Data Analysis Project Description

In this project, I wish to predict the proportion of cabinet ministries each party gains based on various factors, such as their political ideology (left-wing / right-wing), number of seats in the past cabinet, the party's shapley index, and more. For prediction, I used and compared the performance of various ML models in including linear regression model, decision tree, XGBoost, and polynomial regression model. The data are from European parliamentary democracies and ministries (e.g., minister of defense) are the rewards parties gain from forming a winning coalition. See, https://en.wikipedia.org/wiki/Coalition_government for an overview.

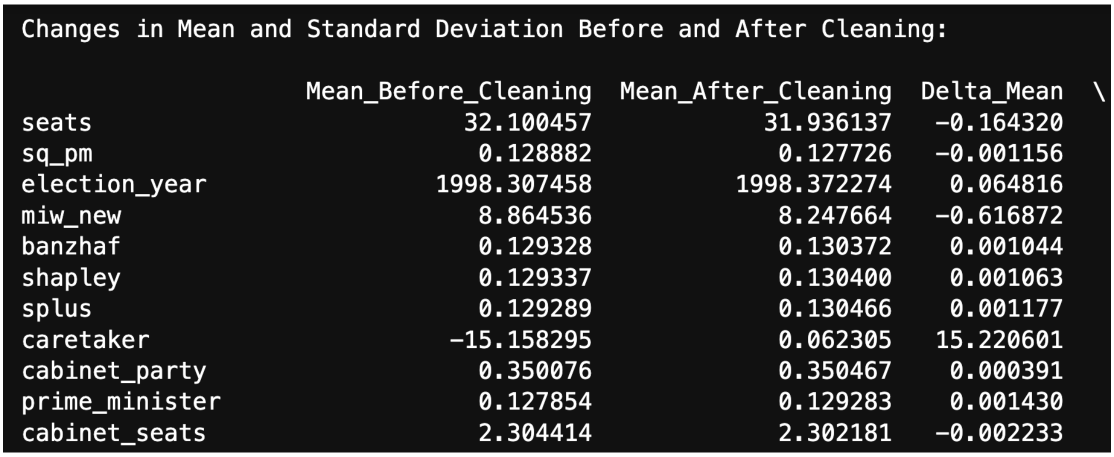
Data Preprocessing

1. Predictor & Empty/Outlier Entry Removal:

First, I dropped all the rows where the value of the dependent variable 'cabinet_proportion' is missing. Then, I dropped 'party', 'cabinet_name', 'party_name', 'party_name_english', 'country' because the value of these variables is string, which isn't very helpful in a regression. I proceed to drop the ID variables like 'cabinet_id', 'party_id', 'country_id', 'election_id' because I believe if I create a bunch of dummy variables for every ID in every ID variable, it might hinder my model's interpretability and cause it to overfit on this dataset.

After that, I dropped the features that are essentially the same as others to further increase interpretability and reduce collinearity. Specifically, I dropped 'base', 'lag_largest_parl', and 'lag_largest_cab' because it's the same as 'seats', 'largest_parl', and 'largest_cab'. I also dropped 'election_date', 'start_date' because they provide similar information as 'year'. In addition, I dropped 'post_election' because we were told to ignore it.

Finally, I dropped columns with more than 15 missing entries, this include variables: 'left_rightx', 'left_righty', and 'coalition_total'. I then removed rows that has missing value placeholders like '-9999' by removing rows that have outliers that are more than 10 std away from the mean of the column. I have checked the mean and std before and after I removed the rows to ensure that the isn't a significant statistic change in mean and std for the columns due to the row removal. The following is a part of the statistics I measured:



```
Changes in Mean and Standard Deviation Before and After Cleaning:

                Mean_Before_Cleaning  Mean_After_Cleaning  Delta_Mean  \
seats                      32.100457            31.936137   -0.164320
sq_pm                       0.128882             0.127726   -0.001156
election_year            1998.307458          1998.372274    0.064816
miw_new                     8.864536             8.247664   -0.616872
banzhaf                     0.129328             0.130372    0.001044
shapley                     0.129337             0.130400    0.001063
splus                       0.129289             0.130466    0.001177
caretaker                 -15.158295             0.062305   15.220601
cabinet_party               0.350076             0.350467    0.000391
prime_minister              0.127854             0.129283    0.001430
cabinet_seats               2.304414             2.302181   -0.002233
```

The significant change in ‘caretaker’ is due to the removal of the placeholder value ‘-9999’.

Data Standardization

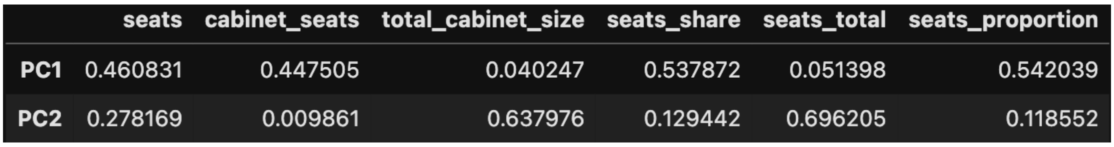
All the data except for the binary variables are standardized.

2. PCA

To reduce the dimension of the training data, I constructed two PCA clusters to mesh together variables I believe to have similar traits or I suspect to be colinear. To assist my understanding of which variables are more associated with a latent variable, I wrote the function ‘top_variables_by_latent’ that returns the top (math.ceil(explained_variance / 1 / number of original variables)) original variables explained by the latent variable. I will use my theory to assume a dimension of latent space, and I will try out different dimensions and update my theory based on the result of the function before I set the final latent space dimension.

After I have run the PCA, I keep the original variables in the dataset if they have a loading less than (1 / total number of original variables) for all latent variables because I believe they are not very well explained by the latent variables.

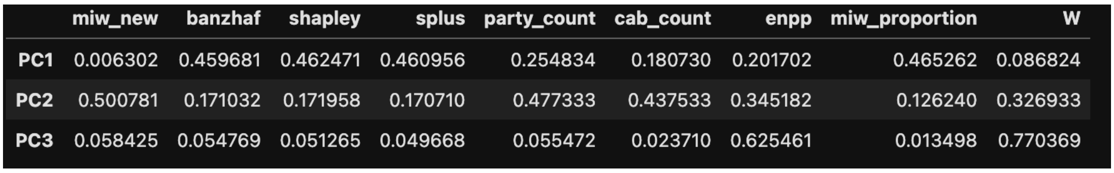
For the first PCA cluster, I lumped together variables I believe that tell how many seats a party occupies in both parliament and cabinet, which indicates their population-wise influence. I eventually formed 2 latent variables that explain 0.811 of the total variance of the 6 variables I merged, and they are: 'seats', 'cabinet_seats', 'total_cabinet_size', 'seats_share', 'seats_total', 'seats_proportion'. Following is the loading table for the two latent variables:



	seats	cabinet_seats	total_cabinet_size	seats_share	seats_total	seats_proportion
PC1	0.460831	0.447505	0.040247	0.537872	0.051398	0.542039
PC2	0.278169	0.009861	0.637976	0.129442	0.696205	0.118552

I decided to keep all 6 variables in the PCA because based on the loading (correlation) and the total variance explained by each latent variable I think they are all pretty well explained by the principle components.

For the second PCA cluster, I lumped together variables I believe that shows how pivotal a party is in terms of forming a majority coalition. I eventually formed 3 latent variables that explain 0.861 of the total variance of the 8 variables I merged, and they are: 'miw_new', 'banzhaf', 'shapley','splus', 'party_count', 'cab_count', 'enpp', 'miw_proportion', 'W'. Following is the loading table for the three latent variables:



	miw_new	banzhaf	shapley	splus	party_count	cab_count	enpp	miw_proportion	W
PC1	0.006302	0.459681	0.462471	0.460956	0.254834	0.180730	0.201702	0.465262	0.086824
PC2	0.500781	0.171032	0.171958	0.170710	0.477333	0.437533	0.345182	0.126240	0.326933
PC3	0.058425	0.054769	0.051265	0.049668	0.055472	0.023710	0.625461	0.013498	0.770369

I decided to keep all 8 variables in the PCA because based on the loading (correlation) and the total variance explained by each latent variable, I think they are all pretty well explained by the principle components.

3. Models attempted and k-fold cross-validation

For hyperparameter optimization, I organized the models I wanted to compare into a list of dictionaries. The four types of models I want to compare are: lasso linear regression, lasso degree 2 polynomial regression, XGBoost, and a single decision tree.

For lasso linear and degree-2 polynomial regression model, the following list of lambda values are used to find the best lambda coefficient: [0.0001, 0.001, 0.01, 0.1, 1, 10].

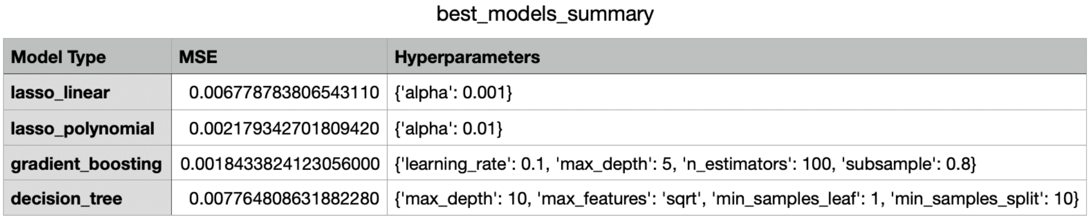
For XGBoost, a grid search with the following entries is used:

```
gb_param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [3, 5, 7],

    'learning_rate': [0.01, 0.1, 0.2],

    'subsample': [0.8, 1.0]

}
```

For a single decision tree, a grid search with the following entries is used:

```
dt_param_grid = {

        'max_depth': [None, 5, 10, 20],

        'min_samples_split': [2, 5, 10],

        'min_samples_leaf': [1, 2, 4],

        'max_features': ['sqrt', 'log2']

}
```

Model Results:



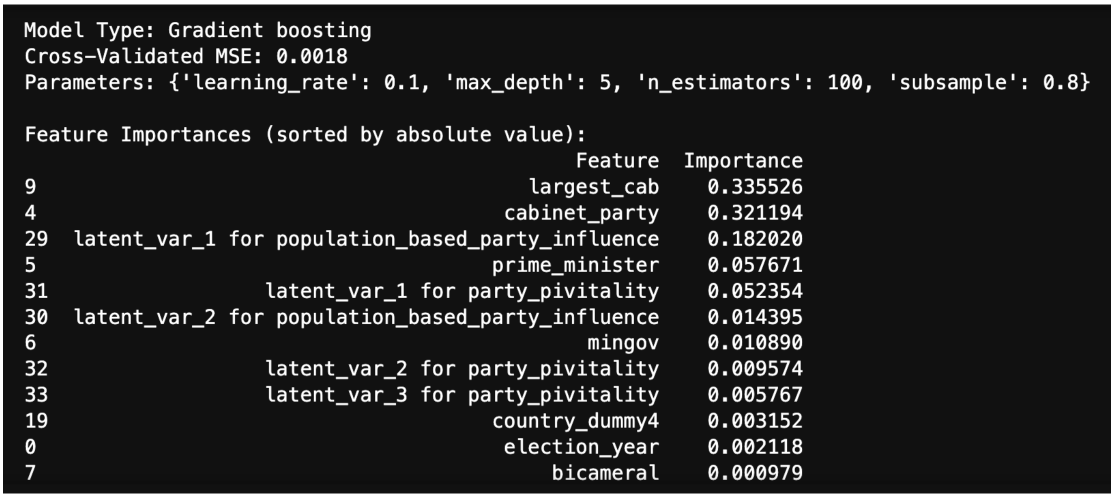
best_models_summary

Model Type	MSE	Hyperparameters
lasso_linear	0.006778783806543110	{'alpha': 0.001}
lasso_polynomial	0.002179342701809420	{'alpha': 0.01}
gradient_boosting	0.0018433824123056000	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 100, 'subsample': 0.8}
decision_tree	0.007764808631882280	{'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 10}

Among all the models tested, XGBoost has the smallest CV MSE (0.0018) and a single decision tree has the largest CV MSE (0.0078). For the lasso models, degree-2 polynomial model outperformed the linear model with a CV MSE of 0.002 compared to linear model's 0.006.

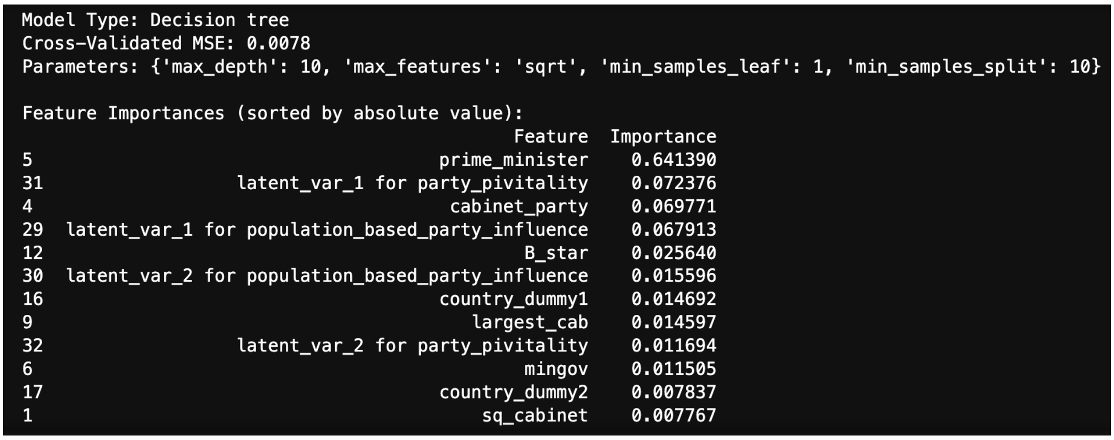
Predictor Importance Comparison

Predictor importance given by XGBoost:



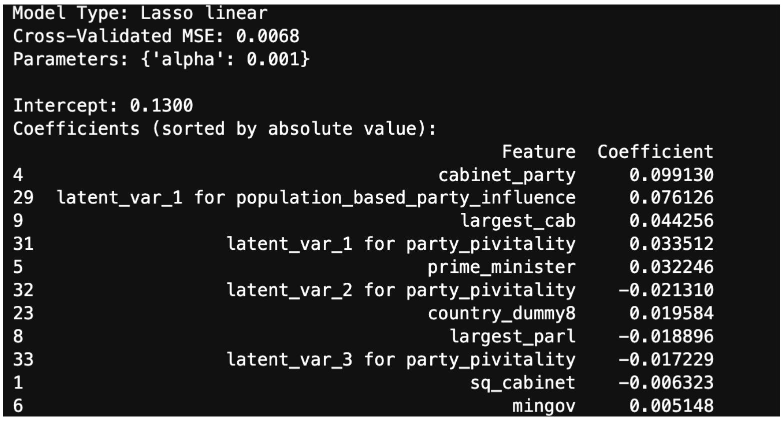
```
Model Type: Gradient boosting
Cross-Validated MSE: 0.0018
Parameters: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 100, 'subsample': 0.8}

Feature Importances (sorted by absolute value):
                                             Feature  Importance
9                                        largest_cab    0.335526
4                                      cabinet_party    0.321194
29  latent_var_1 for population_based_party_influence    0.182020
5                                     prime_minister    0.057671
31                  latent_var_1 for party_pivitality    0.052354
30  latent_var_2 for population_based_party_influence    0.014395
6                                             mingov    0.010890
32                  latent_var_2 for party_pivitality    0.009574
33                  latent_var_3 for party_pivitality    0.005767
19                                    country_dummy4    0.003152
0                                      election_year    0.002118
7                                          bicameral    0.000979
```

Predictor importance given by the single decision tree:



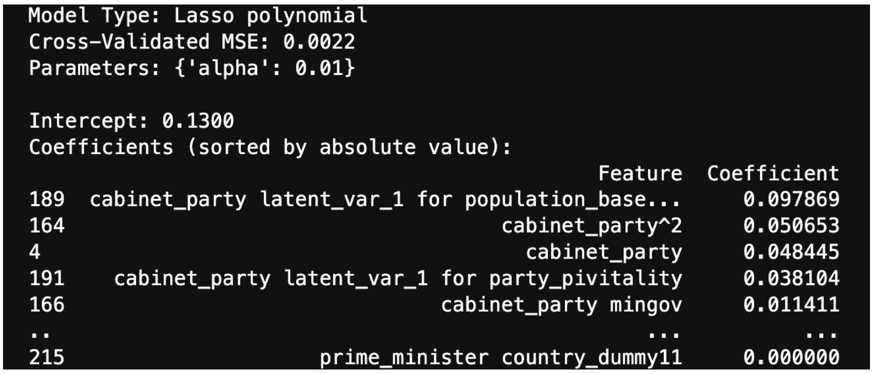
```
Model Type: Decision tree
Cross-Validated MSE: 0.0078
Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 10}

Feature Importances (sorted by absolute value):
                                             Feature  Importance
5                                     prime_minister    0.641390
31                  latent_var_1 for party_pivitality    0.072376
4                                      cabinet_party    0.069771
29  latent_var_1 for population_based_party_influence    0.067913
12                                            B_star    0.025640
30  latent_var_2 for population_based_party_influence    0.015596
16                                    country_dummy1    0.014692
9                                        largest_cab    0.014597
32                  latent_var_2 for party_pivitality    0.011694
6                                             mingov    0.011505
17                                    country_dummy2    0.007837
1                                         sq_cabinet    0.007767
```

Predictor importance given by lasso linear regression:



```
Model Type: Lasso linear
Cross-Validated MSE: 0.0068
Parameters: {'alpha': 0.001}

Intercept: 0.1300
Coefficients (sorted by absolute value):
                                             Feature  Coefficient
4                                      cabinet_party     0.099130
29  latent_var_1 for population_based_party_influence     0.076126
9                                        largest_cab     0.044256
31                  latent_var_1 for party_pivitality     0.033512
5                                     prime_minister     0.032246
32                  latent_var_2 for party_pivitality    -0.021310
23                                    country_dummy8     0.019584
8                                       largest_parl    -0.018896
33                  latent_var_3 for party_pivitality    -0.017229
1                                         sq_cabinet    -0.006323
6                                             mingov     0.005148
```

Predictor importance given by lasso degree-2 polynomial regression:



```
Model Type: Lasso polynomial
Cross-Validated MSE: 0.0022
Parameters: {'alpha': 0.01}

Intercept: 0.1300
Coefficients (sorted by absolute value):
                                              Feature  Coefficient
189  cabinet_party latent_var_1 for population_base...     0.097869
164                                    cabinet_party^2     0.050653
4                                        cabinet_party     0.048445
191    cabinet_party latent_var_1 for party_pivitality     0.038104
166                                cabinet_party mingov     0.011411
..                                                 ...          ...
215                       prime_minister country_dummy11     0.000000
```

Result Interpretation

The ranking of the predictors in the picture is determined by their significance. In all four models, the variable ‘cabinet_party’, which stands for whether the party has won the cabinet for the election, has the greatest importance. This is reasonable because if a party wins the cabinet, it will have the authority to select more of its members for the cabinet, thus increasing its cabinet proportion. This logic aligns with the positive coefficient of the variable ‘cabinet_party’ in the lasso linear regression. Another predictor that appeared in the top 3 of all models is the variable ‘latent_var_1 for population_based_party_influence’. This latent variable mainly represents the combination of the seat share of the party in the parliament and the number of ministries the party has in the government, and this might indicate that the larger population a party has in the parliament, the more seats it will be able to secure in the cabinet, which contributes to its proportion in the cabinet.

Overall, the importance of the predictor given by each model drops significantly after the 9th predictor. For example, the coefficient of the 9th important predictor given by the lasso linear model is 0.017, and this number plummeted to 0.006 at the 10th predictor. This cutoff seems to illustrate a very intriguing trend, as the predictors above the cutoff line seem to be indicators of how many members a party has in the election system and how pivotal it is in forming a majority party coalition. On the other hand, the predictors below the cutoff seem to relate more to the structure of the elective government, such as ‘mingov’ , ‘bicameral’, and the qualitative variable that categorizes how chaotic the parliament is. This could imply that the overall structure of the voting system might not vastly change the outcome of the voting and the coalitions formed.

Lasso linear model for reference:

0.1300 + 0.0991*(cabinet_party) + 0.0761*(latent_var_1 for population_based_party_influence) + 0.0443*(largest_cab) + 0.0335*(latent_var_1 for party_pivitality) + 0.0322*(prime_minister) - 0.0213*(latent_var_2 for party_pivitality) + 0.0196*(country_dummy8) - 0.0189*(largest_parl) - 0.0172*(latent_var_3 for party_pivitality)

10 terms