Teodor Ilie
Professor David Skillicorn
CISC251
November 20, 2020

Predicting Presidential Speech Efficacy

Introduction

The primary purpose of this project was to make a prediction about the efficacy of American presidential speeches, given a data set of 431 speeches made by American presidential candidates from 1992 to 2012. Each speech has attribute values corresponding to the frequency of the most frequently used 1000 words.

As well as trying to build a model that predicts electoral success using the most used words, we also attempted to prove or disprove the hypothesis that deception (deceptive words) can affect the effectiveness of a presidential speech, given that a large part of the population, especially a large quantity whose political opinions revolve around their impressions of a politician and not necessarily their political platform, may be susceptible to deceit.

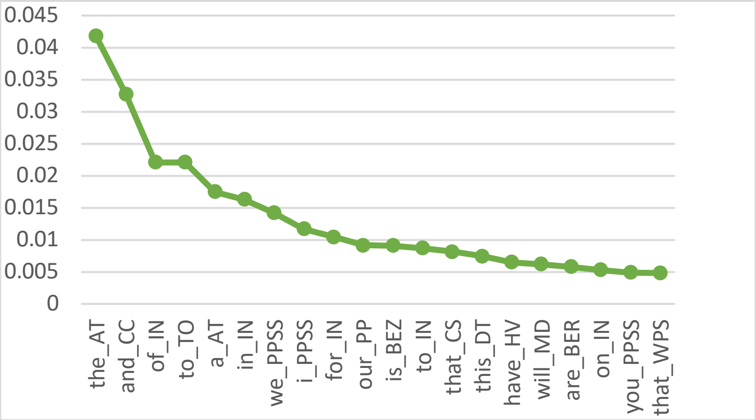
Finally, we concluded with the payoff: how could our analysis, theoretically, help future candidates craft more winning speeches? This took the form of some good and bad strategy ideas, and a summary of how much impact something as simple as word analysis may have.

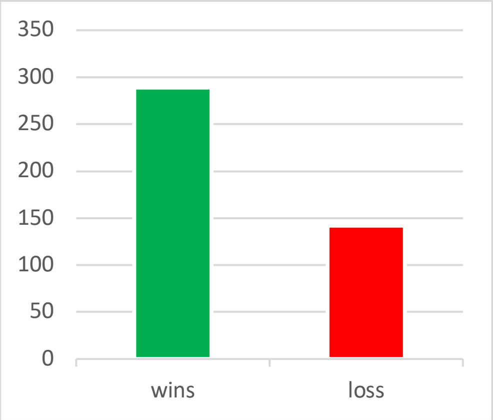
Initial Strategies and Statistical Analysis

The data had no missing values, but there were some problems with the column headers, including unrecognized characters, and some other characters which were throwing off the CSV processing in KNIME. Once these were removed, the data was formatted by adding the winning and losing column at the end, for prediction purposes, adding the 1000 words themselves as column headers, and adding the speeches as the first column.

An important note is that the data for the 1000 word frequencies was already normalized, because the entries represent the percentage of word use. The deception words data was not normalized, so normalization had to be applied to yeild useful results.

Some initial statistical analysis in KNIME, with Excel graphing, showed that there were 289 winning and 142 losing speeches, so the data set was fairly well balanced. The most commonly used words, and the means of their frequencies, were also plotted below:

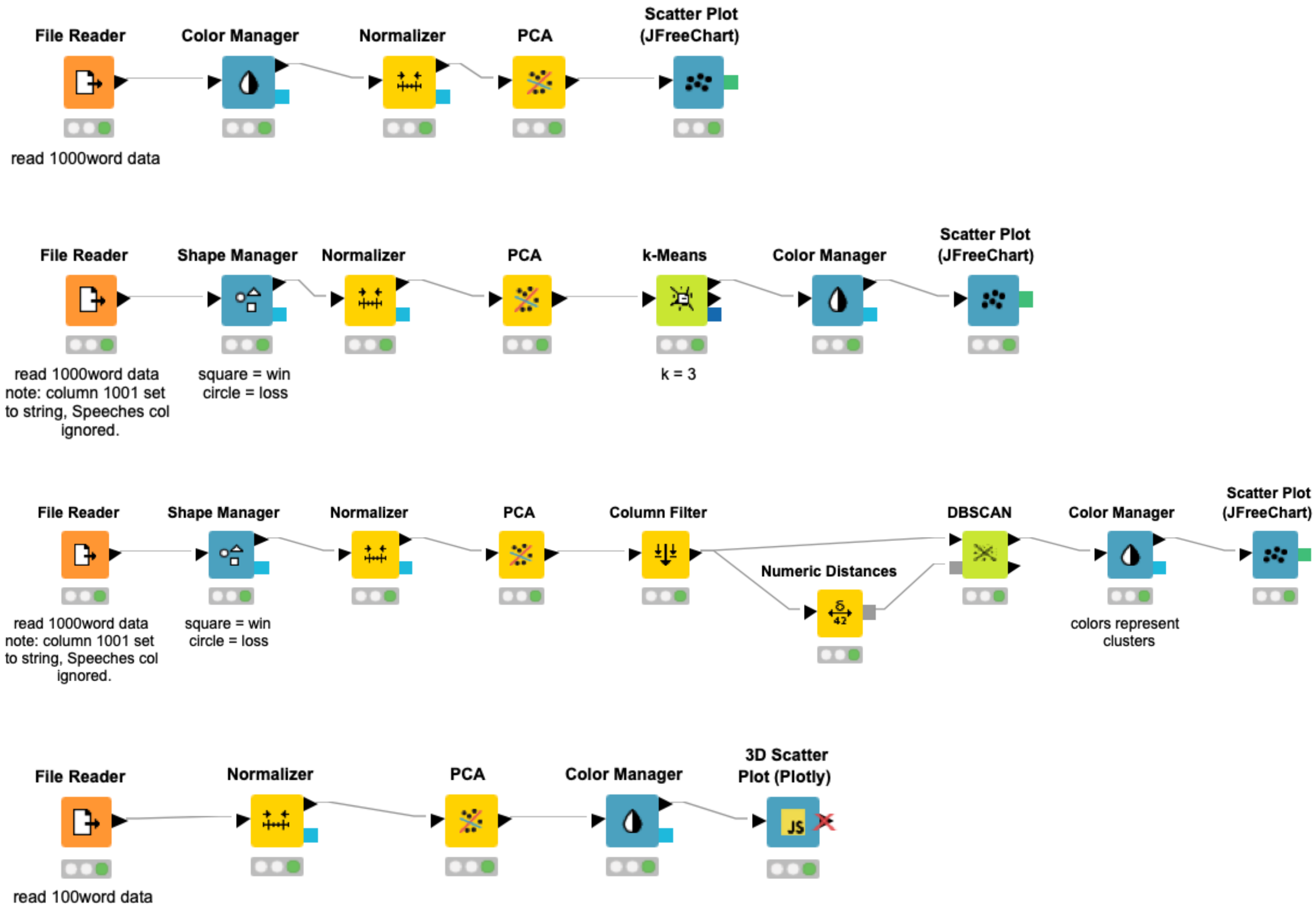




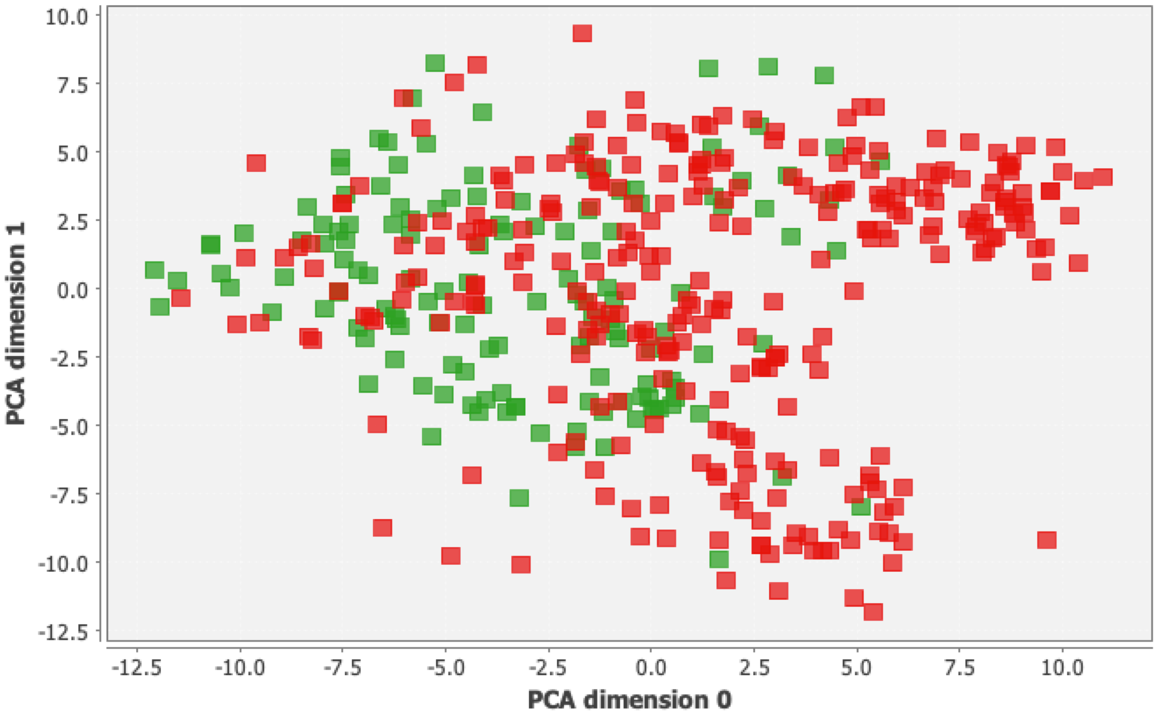
Clustering and Visualisation of 1000 Words and Deceptive Words

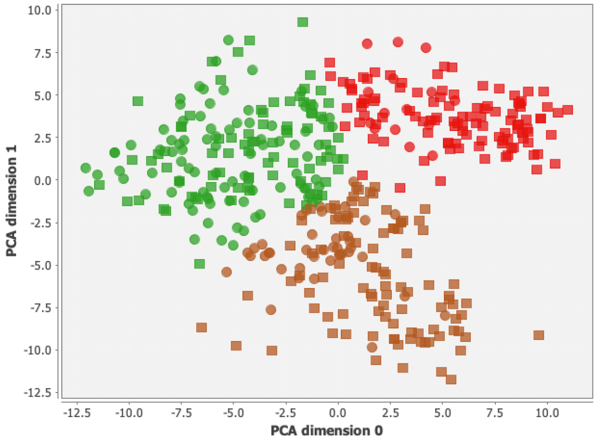
The first step was clustering and exploration. The issue that this posed was, having so many attributes, dimensionality reduction was necessary in order to make use of all the attributes. In order to do this, single value decomposition was used, specifically Principal Component Analysis, using the PCA node in KNIME. PCA is advantageous because it minimizes the loss of information while allowing us to see the data in a smaller dimension. A drawback, aside from some loss of information is that it generates new attributes which cannot be traced back to the source data.

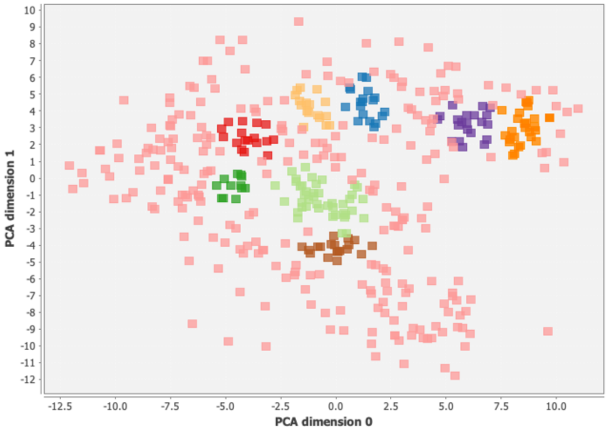
The following were the workflows built in KNIME for the 1000 word data set, with workflows top to bottom representing 2-D with PCA, 2-D PCA with k-means, 2-D PCA with DBSCAN, and 3-D PCA. Note normalization nodes were used because the partitions were stronger cut with it than without:


File Reader
Color Manager
Normalizer
PCA
Scatter Plot (JFreeChart)
read 1000word data
File Reader
Shape Manager
Normalizer
PCA
k-Means
Color Manager
Scatter Plot (JFreeChart)
read 1000word data note: column 1001 set to string, Speeches col ignored.
square = win circle = loss
k = 3
File Reader
Shape Manager
Normalizer
PCA
Column Filter
Numeric Distances
DBSCAN
Color Manager
Scatter Plot (JFreeChart)
read 1000word data note: column 1001 set to string, Speeches col ignored.
square = win circle = loss
colors represent clusters
File Reader
Normalizer
PCA
Color Manager
3D Scatter Plot (Plotly)
JS
read 100word data

The results of the first three are below, in order, top to bottom and left to right. Note that for PCA with k-means and DBSCAN, the point shape represents win/loss (square for win, circle for loss), and colours represent clusters given by the respective clustering algorithms. In the top graph of simple PCA, colours represent win/loss (red is win, green is loss):

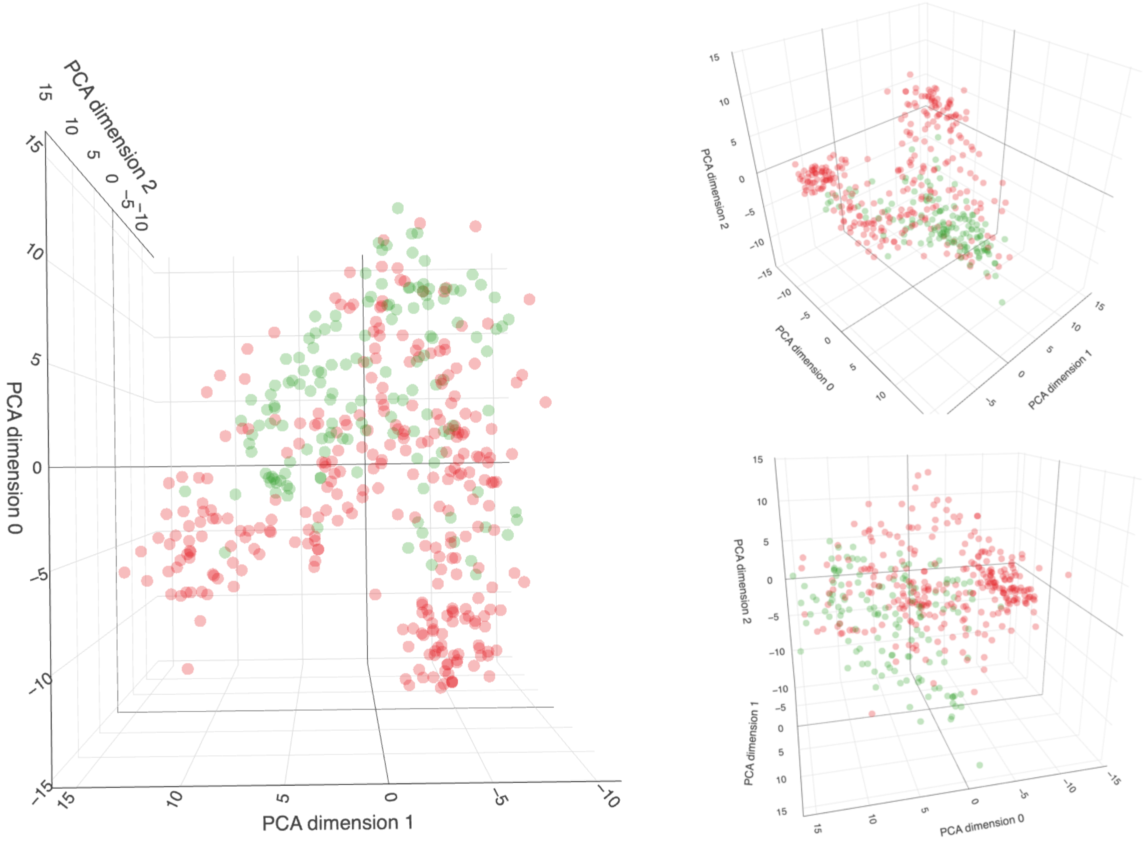






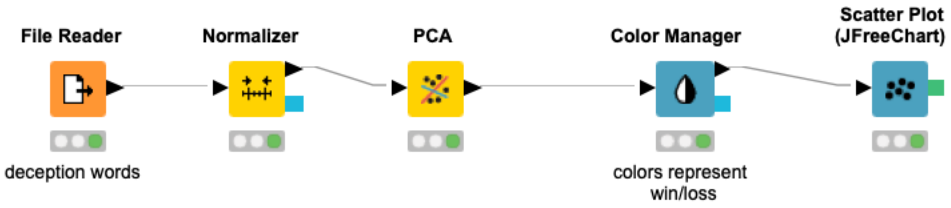
The top figure already seemed to show a partition happening between win, in red, and loss, in green, and more specifically, it also seemed to indicate that the green loss speeches clustered into one group, whereas the red winning speeches clustered into two distinct groups. Using k-means (bottom left) with k = 3, and DBSCAN (bottom right), we grouped the data above further to get some more idea of what is happening. While DBSCAN was a dead end, k-means showed us clearly how the points cluster quite well into three groups, where the green cluster is the main losing cluster, and the brown and red are the two winning clusters. Next we broadened our exploration to 3-D.

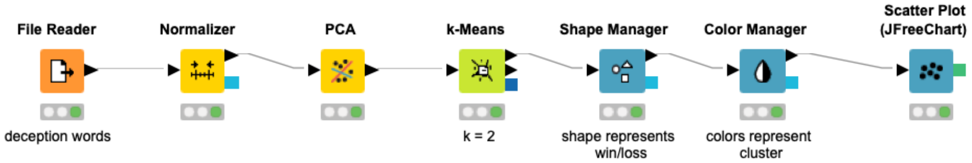
Adding another dimension, the following are three different views of the same 3-D scatter plot, again with PCA, where green and red represent win/loss. The added dimension allowed PCA to have less information loss, and so showed us a more accurate picture of the problem difficulty:

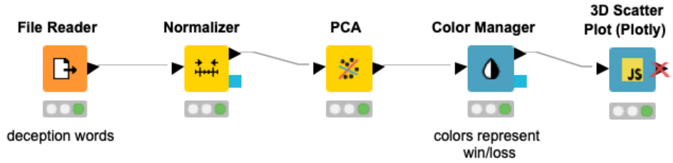


We can see the partition became stronger still. This suggested that the prediction problem had a good chance of success, and so going into our initial prediction modelling stages we were fairly confident in expecting some strong accuracy, given that 3 dimensions already yielded a fairly strong distinction between winning and losing speeches (and the true problem is 1000 dimensions). We saw the same emergent structure as from the 2-D scatter: green losses clustered in one area, whereas red winning speeches clustered in two different areas. Due to the nature of PCA it is hard to know what this means, but it does show that there are strong patterns in the data associated with winning and losing, and so strong predictive potential.

Next we moved on to a similar examination of the deceptive word file. As before, PCA was used for dimensionality reduction, and the following are workflows, in order, for PCA in 2-D, PCA in 2-D with k-means, and PCA in 3-D. Note DBSCAN was not used this time:

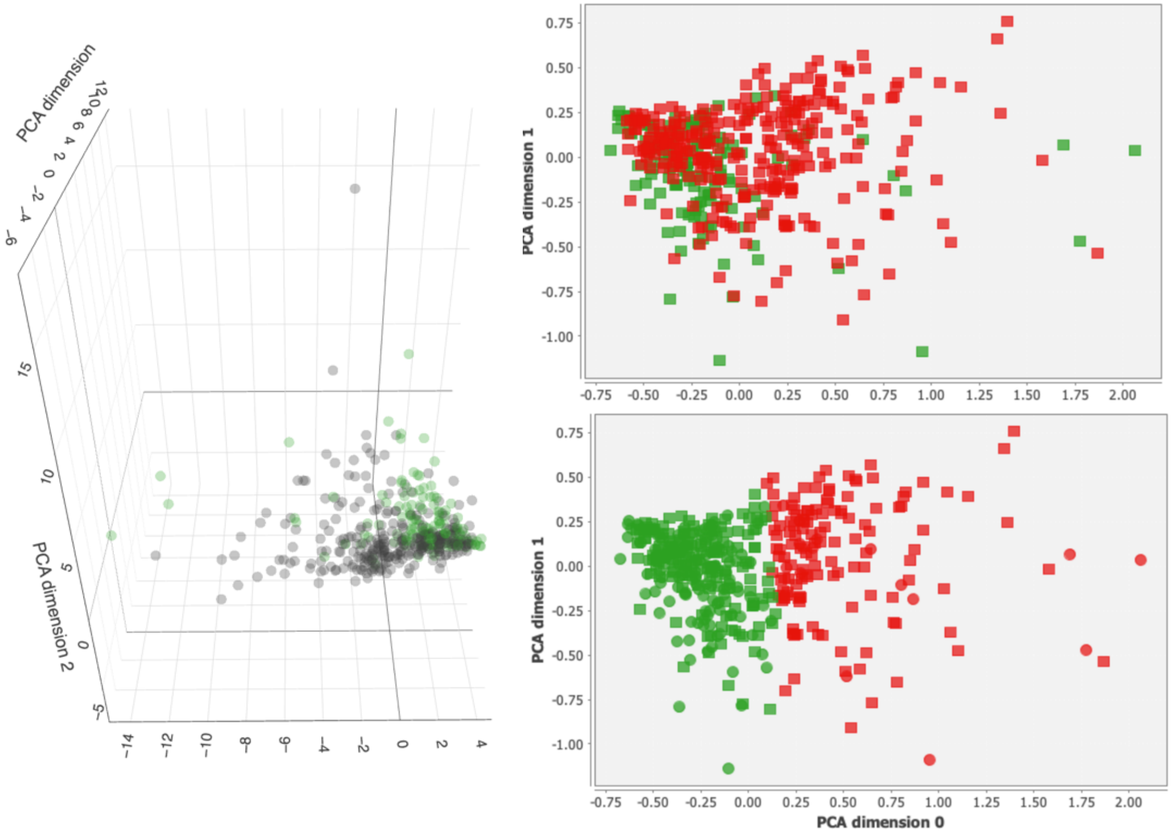






Below are the figures, where the left figure is the 3-D examination (with colours representing win/loss), top right is 2-D without clustering (colours represent win/loss), and bottom right is 2-D with k-means clustering in red/green and success in squares/circles:

We see that the partition of win/loss appears less distinct for the deceptive word file than it appeared previously for the 1000 word frequency data. This suggested that we could expect the deceptive word data to perform less accurately, in predicting speech success, than the 1000 word data. The top right figure, in particular, shows that wins and losses are very overlayed, and the partition in the bottom right with k-means appears much stronger than it really is, as both green and red clusters are full of circles and squares (wins and losses).



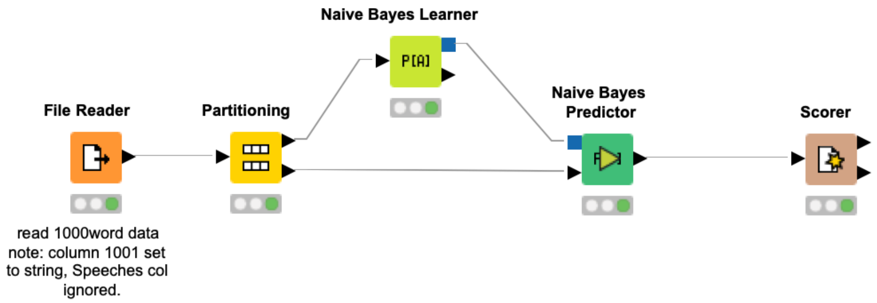
Initial Prediction Attempts on 1000 Words and Deceptive Words

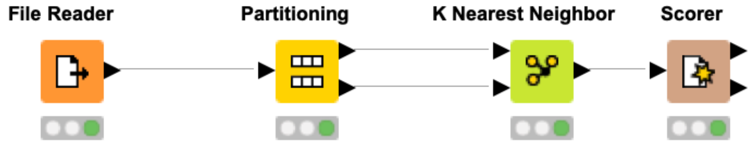
Having finished with clustering, and ascertained that the 1000 word data should be expected to perform fairly well as a predictor, and the deceptive word data somewhat poorer, we moved on to initial modelling. The models we had to choose from were Bayesian, k-Nearest Neighbour, Neural Networks, Random Forests, and Support Vector Machines. The main points that were kept in mind when deciding on a model to use for this problem were dataset size, the outcome size, and the relationships of the attributes with respect to one another.

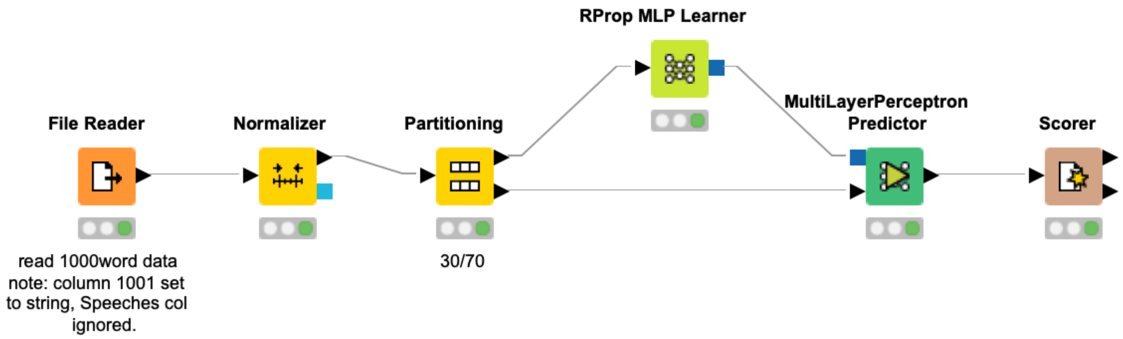
Bayesian predictors were probably not the best choice in this context due to their opacity, and k Nearest Neighbour also seemed less than ideal, but both were tried anyway to test their accuracy.

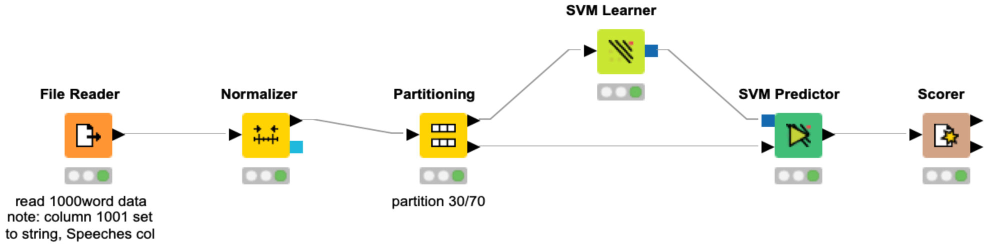
For our particular problem we expected Neural Nets to be quite accurate, as they take the sum total of information of all the attributes. This is what we wanted, as no particular attribute in itself is very helpful in predicting the outcome. The disadvantage of Neural Nets, as usual, is that they are opaque, so while they may turn out useful for judging a new speech (a new record) for its winning potential, they are not helpful in crafting a winning speech to begin with. The same went for SVM's – although potentially accurate, they are opaque as well.

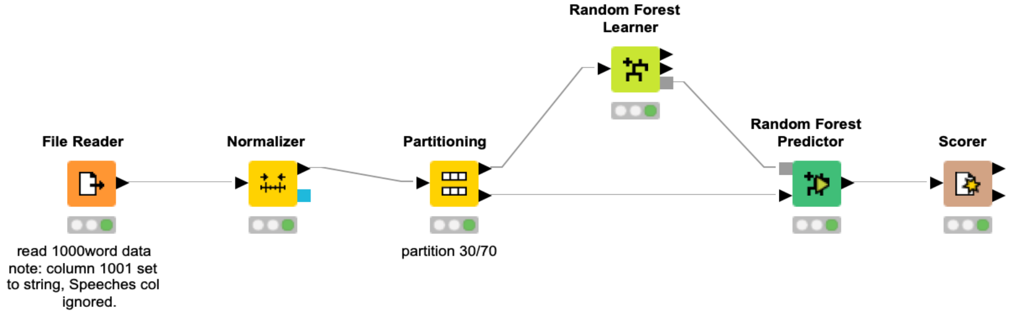
The most useful predictor in this case was the Random Forest model, due both to the fact that we expected an ensemble predictor to be strong in this case, and also because looking at the trees that make up the forest we could go in and gain some transparency as to why the model was making the choices it was, and how certain words were influencing the decisions. This would be useful in learning how to write winning speeches (more on this later). The following were the workflows that were used to test the various models:











All models yielded surprisingly accurate results. Below is a summary of the accuracy of the 5 models, where accuracy and balanced F_1-scores are reported. Accuracy is more useful for True Positives and True Negatives, while F-score is better for False Positives and False Negatives:

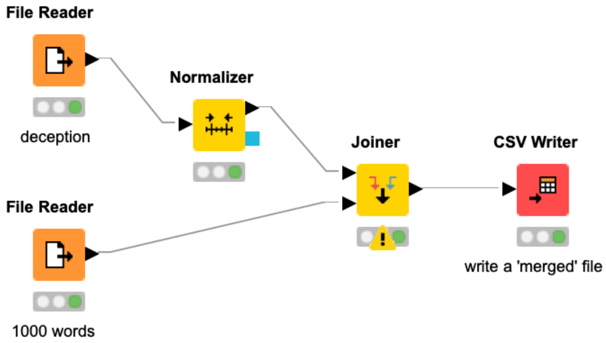
1000 Words	**Accuracy (%)**	**F_1-score**
Bayes	84.6	0.886
kNN	74.6	0.841
NN	86.9	0.903
SVM	93.1	0.948
RF	81.5	0.859

SVM outperformed every other model for the dataset, even Neural Nets (partly because KNIME Neural Nets are not as advanced). Nonetheless, Random Forest remained the most useful predictor due to its transparency, and 81.5% accuracy was still a very strong accuracy. Next, we deployed the same models, this time on the deceptive word data. The workflows were the same as for the 1000 word frequencies, so only the results are shown:

Deceptive	**Accuracy (%)**	**F_1-score**
Bayes	33.8	0.023
kNN	70.0	0.791
NN	69.2	0.772
SVM	72.3	0.798
RF	71.5	0.781

While the SVM model still performed the best, the deceptive word data was not as accurate in predicting a speech outcome as the 1000 word data, and Bayes in particular was extremely weak. This was also in part due to the fact that there are far less than 1000 deceptive words given, and less data means less for the models to use in prediction, but these results were in line with what we expected given our initial clustering: the 1000 words data is a better predictor than the deceptive words data.

Combined Models, and Final Models for Most Frequent 1000 Words



The next step in the process was to combine the 1000 word data with the deceptive word data and see if this built a stronger predictor than either of the two on their own. To the right was the workflow used in KNIME, where the error from the Joiner node is due to the fact that there is overlap in the words in the two datasets. For such words, only one of the data columns was kept because otherwise that word attribute would have a disproportionate effect on the model. Next, the three strongest models were applied to the merged data: Neural Net, Support Vector Machine, and Random Forest, again using workflows identical to the ones used the initial prediction attempts. The results were as follows:

Combined	**Accuracy (%)**	**F_1-score**
NN	87.7	0.917
SVM	92.3	0.938
RF	84.6	0.884

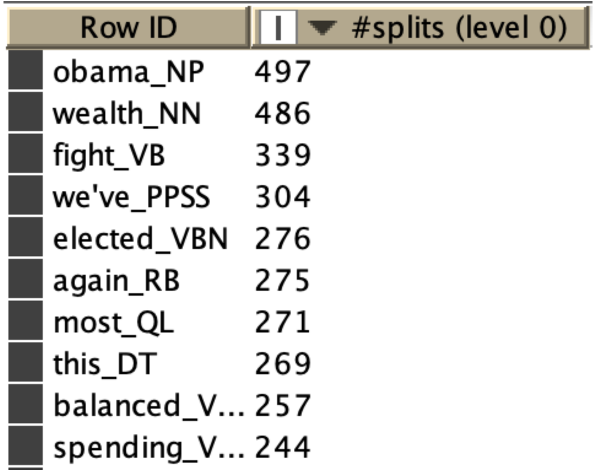
Surprisingly, the results showed that, while SVM accuracy decreased slightly with respect to 1000 words alone, going from 93.1% to 92.3%, the other two models scored higher accuracy with the merged data set than with the 1000 words alone: going from 86.9% to 87.7% with Neural Net, and 81.5% to 84.6% with Random Forest. This showed that, although deceptive words alone were not as accurate in predicting winning as the 1000 words alone, the combined data set was more accurate than either alone. Nonetheless, the highest accuracy obtained thus far remained SVM on 1000 words alone, with accuracy 93.1% and F-score 0.948.

The next step moving forward was to take the 3 strongest models: SVM on 1000 words, NN on the combined and RF on the combined, and tweak the parameters to get stronger accuracies. For SVM, different Kernel functions and C values were tried, and the result of the trials found Polynomial Kernel with C = 0.2 to be the best, where Polynomial Kernel and C = 1 is the default. For NN, the best result was found with 1000 iterations instead of the default 100, and while more may have been better still, 1000 already took some time to run. Finally, for RF, 10,000 trees rather than the default 100 yielded stronger accuracy, but again, this already took a significant amount of time to run, so larger values were not tried, although they may have given marginally better accuracy results.

Best Models	**Accuracy (%)**	**F_1-score**
SVM – 1000 words	94.6	0.960
NN – Combined	90.7	0.938
RF – Combined	86.9	0.902

Of these models, SVM and NN were still strongest, but they have the disadvantage of opacity. RF on the combined data set therefore remained the most useful model, and in the next section we will discuss how its transparency was used to determine winning words.

Words that Predict Success



Row ID	#splits (level 0)
obama_NP	497
wealth_NN	486
fight_VB	339
we've_PPSS	304
elected_VBN	276
again_RB	275
most_QL	271
this_DT	269
balanced_V...	257
spending_V...	244

Using the *Attribute Statistics* option of the KNIME Random Forest Learner node, we sorted the combined data set by attributes (words) most often at the top of tree models (level 0). Due to the nature of building decision trees, these words are, by definition, the strongest single attribute predictors of winning or losing a speech, however they are not all necessarily useful words to use. Some are advisable to use and some are to be avoided. By going through the trees, we judged which words are good predictors of success above some frequency threshold, and which are predictors of failure above a threshold.

Of these 10 top words, all are indicators of losing speeches, except for *we've* and *this*. In fact the top three words – *Obama, wealth,* and *fight* – all are indicators of losing. This makes sense because Obama served two consecutive terms in office, and it would have been his opponents using his name, both of whom obviously lost. *Wealth* and *spending* are also loss predictors, which makes sense, as Americans don't trust their government very much with their money, and often therefore prefer not pay as many taxes as, for example, Europeans. As for *fight, elected, again, most,* and *balanced,* the reason for their negative influence is less clear.

Because it is more useful often to focus on the things to do, rather than avoiding those not to do, when writing a speech, some further words which were strong predictors of success rather than failure were *laughter, country, that's* and *we. Country* makes sense, as Americans are often patriotic, *laughter* because it brings joy to listeners, and *that's* and *we've* were likely used in the context of encouraging and uplifting promises of a better future together, a good strategy in an electoral speech.

Deception: A Winning Strategy?

Having assessed the performance of the 1000 word and deceptive word data sets individually and combined, the next order of business was deciding if deception is a winning strategy as far as electoral speeches are concerned. We propose that deception remains a sub-optimal strategy, as the strongest model remained the SVM on the 1000 word data alone. Nonetheless, the results were indeed stronger for Neural Net and Random Forest models when combined with the deceptive word data, so these words do have an influence on a speech's outcome to an extent.

Conclusion

To conclude, the predictive power of word frequency far exceeded expectations, as the largest accuracy achieved, 94.6%, is very high indeed. Words that were useful in speeches included *we've, this, laughter, country, that's,* and *we*, all of which suggest that speeches that champion unity, positivity, and patriotism are more likely to be successful in American elections.