# Government Agency Creation: A Machine Learning Case Study

Guy Freedman

October 19, 2020

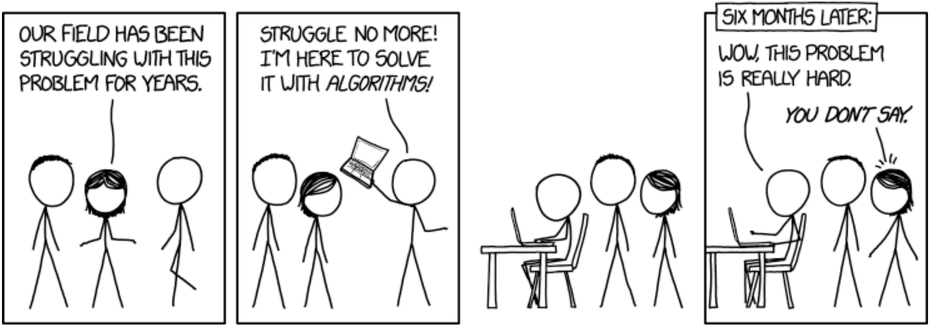
# The Problem

How to identify observations that relate to the creation of government agencies in the old congressional hearings dataset?

- 1868 – 1946 (40th–79th Congress).
- N = 30,551

# Proposed Solution

Supervised machine learning!

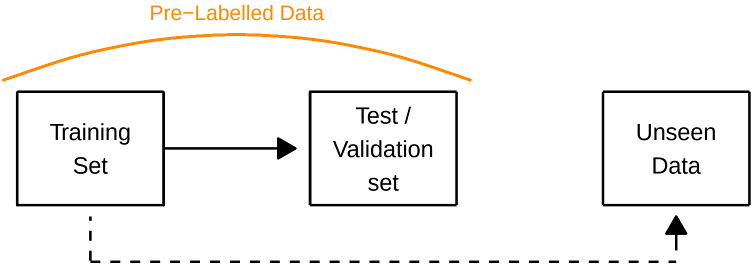


# Proposed Solution

Supervised machine learning!

- Use existing data that have already been manually coded for agency-creation.
- The algorithm identifies patterns that maximize accuracy of prediction and can be applied to unseen data.
- Congressional hearings 1946 – 2017 (79th–115th Congress).
- N = 100,254 (91,872 are useable).

# Supervised Machine Learning



## Spoiler Alert: Results

- **Narrowed down results:**
- High probability agency-creation hearings: N = 1,225.
  - Next: Manually review.
- Low probability agency-creation hearings: N = 3,130.
  - Next: Manually review a random sample (N = 1,000).
- Not likely agency-creation hearings: N = 26,191.
  - Next: Manually review a random sample (N = 1,000).
- **Bottom line: Manually review 3,225 observations instead of 30,551 (10.6%).**

# § Important Differences between Inference and Prediction §

- Goal: Predict with high accuracy; less important to confirm theoretical argument through statistical inference.
- Independent variables and/or effects may be nonsensical or difficult to interepret ...
    - because they're based on some mathematical transformation.
    - because they have lots of categories.
    - because they involve high-order interactions.

But we're not interesting in interpreting their effects!

# § Important Differences between Inference and Prediction §

- Goal: Predict with high accuracy; less important to confirm theoretical argument through statistical inference.
- To some extent, willing to violate some assumptions (like multicollinearity) ...
  - if it improves prediction.
  - because biases in variance and lack of interpretation are less of a concern.

But some assumptions are very important (like the consequences of measurement error) - if violating them hurt our prediction and make it hard to generalize to unseen data.

# § Important Differences between Inference and Prediction §

- Goal: Predict with high accuracy; less important to confirm theoretical argument through statistical inference.
- Much like in inference, overfitting the model to our training set is the biggest concern.
  - In the entire process, we're constantly trying to balance optimizing our prediction based on the training set, with avoiding overfitting the model to our training set.

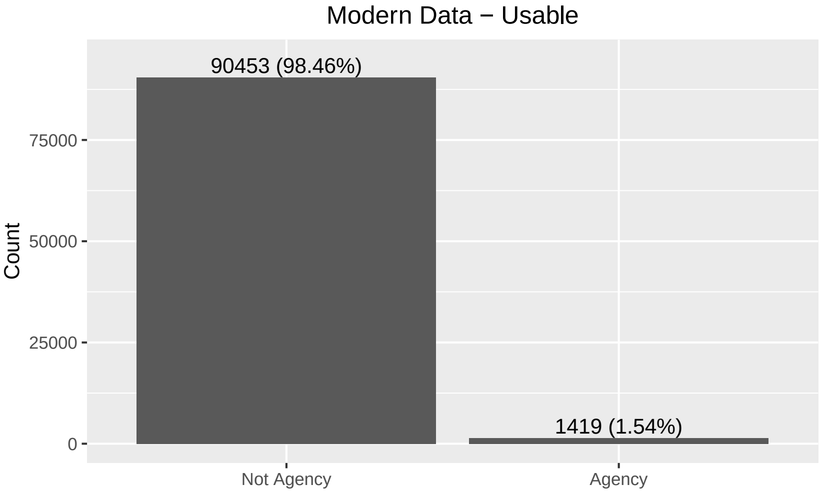
# Things the Human User Needs to Decide

**(or: Things that you have control over and affect the performance of the model)**

1. Figure out what your population data are.
2. Test set – size, structure, ratio between categories of DV, observations to include/exclude, population to sample from.
3. Training set – size, structure, ratio between categories of DV, observations to include/exclude, population to sample from.
4. The model – the algorithm, the features, resampling methods, tuning parameters.
5. Assessing performance – choosing a final training set, a final model, a threshold for prediction and apply to the test set.
6. Apply to unseen data.

# Population Data

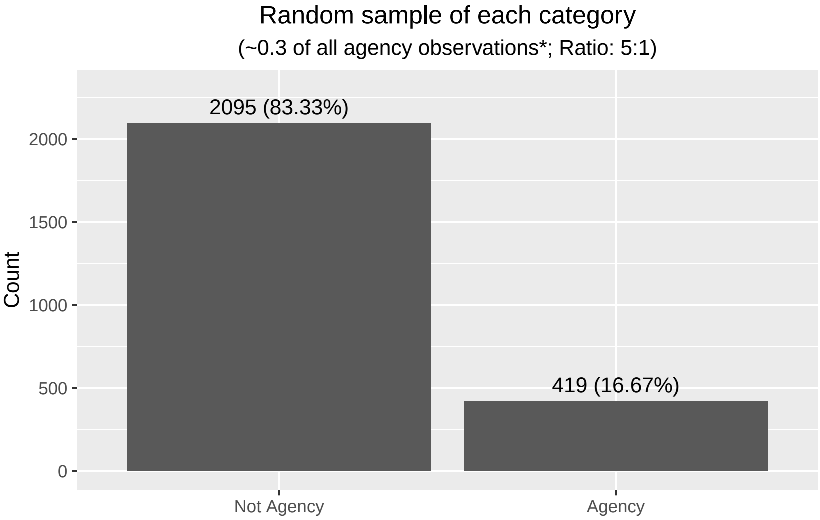
# Population Data




8382 missing / insufficient text.

# Test Set

# Test Set


Random sample of each category
(~0.3 of all agency observations*; Ratio: 5:1)
2095 (83.33%)
419 (16.67%)
Count
2000
1500
1000
500
0
Not Agency
Agency

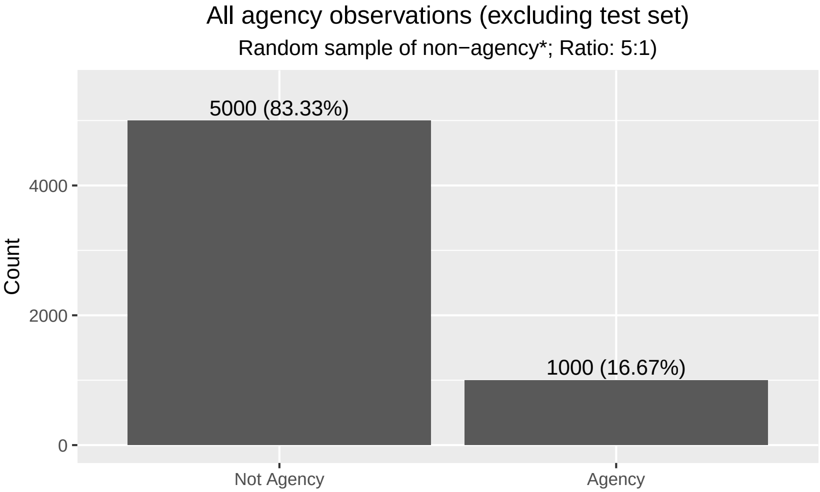
# Now leave your test set alone!

# Training set

# Training set

- You can play all you want with your training set, but DO NOT in any way let your test set inform the algorithm or your training set.
- Training set = a sample of your population data after excluding your test set.
- Try different sample sizes, different ratios between categories, include/exclude specific observations.

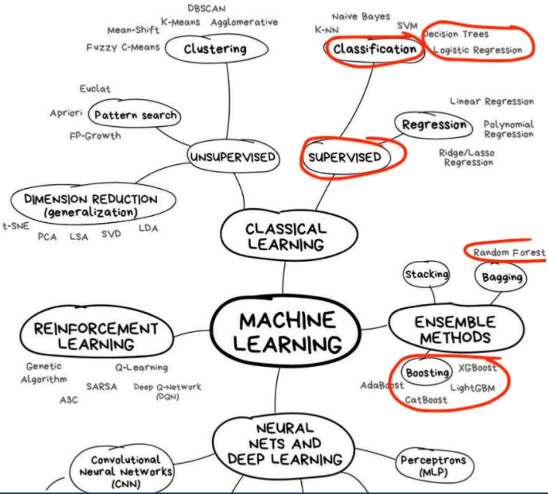
# Training Set



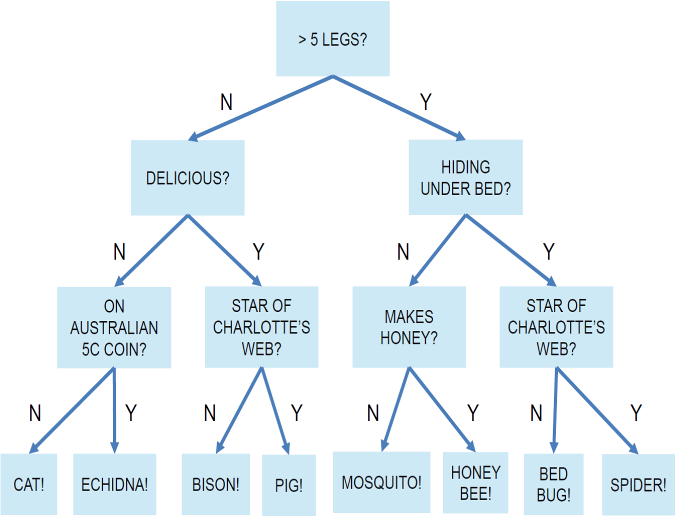

*Performed best on training set; also tried 1:1, 2:1, 9:1.

# The Model

# Choosing an approriate algorithm



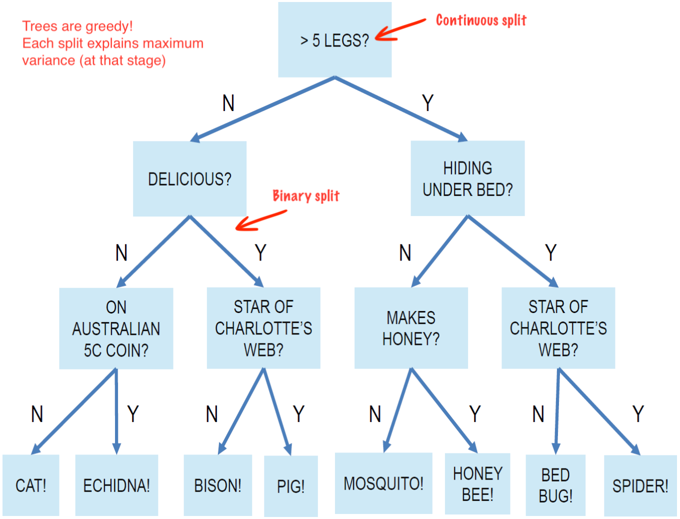
# Trees




https://www.acadian-asset.com/viewpoints/machine-learning-in-quant-investing-revolution-or-evolution

# Trees

Trees are greedy!
Each split explains maximum variance (at that stage)




https://www.acadian-asset.com/viewpoints/machine-learning-in-quant-investing-revolution-or-evolution

# Choosing an approriate algorithm

- DV: Categorical (dichotomous) = classification problem.
- Popular classification algorithms:
  - Logistic regression (terrible with imbalanced data)
  - Decision trees (is one enough?)
  - Random forest (lots of trees! Bagging method, underperforms with imbalanced data, can handle multiple categories in DV)
  - GBM — Gradiant boosting (lots of trees! Boosting method = each tree tries to correct the errors of the previous tree, better with imbalanced data)
  - catboost — an improvement on GBM, can have categorical predictors (and not just binary/numeric)
- Sometimes worth trying different ones on your training set.

# Model Features

- Independent variables are often termed features.
- Features often make theoretical sense, but may be difficult to interpret (complicated calculations, high order interactions, etc.)
- Features do not have to be causal; they have to be strong predictors, maximizing explained variance.
- In our example:
  - Text features — words/terms from the hearings' description.
  - Metadata — things we know about the hearing that is not expressed in the text.

## § Text Features – Pre-processing §

- Remove stop words — common words like a, the, etc.
- Removing numbers, punctuation, whitespace — keep only letters.
- Stemming is often (but not always!) recommended (econom instead of economy/economic) — groups together different variants of the same word.
- Removing sparse terms — crucial to prevent overfitting to the training data. Can try different sparsity levels to find optimal level.
  - In our data, keeping only .999 most frequent terms reduced the number of terms from 14,447 to 1,070.
  - Each term is a feature, i.e. a predictor in a statistical model!

# § Example Code §

```
# import data
hearings <- read_csv("https://comparativeagendas.s3.amazonaws.com/datas

# remove missing cases
hearings <- filter(hearings, filter_Agency %in% c(0,1))

# create unique id
hearings$myid <- 1:nrow(hearings)

# pre-process data
fulldtm <- as.data.frame(hearings) %>%
  filter(grepl("[a-z]",description, ignore.case = T)) %>%
  unnest_tokens(output = word, input = description) %>%
  filter(!str_detect(word, "^[0-9]*$")) %>% # remove numbers
  anti_join(stop_words) %>% # remove stop words
  mutate(word = SnowballC::wordStem(word)) # stem the words
```

# § Example Code §

```
# create documen-term-matrix
fulldtm <- fulldtm %>%
  count(myid, word) %>% # count of each word in each observation
  cast_dtm(document = myid, term = word, value = n) # no weights
```

# § Example Code §

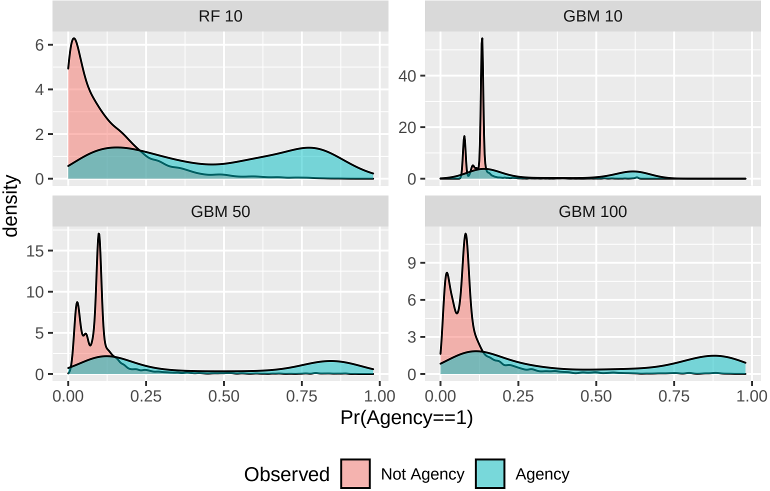
```
mygbm50 <- train(x = as.matrix(mytrain), # training set
                  y = factor(trainingset2$filter_Agency, # DV
                             levels = c(0,1),
                             labels = c("NotAgency","Agency")),
                  method = "gbm", # use "ranger" for RF
                 # resampling:
                 trControl = trainControl(method = "repeatedcv",
                                          number = 10,
                                          repeats = 3,
                                          classProbs = T,
                                          savePredictions = T),
                 # tuning parameters:
                 tuneGrid = data.frame(n.trees = 50,
                                       n.minobsinnode = 2,
                                       interaction.depth = 10,
                                       shrinkage = .1))
```

# § Tuning Parameters §

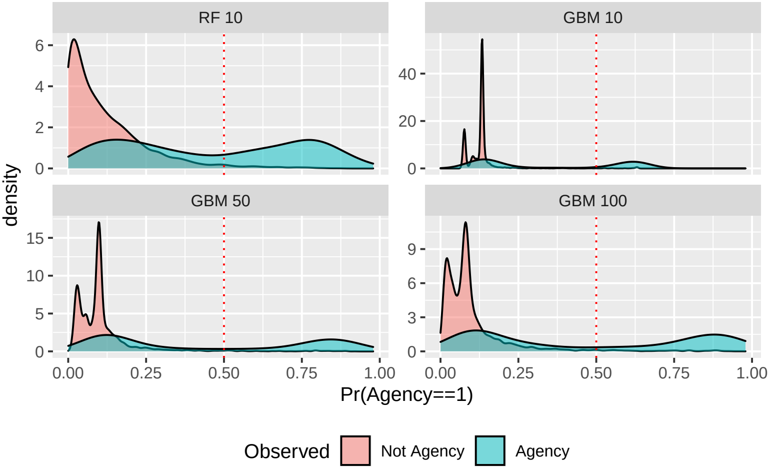
- Usually best to play with tuning parameters after finalizing your training set and choosing your algorithm.
- Tuning parameters improve your predictions slightly — a good model will become a better model; a bad model will still be pretty bad.
- Different models have different parameters; common ones include number of trees, minimum number of observations and interaction depth.

# Assessing Performance

# Predicted Probabilities: Training Set

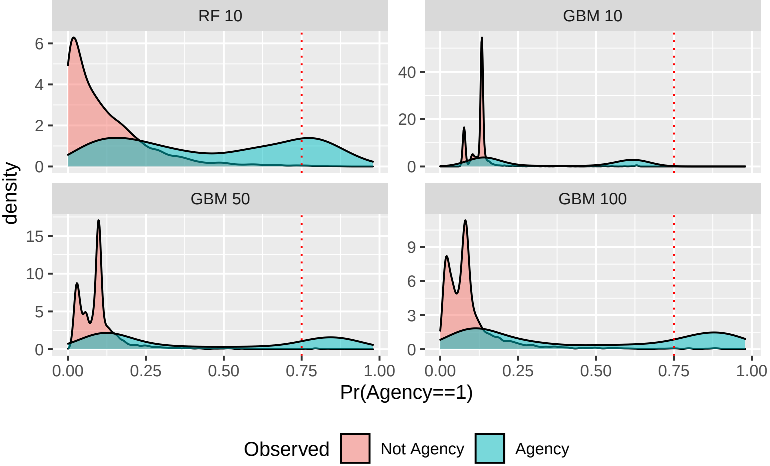


# Choosing a Threshold for Prediction



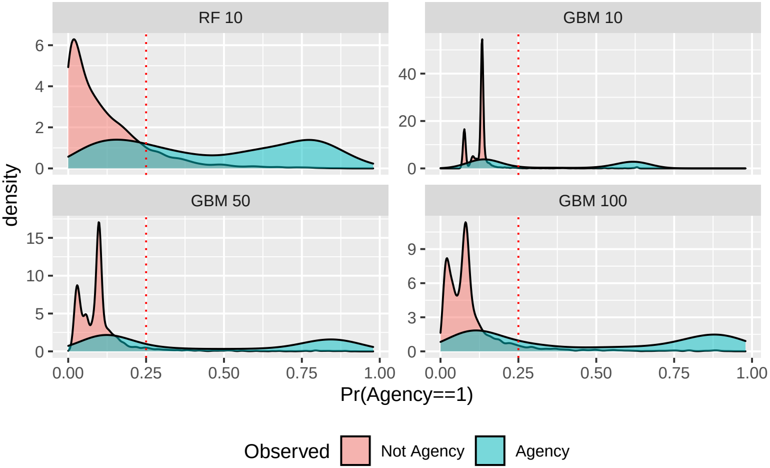

Pr(Agency)==.5 ?

# Choosing a Threshold for Prediction



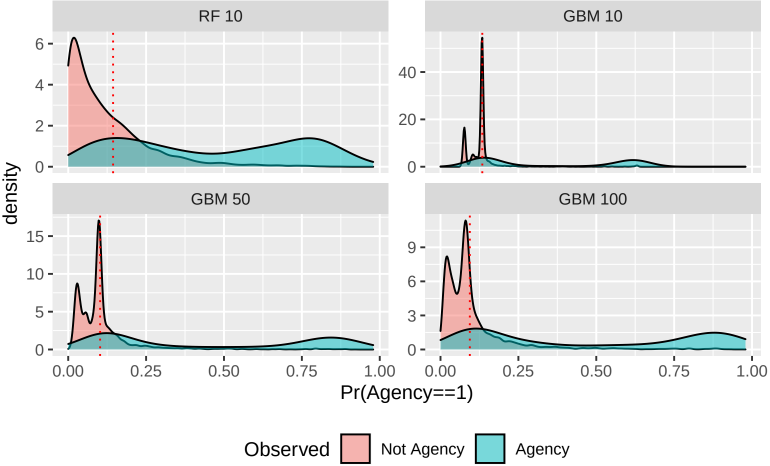

Pr(Agency)==.75 ?

# Choosing a Threshold for Prediction




Pr(Agency)==.25 ?

# Choosing a Threshold for Prediction

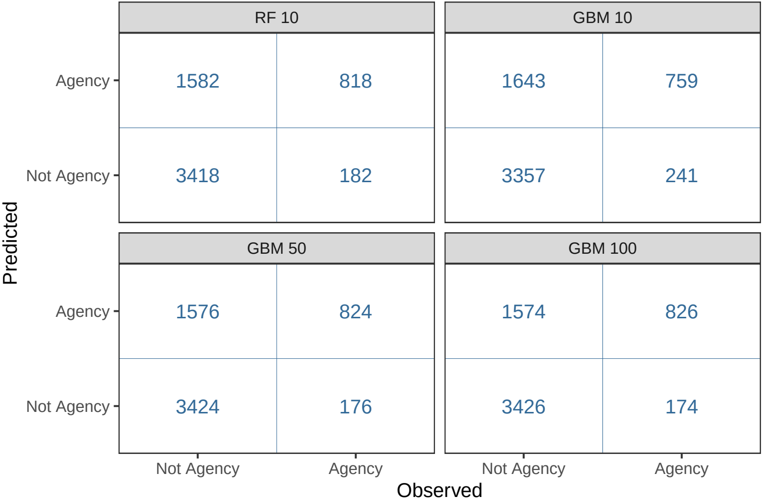



Pr(Agency)== Qauntile .6 (60%) ?

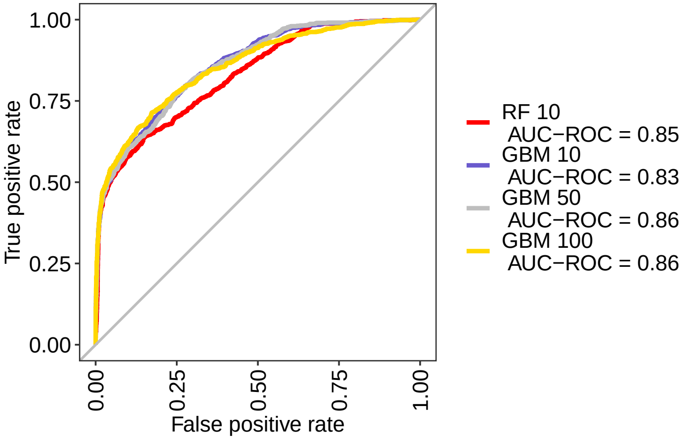
# § Performance Metrics §

- Accuracy
- Precision
- Specificity, TRUE-NEGATIVE-RATE
- Sensitivity, TRUE-POSITIVE-RATE, Recall

# Confusion Matrix (quantile >= .6)



# ROC Curve and AUC



# Adding Non-Textual Features

- House
- Joint
- subtopic_count_scaled
- prop_subtopic
- Party control:
  - SamePartyAll
  - SamePartyHouseSenate
  - SamePartyHousePresident
  - SamePartySenatePresident
- DW:
  - avg_dw
  - avg_dw_chamber
  - avg_dw_dem
  - avg_dw_rep
  - avg_dw_dem_chamber
  - avg_dw_rep_chamber

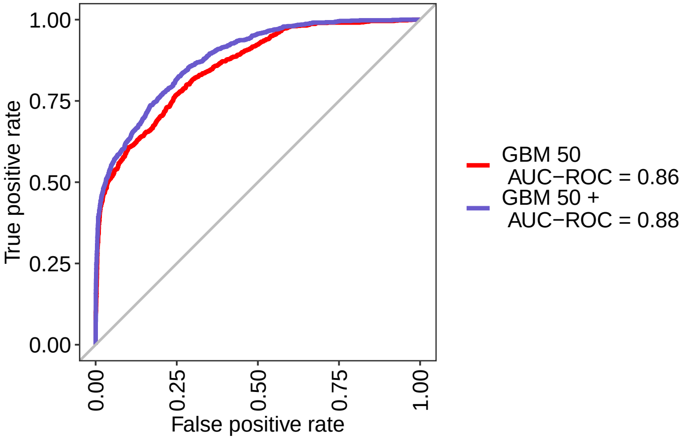
# Features: Illustration

| | econom | employ | feder | inflat | insur |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 0 |
| 4 | 1 | 0 | 0 | 0 | 0 |
| 5 | 1 | 0 | 0 | 0 | 0 |
| 6 | 1 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 1 | 1 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 |
| 9 | 2 | 0 | 0 | 0 | 0 |
| 10 | 2 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 |
| 12 | 1 | 0 | 0 | 1 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 |
| 14 | 1 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 |
| 16 | 1 | 0 | 0 | 0 | 0 |
| 17 | 1 | 1 | 0 | 0 | 0 |
| 18 | 1 | 0 | 0 | 0 | 0 |

# Features: Illustration

| | econom | employ | feder | inflat | insur | House | prop_subtopic | SamePartyAll | avg_dw_rep | avg_dw_cham |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0070103093 | 0 | 0.2539383 | -0.056156863 |
| 2 | 1 | 0 | 0 | 0 | 0 | 1 | 0.0101098901 | 0 | 0.2502565 | -0.075153153 |
| 3 | 1 | 0 | 0 | 0 | 0 | 1 | 0.0123924269 | 0 | 0.2616513 | -0.065850679 |
| 4 | 1 | 0 | 0 | 0 | 0 | 1 | 0.0123924269 | 0 | 0.2616513 | -0.065850679 |
| 5 | 1 | 0 | 0 | 0 | 0 | 0 | 0.0123924269 | 0 | 0.2616513 | -0.065625000 |
| 6 | 1 | 0 | 0 | 0 | 0 | 0 | 0.0123924269 | 0 | 0.2616513 | -0.065625000 |
| 7 | 0 | 0 | 1 | 1 | 0 | 0 | 0.0123924269 | 0 | 0.2616513 | -0.064647059 |
| 8 | 0 | 0 | 0 | 0 | 0 | 1 | 0.0144014401 | 0 | 0.2661351 | -0.121886621 |
| 9 | 2 | 0 | 0 | 0 | 0 | 0 | 0.0144014401 | 0 | 0.2661351 | -0.115621771 |
| 10 | 2 | 0 | 0 | 0 | 0 | 0 | 0.0129440925 | 1 | 0.2674595 | -0.112181651 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0084724005 | 1 | 0.2910297 | -0.081489871 |
| 12 | 1 | 0 | 0 | 1 | 0 | 0 | 0.0084724005 | 1 | 0.2910297 | -0.081489871 |
| 13 | 0 | 0 | 0 | 0 | 0 | 1 | 0.0116556291 | 0 | 0.3072680 | -0.032970721 |
| 14 | 1 | 0 | 0 | 0 | 0 | 0 | 0.0116556291 | 0 | 0.3072680 | 0.030455446 |
| 15 | 0 | 0 | 0 | 0 | 0 | 1 | 0.0116556291 | 0 | 0.3072680 | -0.032970721 |
| 16 | 1 | 0 | 0 | 0 | 0 | 0 | 0.0116556291 | 0 | 0.3072680 | -0.021216514 |
| 17 | 1 | 1 | 0 | 0 | 0 | 1 | 0.0068662455 | 0 | 0.3210000 | -0.063120729 |
| 18 | 1 | 0 | 0 | 0 | 0 | 1 | 0.0068662455 | 0 | 0.3210000 | -0.063120729 |

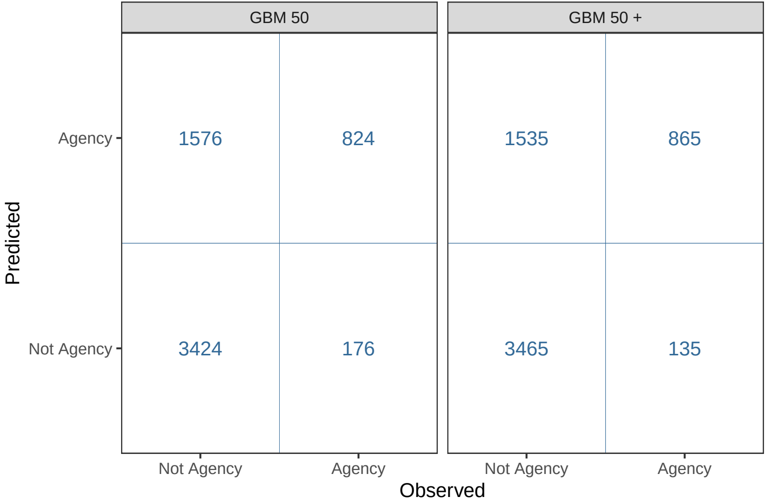
# Adding Non-Textual Features



# Variable Importance (Top 20)

```
##    Word (GBM 50)  Variable (GBM 50 +)
## 1      establish            establish
## 2         examin               avg_dw
## 3       creation           avg_dw_dem
## 4          creat             creation
## 5        reorgan           avg_dw_rep
## 6        commiss              commiss
## 7         review                creat
## 8            h.r              reorgan
## 9        mainten        prop_subtopic
## 10       coordin   avg_dw_rep_chamber
## 11        indian               improv
## 12          park          SamePartyAll
## 13         ethic               indian
## 14       program              mainten
## 15          hear             research
## 16        center subtopic_count_scaled
## 17       preserv             advisori
## 18        improv                ethic
## 19        nation                 park
## 20         nomin                addit
```

# Confusion Matrix (quantile >= .6)



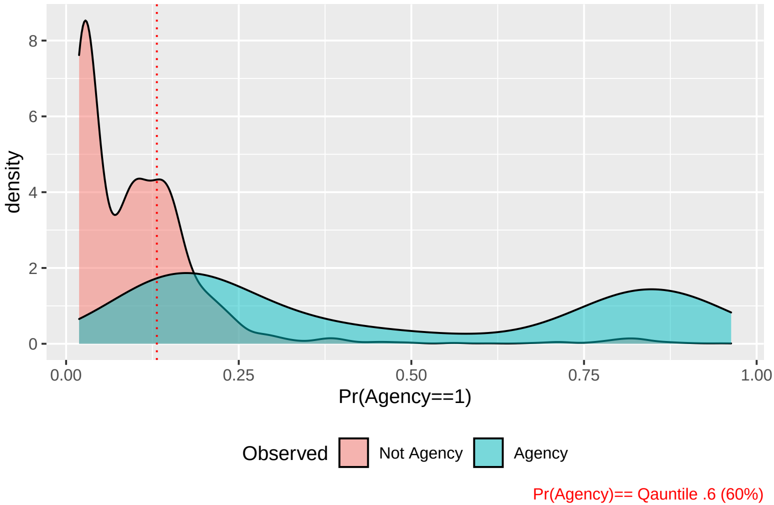
Finally, back to our test set!

# § Applying our model to our test set §

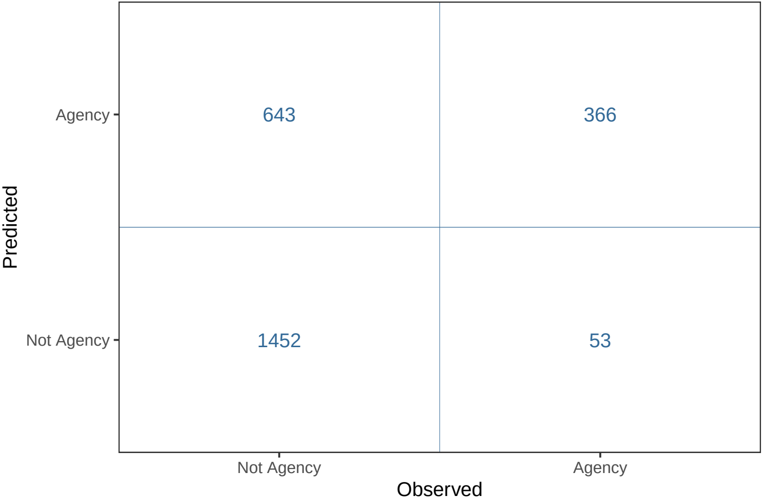
```
testset_predictions <- predict.train(mygbm50_nontext, # trained model
                                     as.matrix(mytest_nontext), # dtm of tes
                                     "prob") %>% # return class probabilitie
  mutate(Observed = factor(testset2$filter_Agency,
         levels = c(0,1),
         labels = c("Not Agency","Agency")))
```

* We can apply several models (from several training sets) to the same test set to compare.

# Applying our model to our test set



# Test Set: Confusion Matrix (quantile >= .6)



# And now, for our unseen data

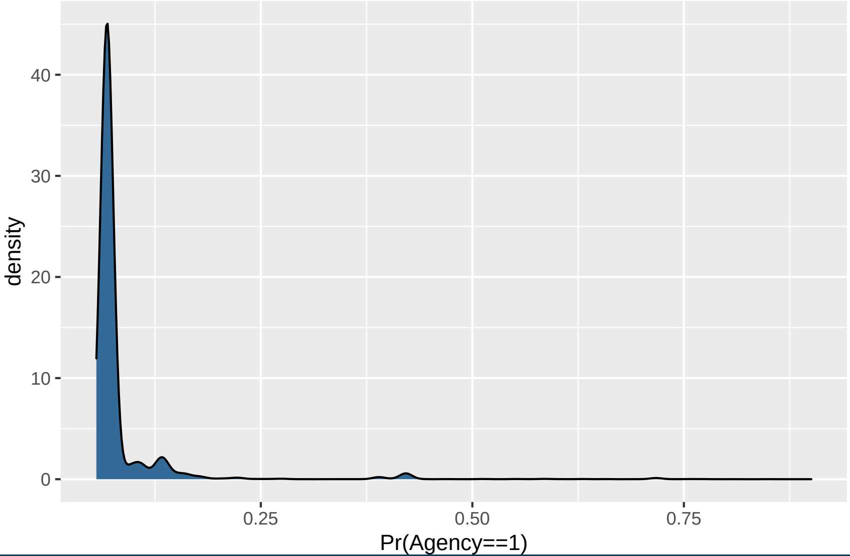
# § Pre-processing §

- Calculate non-textual features for this dataset.
- Create document-term-matrix for this dataset.
- Keep only terms that are included in the list of predictors in the training set – the predictors have to be identical!

# § Applying our model to our unseen data §

```
old_predicted <- predict.train(mygbm50_nontext,
                               newdata = olddtm),
                               type = "prob")
```

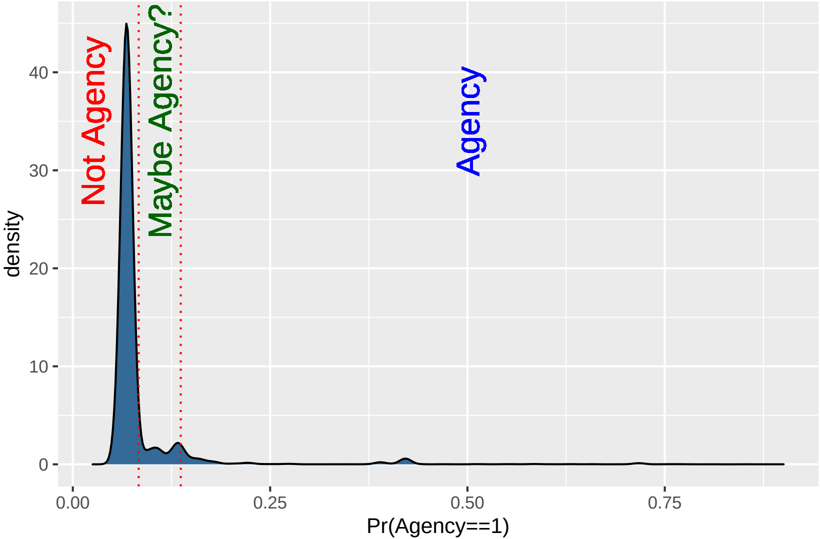
# Distribution of Probabilities



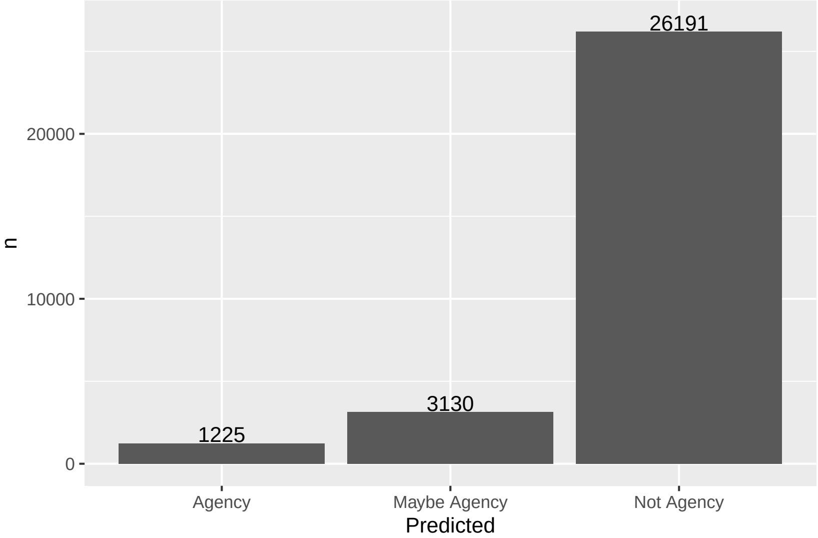
# Distribution of Probabilities: Quantiles

- 0%-4%: Probabilities 0.056-0.061
- 5%-85%: Probabilities 0.068
- 86%-95%: Probabilities 0.083-0.137
- 96%-100%: Probabilities 0.152-0.901

# Distribution of Probabilities: Quantiles



# Distribution of Predictions

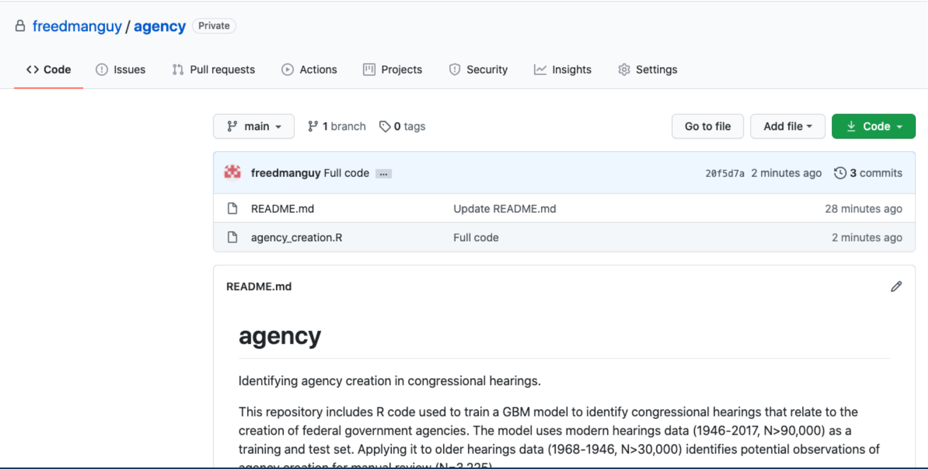


# Next Steps

- Manually review all “Agency” predictions (N = 1225)
- Manually review a sample of “Maybe Agency” predictions (N=1000)
- Manually review a sample of “Not Agency” predictions (N=1000)
- Bottom line: Manually review 3,225 observations instead of 30,551.

# github

https://github.com/freedmanguy/agency



# Questions