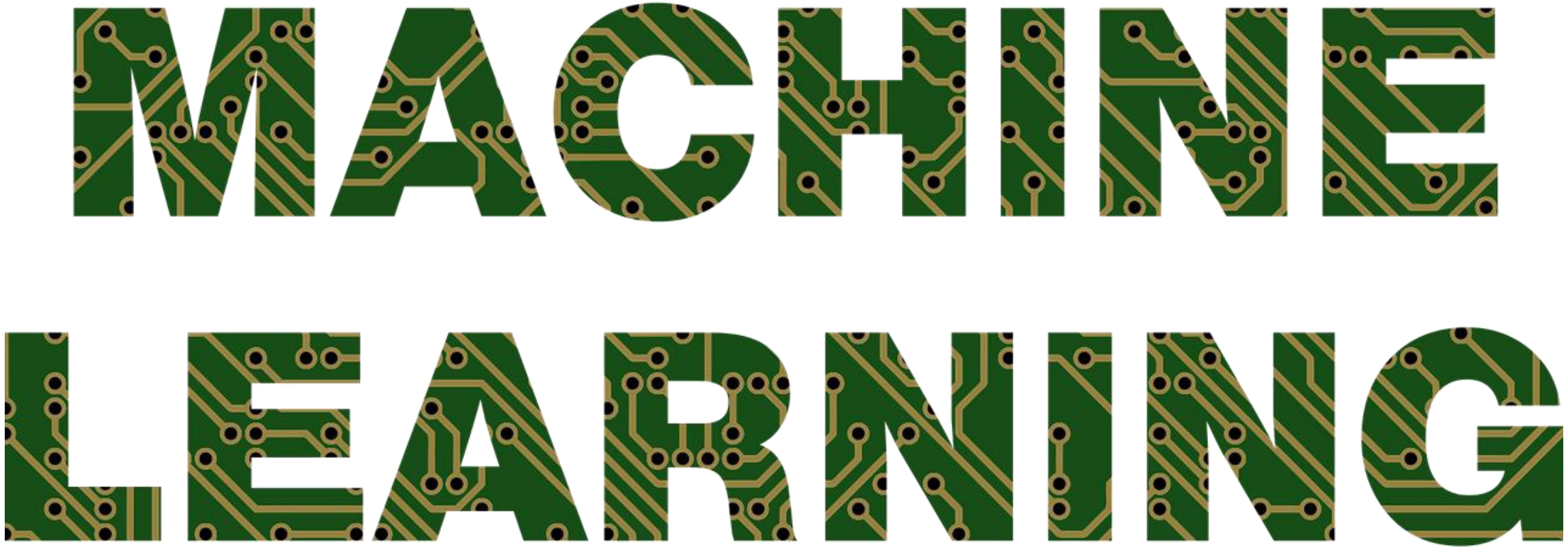

MACHINE
LEARNING

The tidymodels packages revisited

1. Train-test split


```
# Split data into training and testing sets
set.seed(1234)
vote_split <- voters_select %>%
    rsample::initial_split(prop = 0.8,
                           strata = turnout16_2016)

vote_train <- training(vote_split)
vote_test <- testing(vote_split)

dplyr::glimpse(vote_train)
dplyr::glimpse(vote_test)
```

2. Create recipe

```
# Normalize predictors
vote_recipe <-
 recipes::recipe(turnout16_2016  ~ ., data = vote_train) %>%
 update_role(case_identifier, new_role = "ID") %>%
 themis::step_upsample(turnout16_2016) %>%
 recipes::step_normalize(all_numeric())

vote_recipe
```

3. Specify model

```
## Specify a knn model
knn_spec <- parsnip::nearest_neighbor() %>%
    parsnip::set_engine("kknn") %>%
    parsnip::set_mode("classification")
```

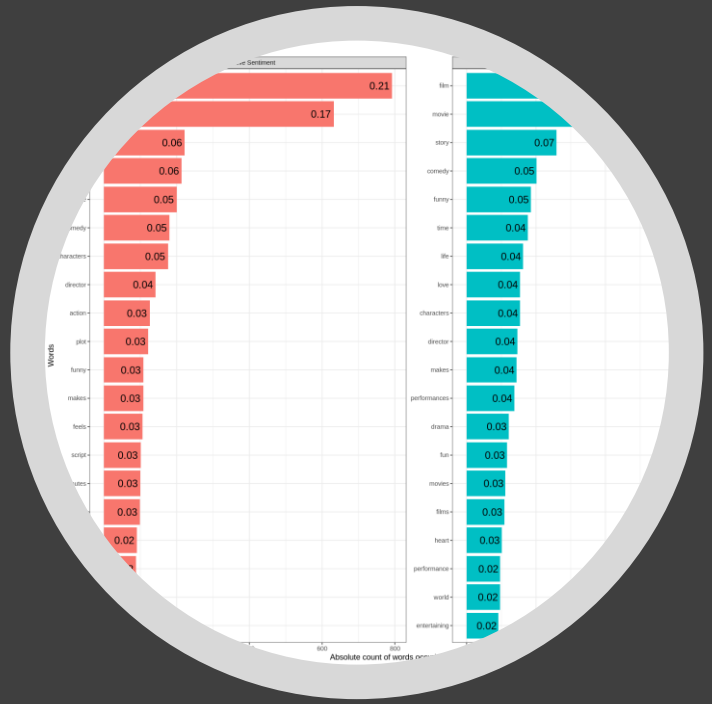
4. Create workflow

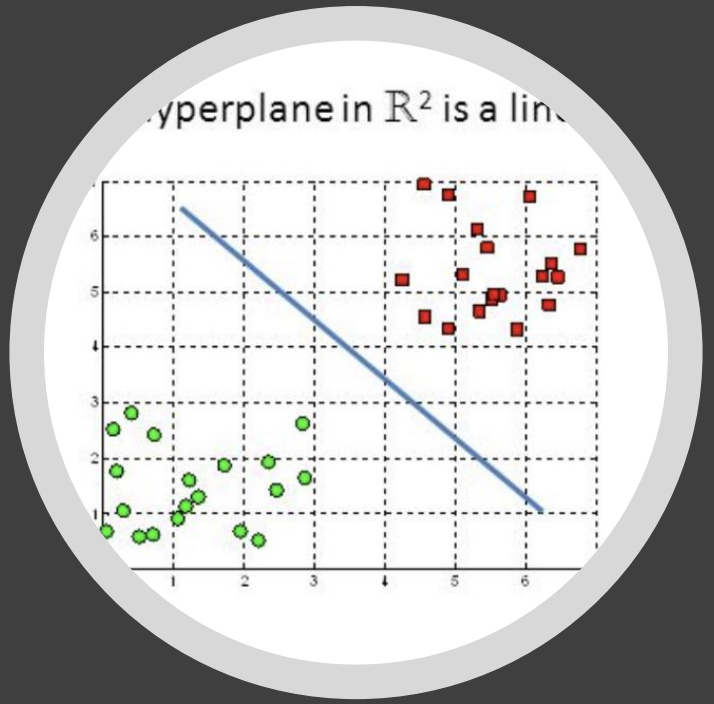

```
## Add the recipe + model to a workflow
vote_workflow <- workflow() %>%
    workflows::add_recipe(vote_recipe) %>%
    workflows::add_model(knn_spec)

vote_workflow
```

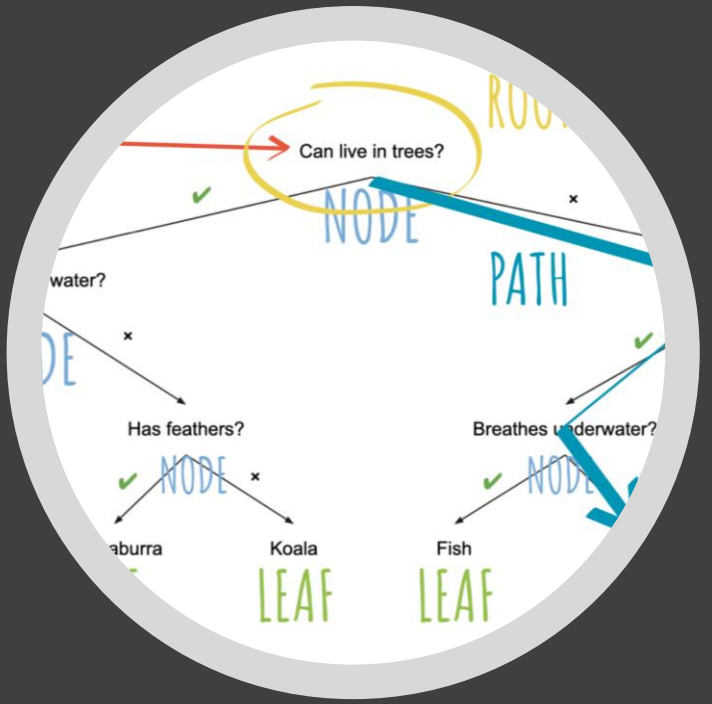
5. Fit model


```
# fit the final best model to the training set
 and evaluate the test set
knn_fit <- vote_workflow %>%
  tune::last_fit(vote_split)

print(knn_fit)
```

https://www.tidymodels.org/packages/
https://dplyr.tidyverse.org/

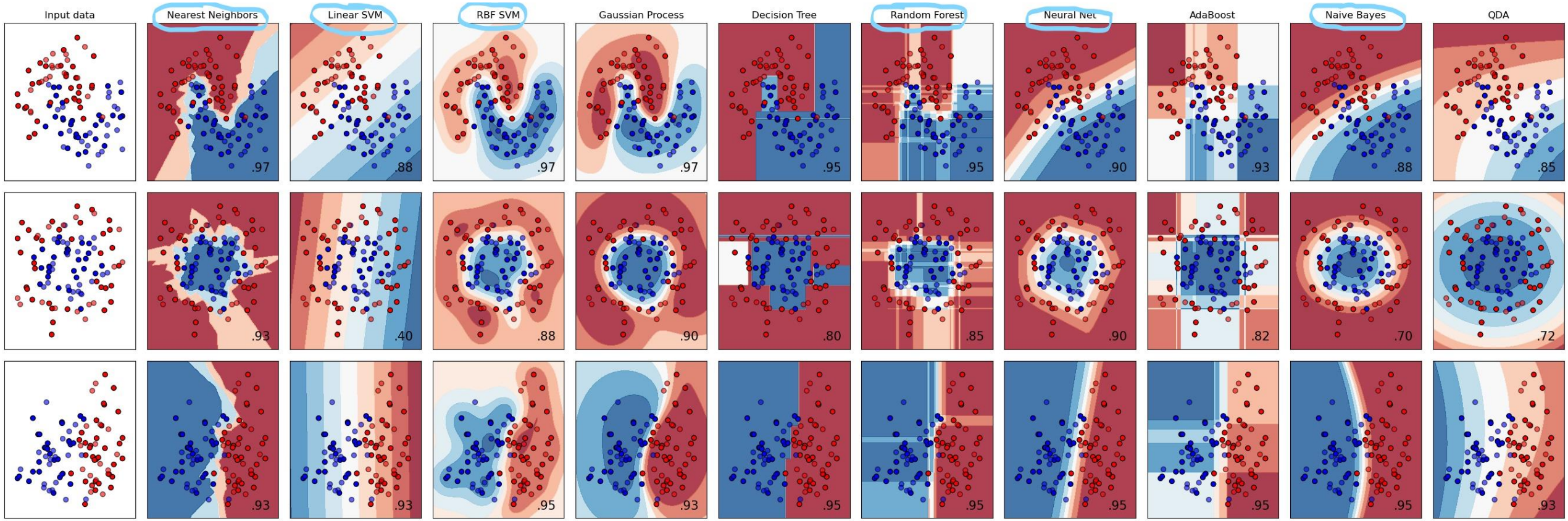






The zoo of supervised learning models

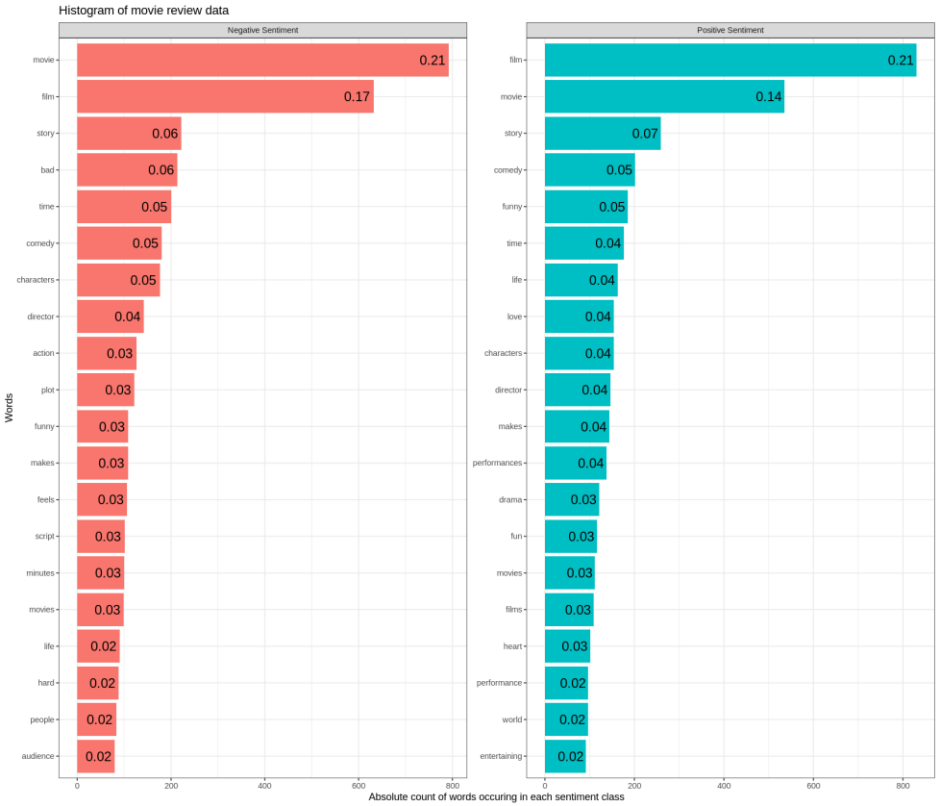
The zoo of supervised learning (classification) models



https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py

The zoo of supervised learning (classification) models – multinominal Naïve Bayes (NB)

Data:



Negative: "it's so laddish and juvenile, only teenage boys could possibly find it funny"

Positive: "provides a porthole into that noble, trembling incoherence that defines us all"

https://www.youtube.com/watch?v=O2L2Uv9pdDA
https://doi.org/10.3115/1219840.1219855
http://www.cs.cornell.edu/people/pabo/movie-review-data/
https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
https://web.stanford.edu/~jurafsky/slp3/
https://towardsdatascience.com/k-nearest-neighbors-knn-explained-cbc31849a7e3

The algorithm:

1. Calculate a list (histogram) of relative frequencies (probabilities/likelihoods) per class using the *bag of words* approach.

2. Calculate probabilities of seeing a positive or negative review

$$p(positive) = \frac{5330}{10660} = 0.5$$

$$p(negative) = \frac{5330}{10660} = 0.5$$

3. Calculate conditional probabilities for a **single movie review**:

$$p(funny\ \ film|\ positive) = p(positive) * p(funny) * p(film) = 0.5\ * 0.05\ * 0.21 = \mathbf{0.00525}$$

$$p(funny\ \ film|\ negative) = p(negative) * p(funny) * p(film) = 0.5\ * 0.00\ * 0.17 = \mathbf{0.00}$$

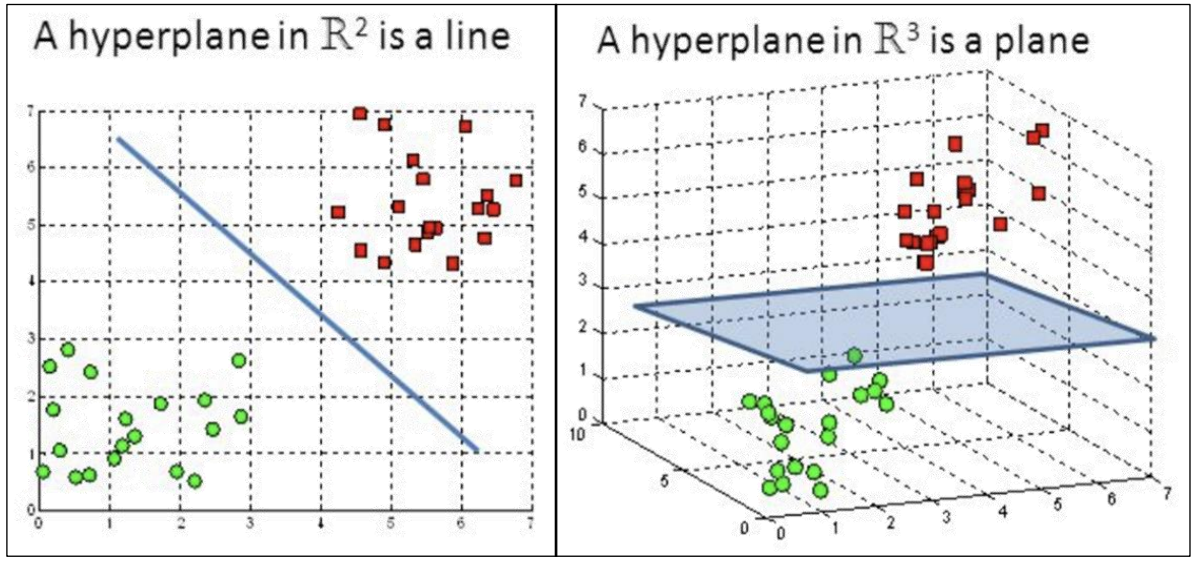
→ Conditional probability scores of seeing the words "funny" and "film" given that the movie review is a positive or negative one

4. Compare scores for both classes: $\mathbf{0.00525} > \mathbf{0.00}$

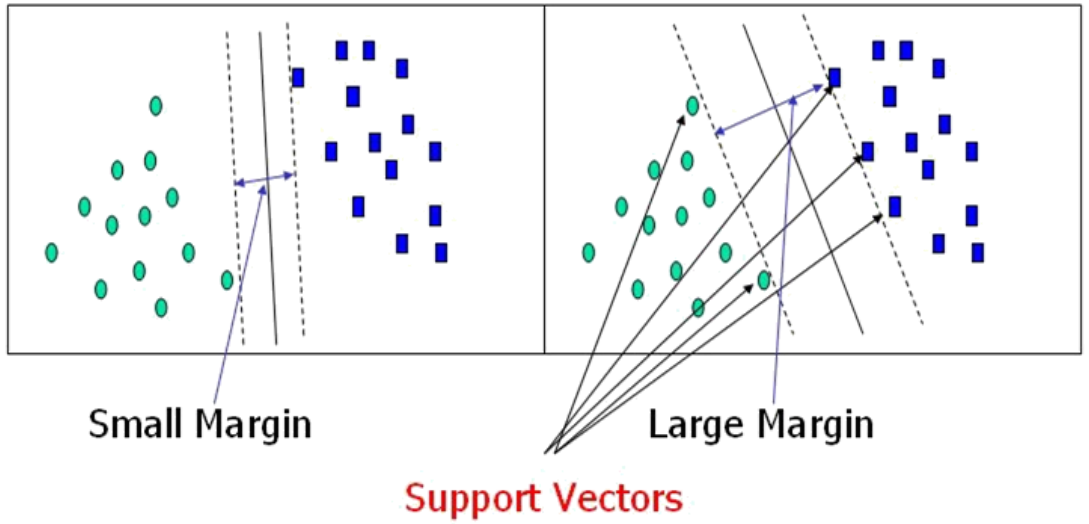
5. Evaluate model on a testing dataset with known labels

The zoo of supervised learning (classification) models – Support Vector Machine (SVM)

Separating hyperplanes:



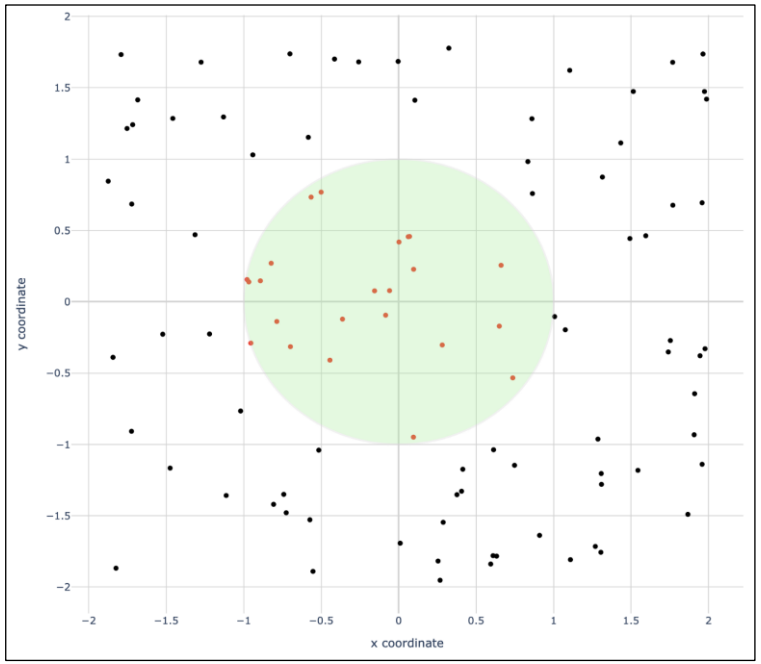
Margins:

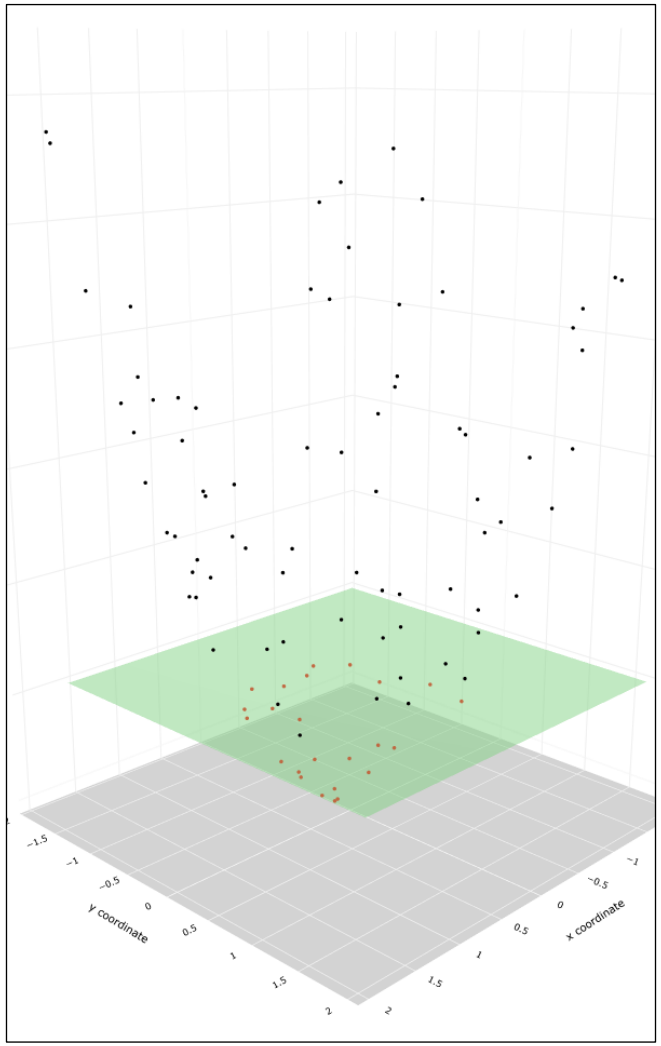


https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
https://medium.datadriveninvestor.com/support-vector-machines-svms-4bcccbd78369
http://vision.stanford.edu/teaching/cs231n-demos/linear-classify/
https://towardsdatascience.com/support-vector-machine-explained-8d75fe8738fd

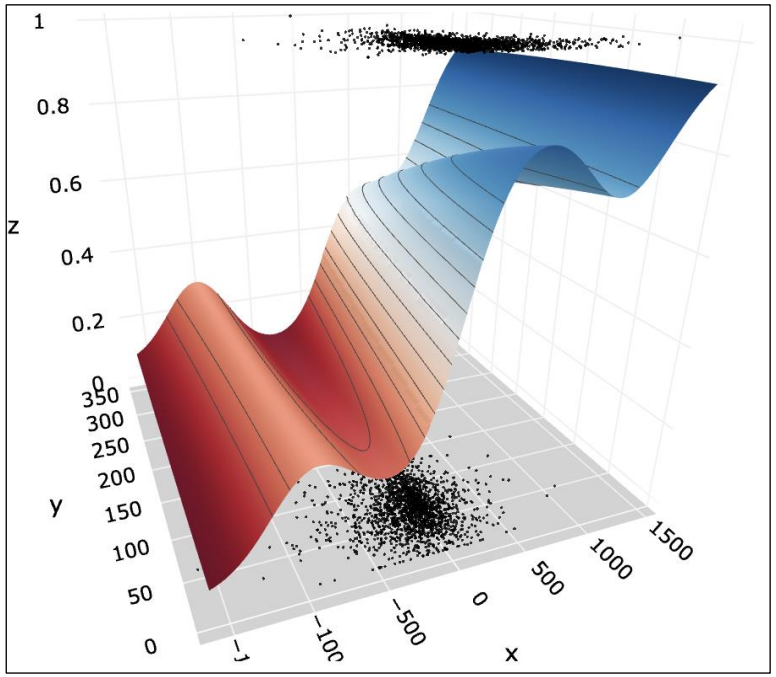
The zoo of supervised learning (classification) models – Support Vector Machine (SVM)

The kernel trick:





Non-linear kernels:



https://towardsdatascience.com/svm-classifier-and-rbf-kernel-how-to-make-better-models-in-python-73bb4914af5b

The zoo of supervised learning (classification) models – Support Vector Machine (SVM)

The algorithm for a linear SVM in two dimensions:

1. "Draw" data points into a cartesian coordinate system with two dimensions
2. If the data are not linearly separable (you cannot draw a straight line that separates them) apply the kernel trick and "pull" the data apart into a third dimension
3. Find a straight line (or hyperplane) that separates the classes
4. Adjust the hyperplane until the average distance to the nearest points of both classes is maximized (maximum margin classifiers)
5. Evaluate model on a testing dataset with known labels

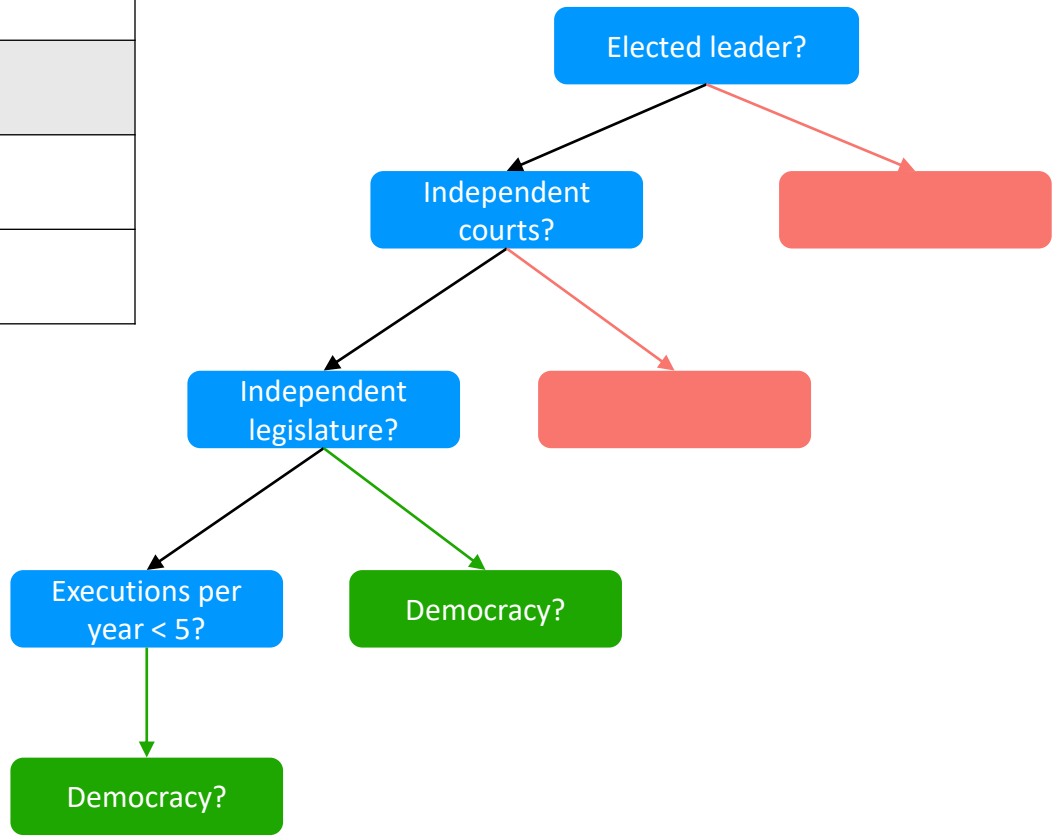
https://www.youtube.com/watch?v=efR1C6CvhmE

The zoo of supervised learning (classification) models – Random Forest

Original dataset

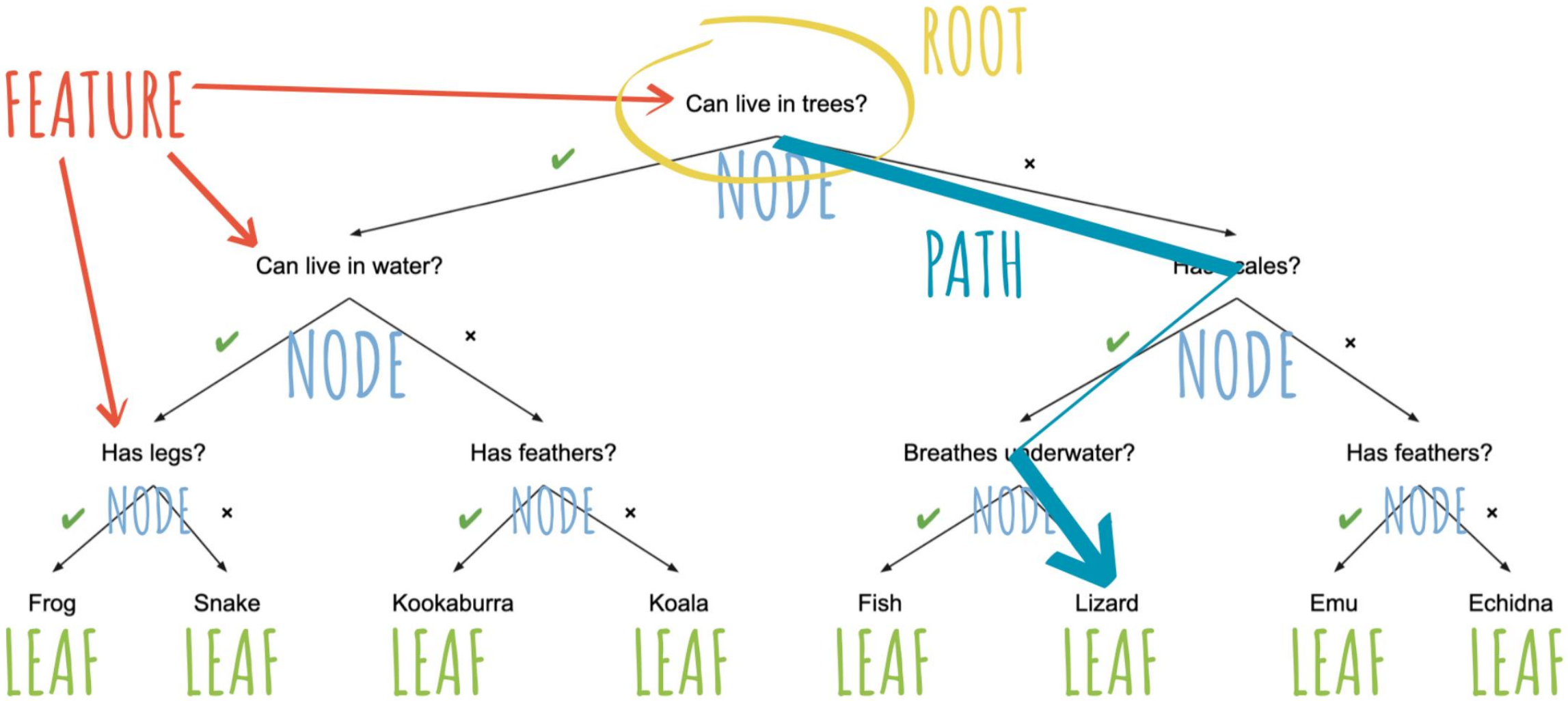
Elected leader	Independent courts	Independent legislature	Executions per year	Democracy
No	No	No	80	No
Yes	Yes	Yes	2	Yes
Yes	Yes	No	5	No
Yes	No	Yes	61	No

A decision tree



https://www.youtube.com/watch?v=J4Wdy0Wc_xQ
https://towardsdatascience.com/understanding-decision-trees-for-classification-python-9663d683c952
https://towardsdatascience.com/decision-tree-and-random-forest-explained-8d20ddabc9dd

The zoo of supervised learning (classification) models – Random Forest

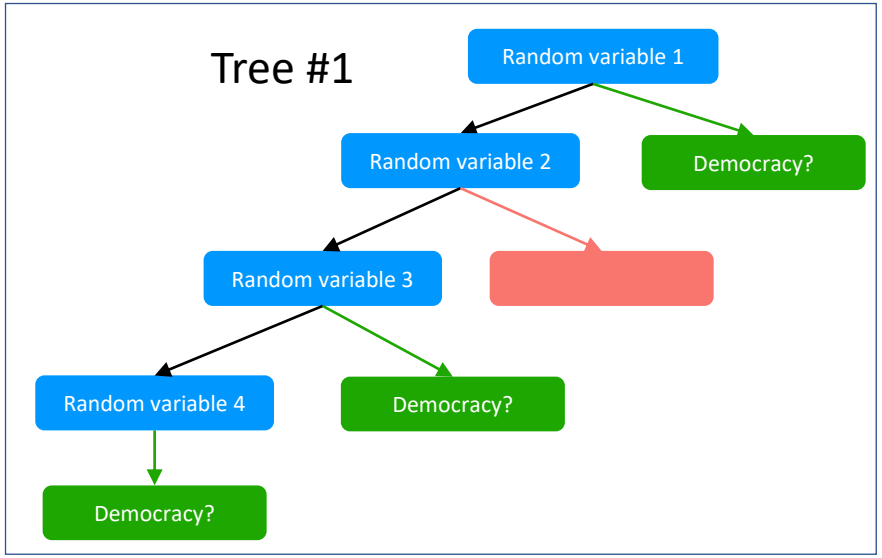


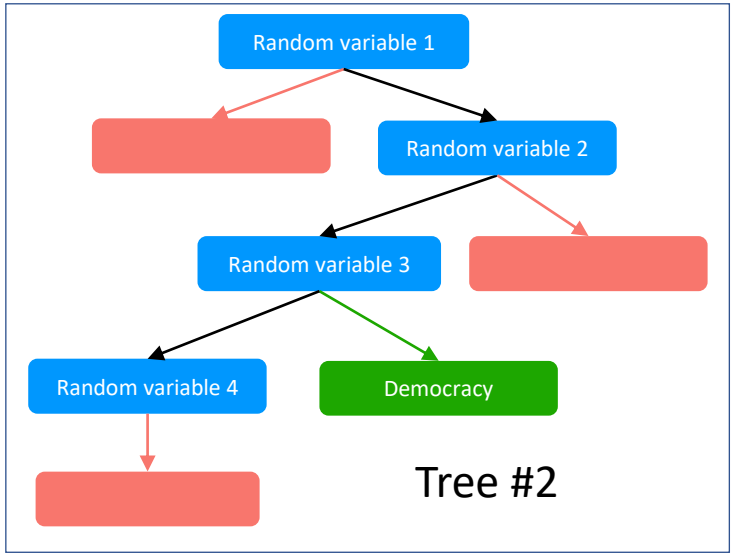
https://conf20-intro-ml.netlify.app/slides/02-classifying.html#50

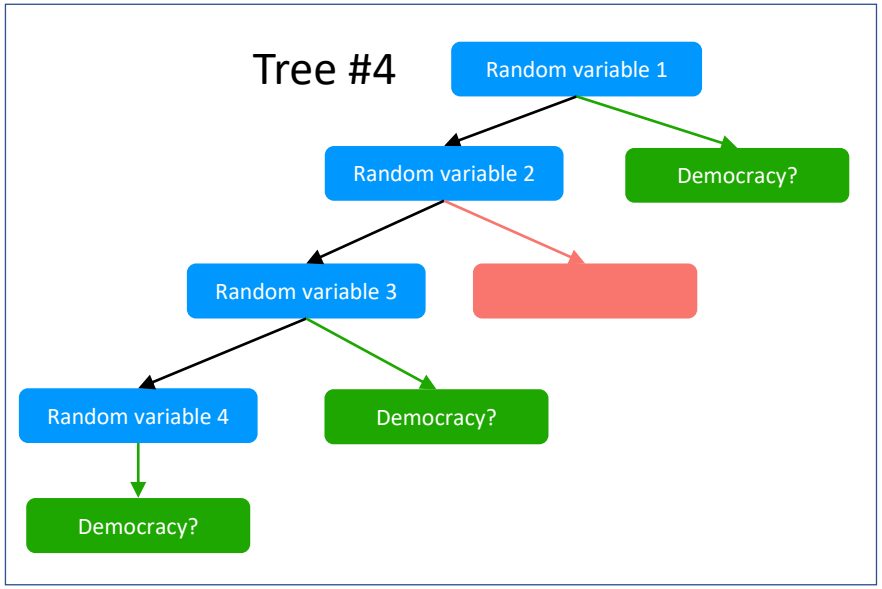
The zoo of supervised learning (classification) models – Random Forest

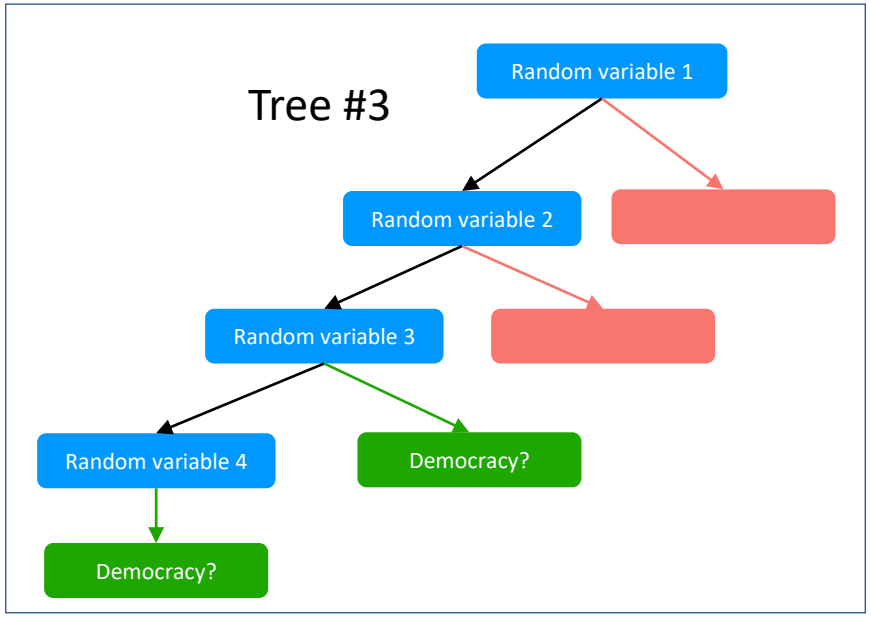
New data point

Elected leader	Independent courts	Independent legislature	Executions per year	Democracy
Yes	Yes	Yes	523	No







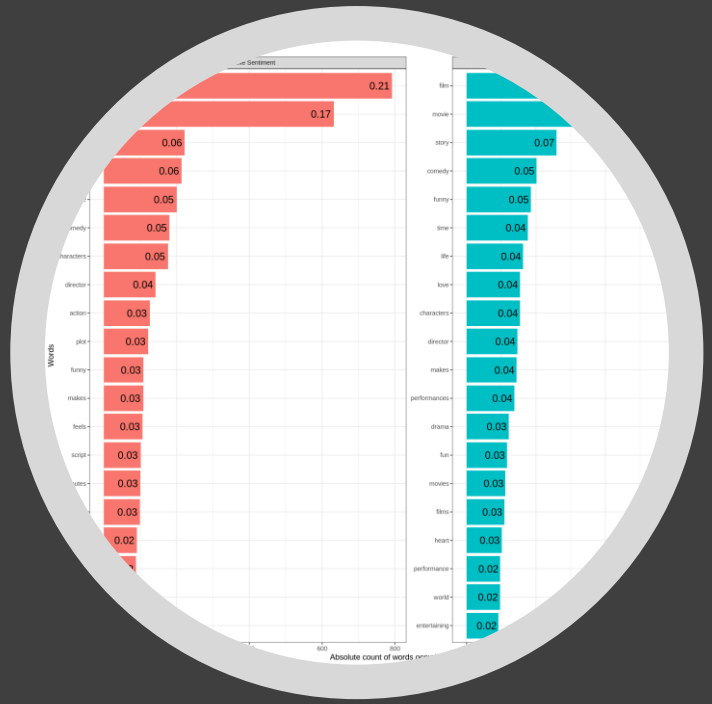


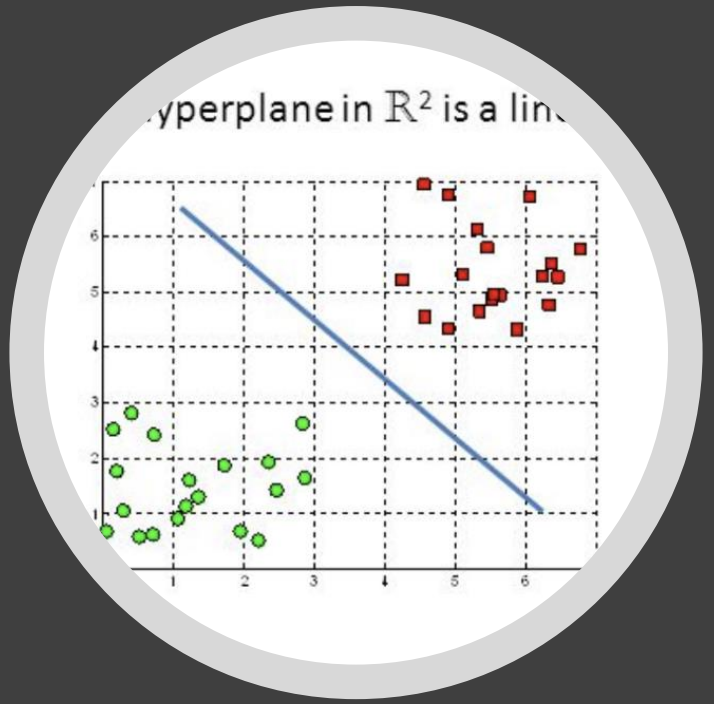
Democracy?
Yes: 1
No: 3

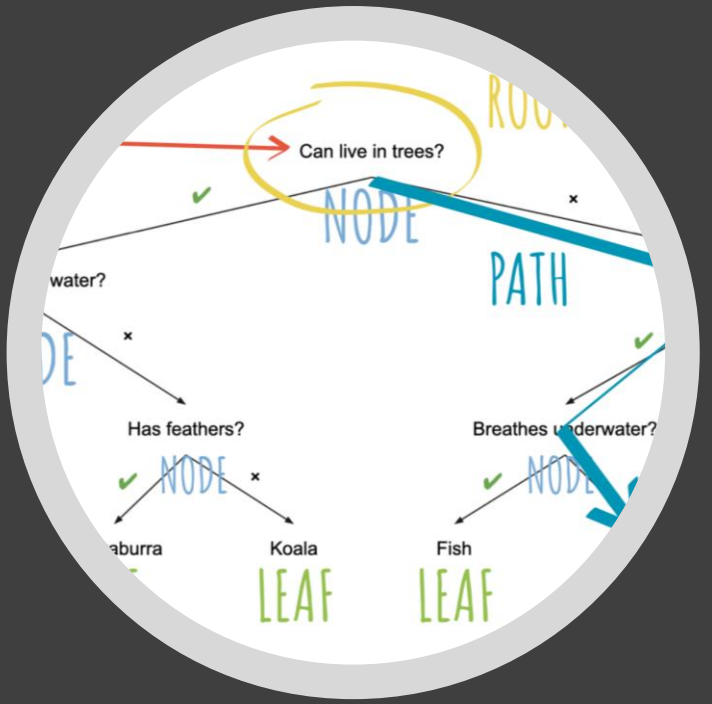
The zoo of supervised learning (classification) models – Random Forest

The algorithm for a random forest with 10 trees

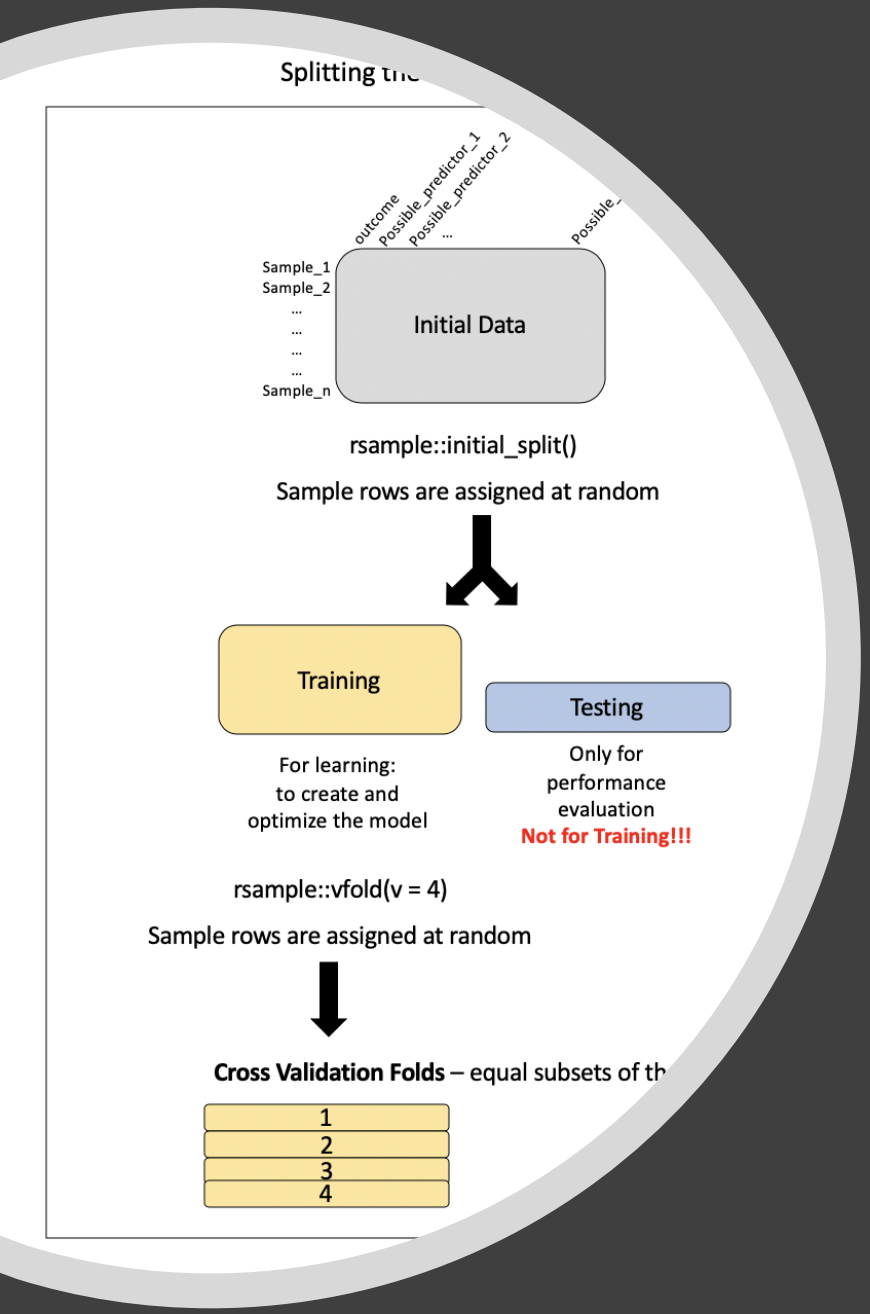
1. Create a bootstrapped dataset from the original dataset
2. Build a decision tree with the number of decision nodes equal to the number of predictors
3. Train the tree with the bootstrapped dataset
4. Repeat steps 1-3 10 times
5. Evaluate model on a testing dataset with known labels
    1. Let every tree decide on a new data point
    2. Let the trees vote on how to classify the new data point

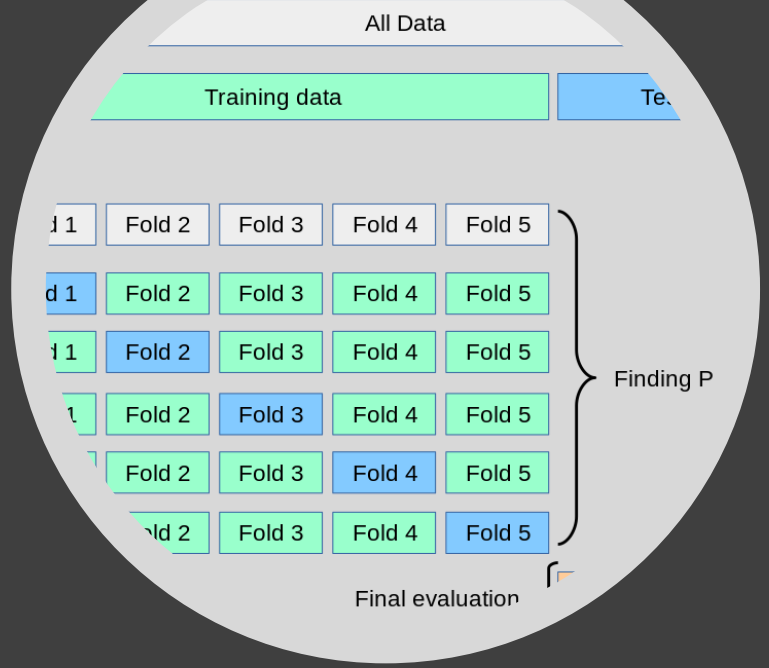


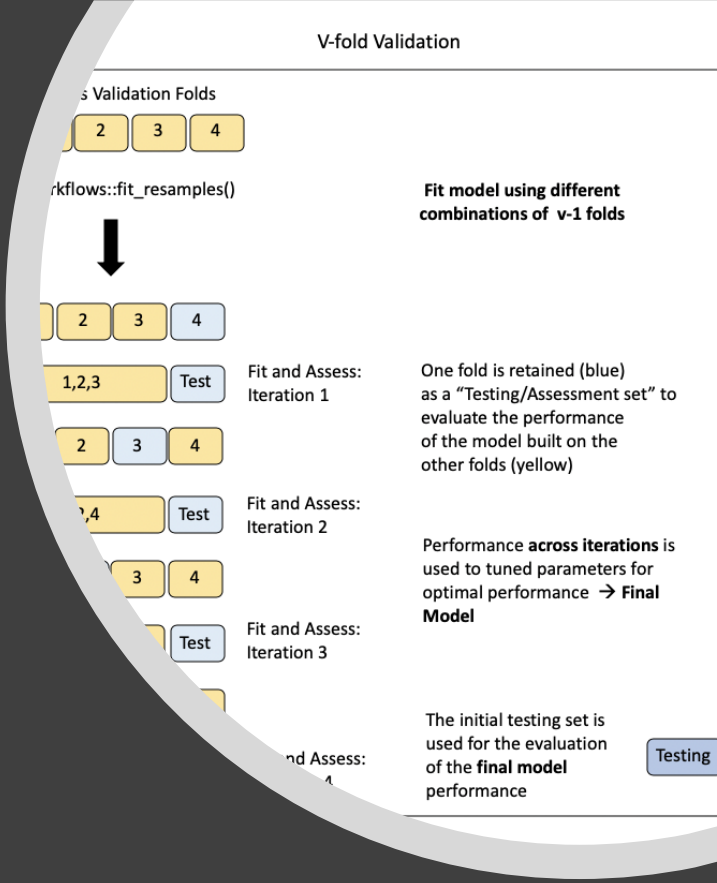




The zoo of supervised learning models

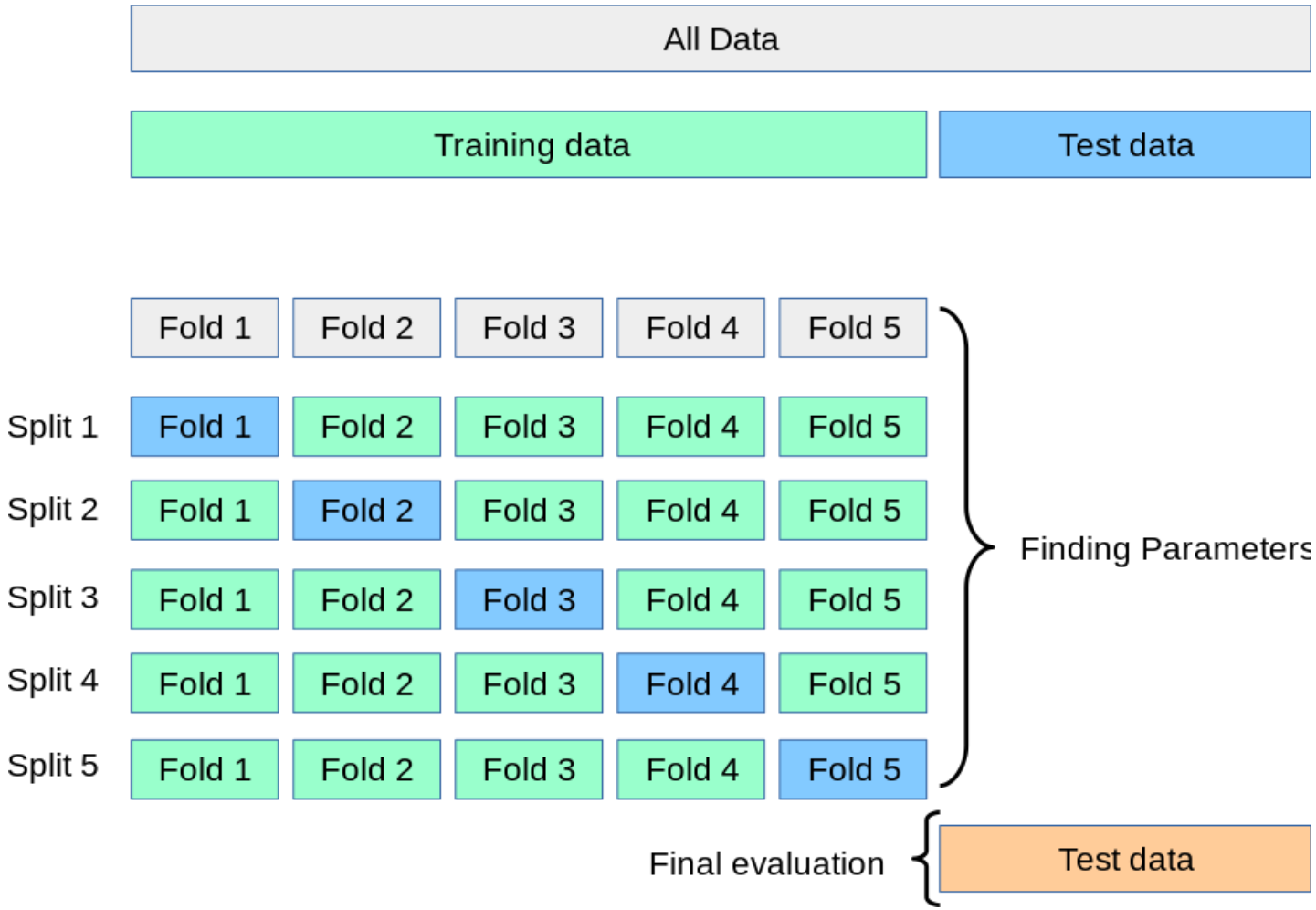






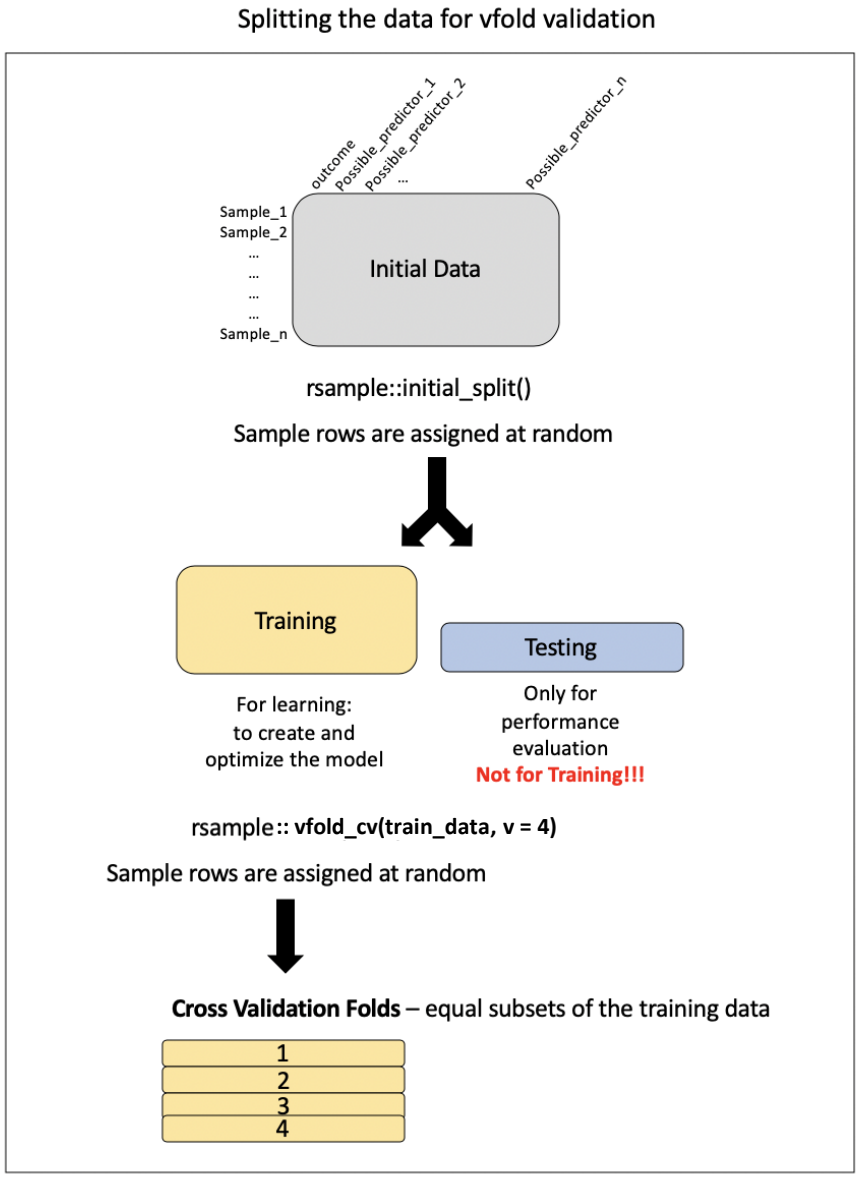
Train test validate

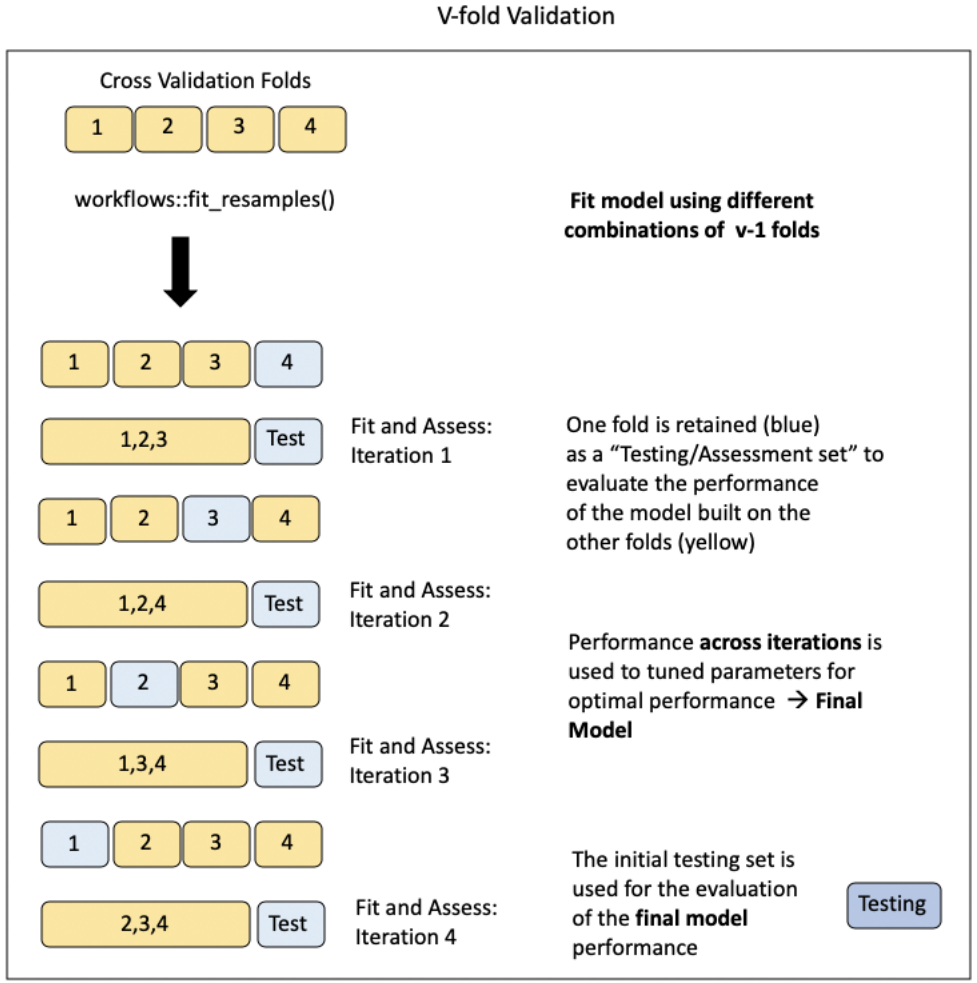
V-fold cross validation



https://scikit-learn.org/stable/modules/cross_validation.html

V-fold cross validation





https://jhudatascience.org/tidyversecourse/model.html