( 0.00000
0.11765 0.00000 0.00000 0.88235 ) *
##      7) voter_urban: Suburban,Urban 212  550.50 Neither approve nor
```

```
disapprove ( 0.26887 0.17453 0.46226 0.06132 0.03302 )
##       14) voter_urban: Suburban 103  240.90 Neither approve nor disapprove
( 0.16505 0.25243 0.51456 0.00000 0.06796 ) *
##       15) voter_urban: Urban 109  265.60 Neither approve nor disapprove
( 0.36697 0.10092 0.41284 0.11927 0.00000 )
##         30) voter_party: Independent 40   82.98 Approve ( 0.50000 0.00000
0.22500 0.27500 0.00000 ) *
##         31) voter_party: Republican 69  150.90 Neither approve nor
disapprove ( 0.28986 0.15942 0.52174 0.02899 0.00000 ) *
```

```
summary(model.3)
```

```
##
## Classification tree:
## snip.tree(tree = model.1, nodes = c(4L, 5L, 31L, 14L))
## Variables actually used in tree construction:
## [1] "voter_party" "voter_urban"
## Number of terminal nodes:  7
## Residual mean deviance:  1.725 = 850.3 / 493
## Misclassification error rate: 0.344 = 172 / 500
```

```
plot(model.3)
text(model.3, pretty = 0)
```



```
predtion2=predict(model.3,val,type="class")
table(predtion2,val$walz_approval)
```

```
##
## predtion2                          Approve Disapprove
##   Approve                               22          1
##   Disapprove                             0         27
##   Neither approve nor disapprove        52         51
##   Strongly approve                      32          0
##   Strongly disapprove                    0          8
##
## predtion2                          Neither approve nor disapprove
##   Approve                                                      9
##   Disapprove                                                   5
##   Neither approve nor disapprove                             107
##   Strongly approve                                             5
##   Strongly disapprove                                          0
##
## predtion2                          Strongly approve Strongly disapprove
##   Approve                                         9                   0
##   Disapprove                                      0                  25
##   Neither approve nor disapprove                  2                   6
##   Strongly approve                               99                   0
##   Strongly disapprove                             0                  40
```

```
mean(predtion2!=val$walz_approval)
```

```
## [1] 0.41
```

#Random forest

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
model_1=randomForest(walz_approval~.,data=train,mtry=3,importance=TRUE)
model_1
```

```
##
## Call:
##  randomForest(formula = walz_approval ~ ., data = train, mtry = 3,
importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 36.4%
## Confusion matrix:
##                                 Approve Disapprove
```

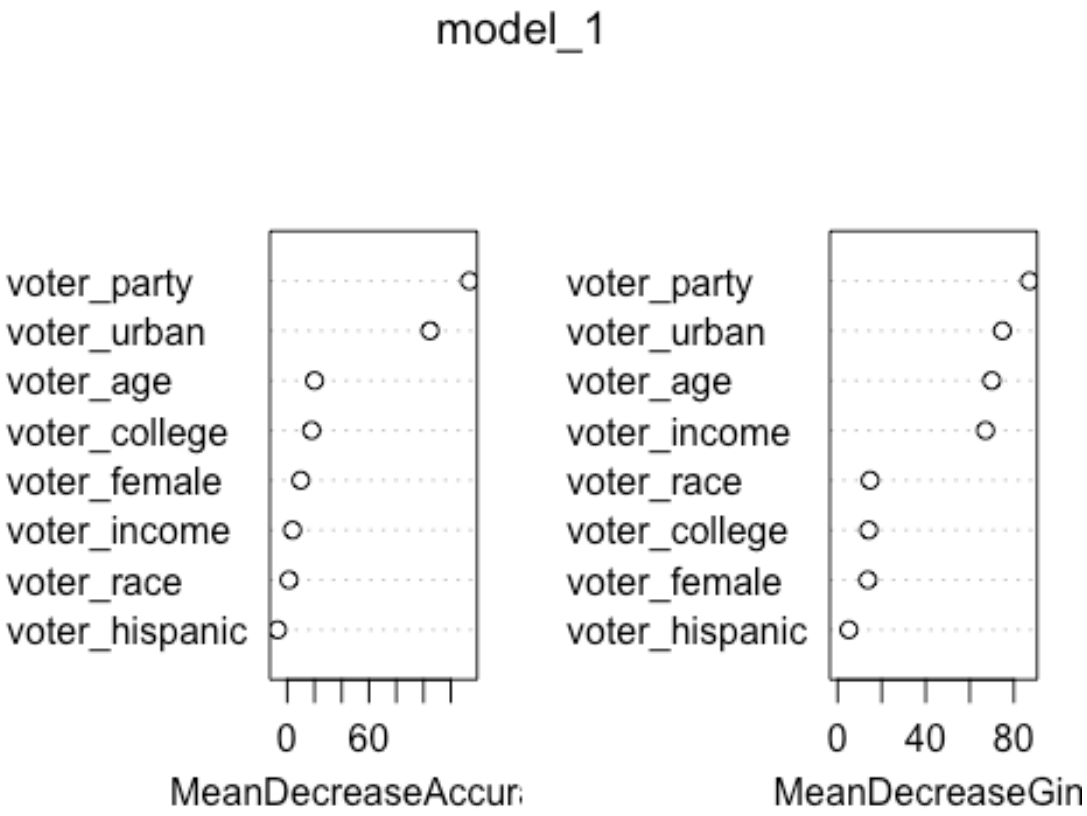
```
## Approve                              36          1
## Disapprove                            4         32
## Neither approve nor disapprove       26         14
## Strongly approve                     17          0
## Strongly disapprove                   0         14
##                                Neither approve nor disapprove Strongly
approve
## Approve                                                    42
16
## Disapprove                                                 33
0
## Neither approve nor disapprove                             92
2
## Strongly approve                                            1
111
## Strongly disapprove                                         1
0
##                                Strongly disapprove class.error
## Approve                                          0   0.6210526
## Disapprove                                      10   0.5949367
## Neither approve nor disapprove                   1   0.3185185
## Strongly approve                                 0   0.1395349
## Strongly disapprove                             47   0.2419355
```

```
importance(model_1)
```

```
##                   Approve  Disapprove Neither approve nor disapprove
## voter_party    35.5371590 40.98501436                      65.5560235
## voter_age       9.5886826  9.34656665                      11.9160387
## voter_college   9.8186807  6.54639881                      -0.2210908
## voter_income    1.5393937 -0.85351831                      -0.2329277
## voter_female    9.5932030  3.09808555                       2.6051013
## voter_race     -1.5046910  0.07314496                       4.9601775
## voter_hispanic  0.5959893 -2.19064768                      -6.2219125
## voter_urban    36.3137775 37.61798920                      58.4781347
##                Strongly approve Strongly disapprove MeanDecreaseAccuracy
## voter_party          110.950887           57.2831418            133.718551
## voter_age              7.272942            6.8355394             20.079595
## voter_college         17.232694            5.8273564             17.943177
## voter_income           4.128146            5.8244935              3.904364
## voter_female           2.628863            3.0654807              9.953851
## voter_race            -4.176064            3.1773338              1.346894
## voter_hispanic        -5.409545           -0.3338688             -7.034718
## voter_urban           67.943020           71.2838337            104.765559
##                MeanDecreaseGini
## voter_party           87.093750
## voter_age             69.976890
## voter_college         14.129530
## voter_income          67.126590
## voter_female          13.737567
```

```
## voter_race             14.752828
## voter_hispanic          5.122627
## voter_urban            74.809406
```

```
varImpPlot(model_1)
```




```
pred=predict(model_1,val)
table(pred,val$walz_approval)
```

```
##
## pred                               Approve Disapprove
##   Approve                               44          2
##   Disapprove                             1         44
##   Neither approve nor disapprove        37         32
##   Strongly approve                      24          0
##   Strongly disapprove                    0          9
##
## pred                               Neither approve nor disapprove
##   Approve                                                      21
##   Disapprove                                                   11
##   Neither approve nor disapprove                               93
##   Strongly approve                                              1
##   Strongly disapprove                                           0
##
```

```
## pred                                Strongly approve Strongly disapprove
##   Approve                                        13                    0
##   Disapprove                                      0                   28
##   Neither approve nor disapprove                  0                    0
##   Strongly approve                               97                    0
##   Strongly disapprove                             0                   43
```

```
mean(pred!=val$walz_approval)
```

```
## [1] 0.358
```