Modelling Writeup

2022-12-01

Methods

Primarily, we are looking to predict the results of the 2025 Canadian Election, i.e., which party (out of the 6 major parties) will win the election. But immediately, we are faced with a major challenge; namely, the Canadian electoral districts are getting redistributed as part of a 10 year cycle. As stated in the introduction, by Canada's parliamentary system, the party that wins the most seats would win the federal elections Canada (n.d.). To work around that, we will instead look to both predict the share of votes at a national level and at the provincial level, the most geographically granular level our data sets allow (both the survey and census data we have).

Framing our goal as primarily a prediction problem opens us up to some interesting candidate models than if our goal were only interpretation or only prediction. One go-to model is the multilevel regression with post-stratification model, popularized in part from Andrew Gelman, who supervised the use of it for adjusting Xbox survey data to the 2012 American presidential election in Wang et al. (2015). We will largely follow this framework but explore extensions to this model by adding more group-level fixed effects and using regression splines through Generalized Additive Mixed Effects Models (GAMMs). We will also explore Generalized Additive Models without random effects (GAMs).

We choose these models over others since they provide the ability to faithfully model random effects, some level of interpretability, along with a unified framework for regularizers (penalization likelihoods) and link functions that all work to create a model that allows for much greater non-linearity. In particular, we liked the "non-parametric" nature of GAMM model, i.e., we can let the data speak for itself in selecting the best functional form for each regression spline. Similarly, we can use restricted maximum likelihood (REML) to select λ for the penalty term contribution.

The main benefit over more linear approaches (e.g., logistic or linear mixed effects models) is that GAMs allow us to capture non-linearities in the data. Additionally, computational support for penalized likelihood is more flexible with GAMMs (and GAMs more broadly) than packages that support generalized linear models (or generalized mixed effects models), and regression splines can be individually penalized. Such properties are especially appealing in our prediction task as penalization goes a long way to controlling model variance, i.e., reducing overfitting. Further, while multilevel regression provides some regularization, post-stratification cells counts can grow extremely quickly and end up being very sparse (very little observations in each cell) Wang et al. (2015), and so extra penalization from the splines could help our prediction accuracy.

One remedy to this would be to look towards Bayesian approaches for intepretable and flexible regularized models, but creating more informative priors that provide meaningful regularization are not sensible. COVID-19 has caused a rather large macro shifts in the Canadian economic and political landscapes, especially with respect to potential predictors, and Statistics Canada in 2021 has noted that the pandemic has impacted certain groups particularly hard and continues to do so Arora (2021). For example, visible minority groups have experienced, as a whole, high unemployment, and certain industries (like manufacturing) have been hit hard, meaning that workers in those industries would be impacted too. As a result of these changes, "standard" Bayesian techniques, such as those used in Wang et al. (2015) are not applicable as we cannot comfortably form reasonable and ethically informative beliefs on parameters for prior distributions, e.g., do we really know how a visible minority worker in manufacturing will value their money with respect to voting preferences today and how to encode that as a distributional parameter(s)?

On the other hand, we prefer GAMs and GAMMs over more "black box" techniques such as neural networks or tree-based models because such models are often more uninterpretable and less comparable to methods present in the literature. We do not consider neural networks here because there is not enough data to fit a likely overparameterized model. We also prefer GAM/GAMMs over tree-based models because while they are more interpretable, they do not allow for the careful addition of predictors into the model, e.g., they do not allow for the careful consideration of group-level effects, and they often do not generalize well to new combinations or levels of "factor" variables (cannot really extrapolate). So GAMMs are a good balance of a predictably capable model and an interpretable one.

However, for feature selection, we used a recursive feature elimination algorithm (RFE) that employs random forests. This is discussed more in detail under the model specifics, but we chose this to have a principled approach towards choosing a set of parameters useful for prediction (and another set of reduced parameters for a simpler model better for interpretation). We decided on random forests as the model for RFE because it is non-parametric (we did not want to assume a functional form of the data before choosing the predictors to explain the data) and allows for extraction of feature importance. Also for feature selection, we did not choose the features that would have the least multicollinearity because our goal ultimately is prediction accuracy and correlated features are still informative our voting choices, e.g., age and income.

Model Specifics and Feature Selection

As mentioned above, we fit either GAMs or GAMMs, and we do so on a 80-20 train-test split. In other words, the models were fit on 80% of our available data with 20% of the remaining data only used for model evaluation. Both sets were chosen randomly.

Then, we first considered closely the predictors we investigated on sociological lines (sex, province, education, and income). And then decided to expand our set of potential predictors to the entire intersection of variables between the census data and survey data. As the full set of predictors has 17 elements, we felt the need to explore dimensionality reduction methods. This could have been done through shrinkage methods like L1 (lasso) penalties, but we wanted to see it more explicit, and as we computed average favour points based on the categorical predictor groupings (later used for post-stratification), such methods would lead to potentially inappropriate average favour scores should an L1 penalty, for example, shrink a predictor to have no effect. We decided on RFE because through cross-validation and resampling, we can pretty exhaustively consider permutations of the full predictor set for all possible sizes of predictor sets (1 to 17) to evaluate which predictors may be useful in prediction. In our experiments, we ran the RFE algorithm with random forest for reasons mentioned above (non-parametric and feature importance calculations). We ran this algorithm using the R `caret` package with 5-fold cross-validation five times.

Algorithm 2: Recursive feature elimination incorporating resampling

```
for Each Resampling Iteration do
    Partition data into training and test/hold–back set via resampling
    Tune/train the model on the training set using all predictors
    Predict the held–back samples
    Calculate variable importance or rankings
    for Each subset size S_i, i = 1 ... S do
        Keep the S_i most important variables
        [Optional] Pre–process the data
        Tune/train the model on the training set using S_i predictors
        Predict the held–back samples
        [Optional] Recalculate the rankings for each predictor
    end
end
Calculate the performance profile over the S_i using the held–back samples
Determine the appropriate number of predictors
Estimate the final list of predictors to keep in the final model
Fit the final model based on the optimal S_i using the original training set
```

Figure 1: RFE Algorithm from Caret Documentation

The RFE algorithm found that a reduced complexity model had the best accuracy in predicting the individual vote choices at $\approx 79.67\%$ accuracy and a ≈ 0.71 Kappa, compared to $\approx 79.54\%$ accuracy and a ≈ 0.7078 Kappa. As these metrics were all really close and are model-based, we decided to try both sets of predictors. We considered Kappa too since it accounts for class imbalance (which election vote choices certain do have) by computing:

$$\kappa = \frac{p_0 - p_e}{1 - p_e}$$

Where p_0 are the correctly classified observations and p_e is the random change probability McHugh (2012).

So, we decided to use the following predictors for the reduced complexity model: `liberal_favour`, `conservative_favour`, `ndp_favour`, `bloc_favour`, `green_favour`, `people_favour`, `province`, and `language`, allowing us to create a new data set that contained only these variables by subsetting the full data set.

We then created three models per data set (full and reduced), one that was just random effects (incorporating all categorical variables available to us in either the reduced complexity data set or the full data set), one that was a mixed effects model, and one that was a fixed effects model (only the favour predictors). Additionally, we used a multinomial logit link as we have that the data (party predictions) are multinomially distributed since they are 6 parties with some probability of being vote for by some voter.

In general, all of our GAMs and GAMMs take a form similar to a GLM. We have here that NumSplines is the number of splines we use (equal to the number of favour predictors), $k = 6$ as we have 6 parties, and Z_i is the matrix of the categorical variables for observation i and U is the matrix of corresponding random effects. Putting it together we have:

$$\begin{aligned}
y_i &\sim \text{Multinomial}(1, \mathbf{p}),\ \mathbf{p} = \{p_1, p_2, ..., p_k\} \\
\ln(\frac{\Pr(y_i = 1)}{\Pr(y_i = k)}) &= \sum_{i=1}^{\text{NumSplines}} f(x_i) + Z_i U \\
\ln(\frac{\Pr(y_i = 2)}{\Pr(y_i = k)}) &= \sum_{i=1}^{\text{NumSplines}} f(x_i) + Z_i U \\
&... \\
\ln(\frac{\Pr(y_i = k-1)}{\Pr(y_i = k)}) &= \sum_{i=1}^{\text{NumSplines}} f(x_i) + Z_i U
\end{aligned}$$

We can use the softmax function to compute the predicted probabilities instead of log-odds for party p and the corresponding linear combination for class p from above:

$$\Pr(y_i = p) = \text{softmax}(\sum_{i=1}^{\text{NumSplines}} f(x_i) + Z_i U)$$

Finally, note that as we are using p-splines, our splines take the following form for some number of basis functions K, where basis function $k,\ 1 \leq k \leq K$ is denoted $\mathbf{b_k}(\mathbf{x})$ Wood (n.d.):

$$\mathcal{P} = \sum_{k}^{K} \beta_k \mathbf{b_k}(\mathbf{x}) = \sum_{\mathbf{k=2}}^{\mathbf{K-1}} (\beta_{\mathbf{k-1}} - \mathbf{2}\beta_{\mathbf{k}} + \beta_{\mathbf{k+1}})^{\mathbf{2}}$$

Recall for above that a basis function is just a simpler function that allows for some smoothness/non-linearities, and a spline is a weighted sum of basis functions. For computation, we used the `mgcv` R package

for all models. The random effects models were fit using REML due to the random effects, while the fixed effects models were fit using penalized iteratively reweighted least squares (PIRLS) Larsen (2015). For the fixed effects model, no explicit penalty term contribution parameter (λ) was chosen, and was chosen using the default (Generalized Cross Validation). Where for given $\mathbf{B}$ (model matrix), H (hat matrix), we have Larsen (2015):

$$GCV = \frac{n\|\sqrt{W}(z - \mathbf{B}'\beta)\|^2}{(n - \mathrm{tr}(H))^2}$$

However, for mixed effects models, λ was chosen as an averaged value over the model weights from the REML likelihood objective Larsen (2015):

$$\mathcal{L}_{\mathrm{REML}}(\hat{\beta}, \lambda) = \int f(y \mid \beta) f(\beta) d\beta$$

For easier conceptual understanding, we present our model formulas in `lmer` style syntax, our `mgcv` model formula is as follows for the full models (with full model code in the appendix):

```
# Random effects model

vote_choice ~ s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
           s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
           s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
           s(born_in_canada, bs = "re") + s(citizenship, bs="re")



# Mixed effects model

vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
            s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
            s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
            s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
            s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
            s(born_in_canada, bs = "re") + s(citizenship, bs="re")


# Fixed effects model

vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
  s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps")
```

Finally, for our reduced complexity models have have the following (with full code in the appendix):

```
# Random effects model

vote_choice ~ s(province, bs="re") + s(language, bs = "re")

# Mixed effects model

vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
  s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
  s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
```

```
  s(province, bs = "re") + s(language, bs = "re")

# Fixed effects model

vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
  s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
  s(people_favour, bs="ps") + s(bloc_favour, bs="ps")
```

Post-Stratification

For post-stratification, we considered the two post-stratification estimators derived from Wang et al. (2015), first the national post-stratified estimator for the prediction (predicted probability) of party i:

$$\hat{y}^{ps}_{i,\text{ National}} = \frac{\sum_{j=1}^{J} N_{ji} \cdot \hat{y}_{ji}}{\sum_{j=1}^{J} N_{ji}}$$

Where above N_{ji} is the number of observations from cell j and party i and $\hat{y}_{ji}$ is the estimate (predicted probability) of party i for cell j before post-stratifying. We also consider the following for provincial predictions:

$$\hat{y}^{ps}_{i,\text{ Province}} = \frac{\sum_{j \in J_s} N_{ji} \cdot \hat{y}_{ji}}{\sum_{j \in J_s} N_{ji}}$$

As we have reduced and full complexity models, we tried to compute the post-stratified predictions for both sets of models. Unfortunately, the full complexity model meant that we had many post-stratification cells, which became extremely sparse, and additionally, the usable survey observations did not cover many post-stratification cells, meaning adjusted estimates were not accurate nor representative of the population. So for full complexity models, we simply did not adjust using post-stratification, but instead used them as a comparison for our post-stratified results from the reduced complexity models.

Results

Our model was able to predict the test set results very accurately on both a national and provincial level.

Limitations

The data we have available is fundamentally out of date. The assumption we are making here is that Canadian voters will behave (have the same voting preferences) in 2025 as they did in 2019, the year of the CES 2019 online survey, and that Canadian demographics will look similar in 2025 as they did in 2016, the year of the GSS data. As a result, while our model performs well on predicting the test set proportions, we cannot be sure that the model will perform similarly with the 2025 election as we are essentially predicting the 2021 election. While the models certainly appear capable, fitting the model on newer data would help increase our confidence in our models.

Additionally, `mgcv` , the package used to fit GAMMs runs into computational constraints. Particularly, multinomial models are not parallelizeable currently, leading to extremely long fitting times, e.g., the mixed effects models take ~5 hours to fit, making evaluation methods like LOOCV practically impossible. This is a major drawback since cross-validation is a less biased evaluation method, especially as cross-validation folds increase.

Reproduceability Note

As noted in the limitations section, models take an extremely long time to fit, so we have not enabled them to fit for kitting chunks. We have included the code to do so at the bottom (not shown in the knitted document). All data cleaning/wrangling is reproduceable by knitting as required.

Appendix

Full Models Code

```
# Random effects only model
mgcv::gam(data = train_full, formula =
  list(
   vote_choice ~ s(province, bs="re") + s(health, bs = "re") +
     s(age_category, bs = "re") + s(education, bs = "re") + s(income_category, bs="re") +
     s(marital_status, bs = "re") + s(language, bs = "re") + s(gender, bs = "re") +
     s(owns_house, bs = "re") + s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
   ~ s(province, bs="re") + s(health, bs = "re") +
     s(age_category, bs = "re") + s(education, bs = "re") + s(income_category, bs="re") +
     s(marital_status, bs = "re") + s(language, bs = "re") + s(gender, bs = "re") +
     s(owns_house, bs = "re") + s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
   ~ s(province, bs="re") + s(health, bs = "re") +
     s(age_category, bs = "re") + s(education, bs = "re") + s(income_category, bs="re") +
     s(marital_status, bs = "re") + s(language, bs = "re") + s(gender, bs = "re") +
     s(owns_house, bs = "re") + s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
   ~ s(province, bs="re") + s(health, bs = "re") +
     s(age_category, bs = "re") + s(education, bs = "re") + s(income_category, bs="re") +
     s(marital_status, bs = "re") + s(language, bs = "re") + s(gender, bs = "re") +
     s(owns_house, bs = "re") + s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
   ~ s(province, bs="re") + s(health, bs = "re") +
     s(age_category, bs = "re") + s(education, bs = "re") + s(income_category, bs="re") +
     s(marital_status, bs = "re") + s(language, bs = "re") + s(gender, bs = "re") +
     s(owns_house, bs = "re") + s(born_in_canada, bs = "re") + s(citizenship, bs="re")
      ),
     family = mgcv::multinom(K=5),
     method="REML")

# Mixed effects model
mgcv::gam(data = train_full, formula =
list(
  vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
    s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
    s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
    s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
    s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
    s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
    ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
    s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
    s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
    s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
    s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
```

```
    s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
    ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
    s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
    s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
    s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
    s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
    s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
    ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
    s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
    s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
    s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
    s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
    s(born_in_canada, bs = "re") + s(citizenship, bs="re"),
    ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
    s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
    s(province, bs="re") + s(health, bs = "re") +s(age_category, bs = "re") +
    s(education, bs = "re") + s(income_category, bs="re") + s(marital_status, bs = "re") +
    s(language, bs = "re") + s(gender, bs = "re") + s(owns_house, bs = "re") +
    s(born_in_canada, bs = "re") + s(citizenship, bs="re")
     ),
     family = mgcv::multinom(K=5),
     method = "REML")

# Fixed effects model
mgcv::gam(data = train_full, formula = list(
  vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
   s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") ,
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
   s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps"),
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
   s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps"),
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
   s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps"),
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
   s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps")
   ),
   family = mgcv::multinom(K=5))
```

Reduced Models Code

```
# Random effects model
mgcv::gam(data = train_reduced, formula =
 list(
   vote_choice ~ s(province, bs="re") + s(language, bs = "re"),
   ~ s(province, bs="re") + s(language, bs = "re"),
   ~ s(province, bs="re") + s(language, bs = "re"),
   ~ s(province, bs="re") + s(language, bs = "re"),
   ~ s(province, bs="re") + s(language, bs = "re")
),
family = mgcv::multinom(K=5),
method="REML")
```

```
# Mixed effects model
mgcv::gam(data = train_reduced,
 formula = list(
   vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
     s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
     s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
     s(province, bs = "re") + s(language, bs = "re"),
   ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
     s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
     s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
     s(province, bs = "re") + s(language, bs = "re"),
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
     s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
     s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
     s(province, bs = "re") + s(language, bs = "re"),
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
     s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
     s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
     s(province, bs = "re") + s(language, bs = "re"),
   ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") +
     s(ndp_favour, bs="ps") + s(green_favour, bs="ps") +
     s(people_favour, bs="ps") + s(bloc_favour, bs="ps") +
     s(province, bs = "re") + s(language, bs = "re")
 ),
 family = mgcv::multinom(K=5),
 method = "REML")


# Fixed effects model

mgcv::gam(data = train_reduced,
 formula = list(
   vote_choice ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
     s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") ,
  ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
     s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") ,
   ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
     s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") ,
   ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
     s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps") ,
   ~ s(liberal_favour, bs="ps") + s(conservative_favour, bs="ps") + s(ndp_favour, bs="ps") +
     s(green_favour, bs="ps") + s(people_favour, bs="ps") + s(bloc_favour, bs="ps")
 ),
 family = mgcv::multinom(K=5))
```

References

Arora, Anil. 2021. "COVID-19 in Canada: A One-Year Update on Social and Economic Impacts." *Government of Canada, Statistics Canada*. Government of Canada, Statistics Canada. https://www150.statcan.gc.ca/n1/pub/11-631-x/11-631-x2021001-eng.htm#a4.

Canada, Elections. n.d. "Timeline for the Redistribution of Federal Electoral Districts – Elections Canada." *Elections Canada*. https://www.elections.ca/content.aspx?section=res&dir=cir%2Fred%2Fover&

amp;document=index&lang=e.

Larsen, Kim. 2015. "Gam: The Predictive Modeling Silver Bullet." *Multithreaded*. Stich Fix. https://multithreaded.stitchfix.com/blog/2015/07/30/gam/.

McHugh, Mary L. 2012. "Interrater Reliability: The Kappa Statistic." *Biochemia Medica*, October. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/.

Wang, Wei, David Rothschild, Sharad Goel, and Andrew Gelman. 2015. "Forecasting Elections with Non-Representative Polls." *International Journal of Forecasting* 31 (3): 980–91. https://doi.org/https://doi.org/10.1016/j.ijforecast.2014.06.001.

Wood, Simon. n.d. "An Introduction to GAMS Based on Penalized Regression Splines." https://www.maths.ed.ac.uk/~swood34/talks/snw-Koln.pdf.