MACHINE LEARNING 1
ADVANCED CRIME ANALYSIS
UCL

BENNETT KLEINBERG

18 FEB 2019

MACHINE LEARNING 1

TODAY

- Recap week 1-5
- Intro to machine learning
    - Types of ML
    - Supervised machine learning
    - Step-by-step example
    - Important algorithms

RECAP WEEK 2

APIS

RECAP WEEK 3

WEBSCRAPING

RECAP WEEK 4

TEXT MINING 1

RECAP WEEK 5

TEXT MINING 2

MACHINE LEARNING?

- core idea: a system learns from experience
- no precise instructions

Applications?

WHY DO WE WANT THIS?

Step back…

How did you perform regression analysis in PSM2?

OKAY ...

- you've got one outcome variable (e.g. number of shooting victims)
- and two predictors (e.g. gender of shooter, age)
- typical approach $victims \ gender + age$
- regression equation with intercept, beta coefficients and inferred error term

BUT!

Often we have no idea about the relationships.

- too many predictors
- too diverse a problem
- simply unknown

ML IN GENERAL

- concered with patterns in data
- learning from data
- more experience results typically in better models
- data, data, data

TYPES OF MACHINE LEARNING

BROAD CATEGORIES

- Supervised learning (today)
- Unsupervised learning (next week)
- Hybrid models
- Deep learning
- Reinforcement learning

DEEP LEARNING

Inspired by the human brain.

- MIT’s course website https://deeplearning.mit.edu/
- Lex Fridman’s courses from MIT –> YouTube

REINFORCEMENT LEARNING

- Excellent YouTube examples from code bullet
- e.g. AI Learns to play the Worlds Hardest Game

Demo

SUPERVISED LEARNING

WTF IS SUPERVISED?

- supervised = labeled data
- i.e. you know the outcome
- flipped logic

Contrary: unsupervised.

CLASSES OF SUPERVISED LEARNING

- classification (e.g. death/alive, fake/real)
- regression (e.g. income, number of deaths)

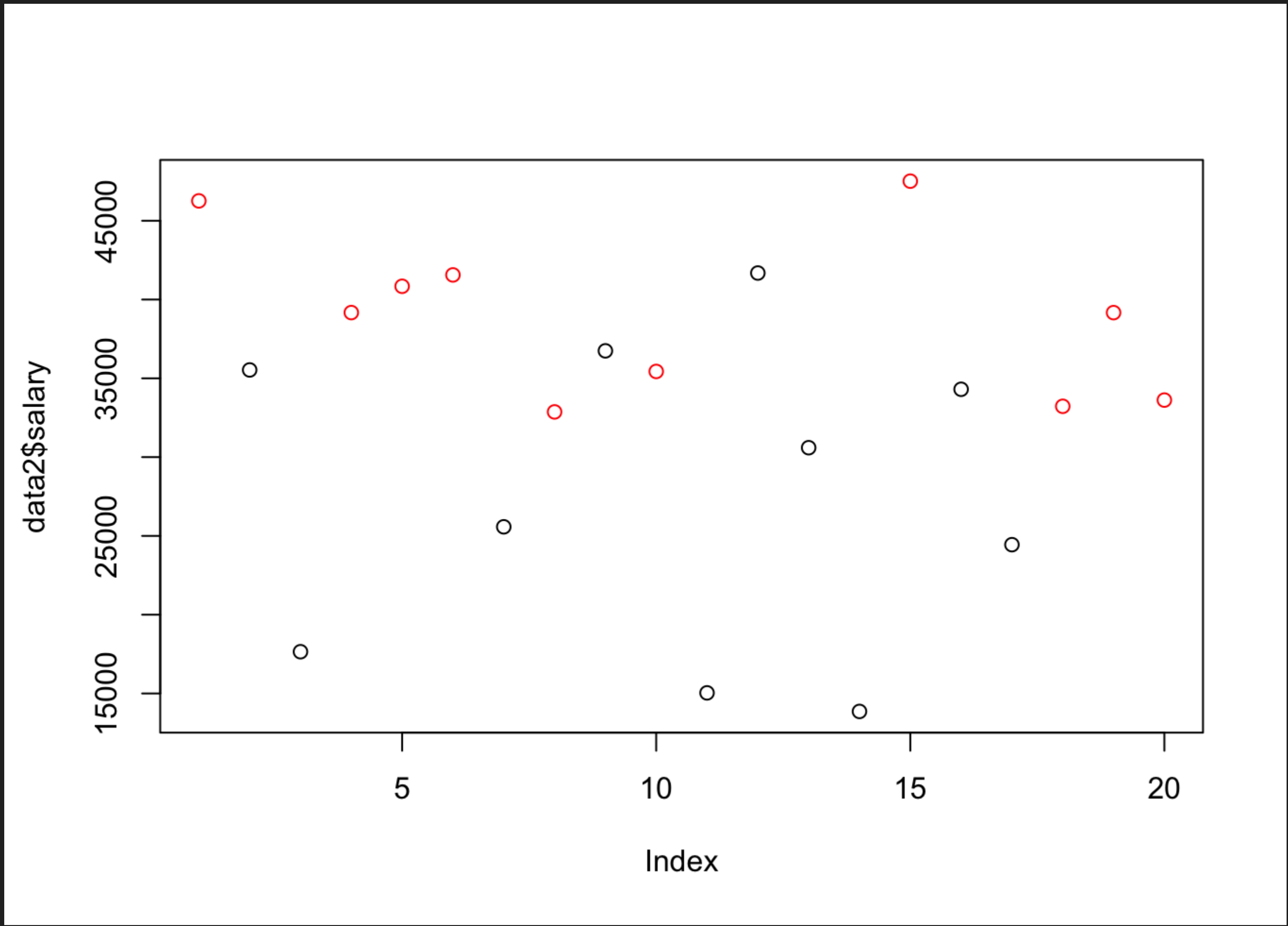
MINI EXAMPLE

Supervised classification

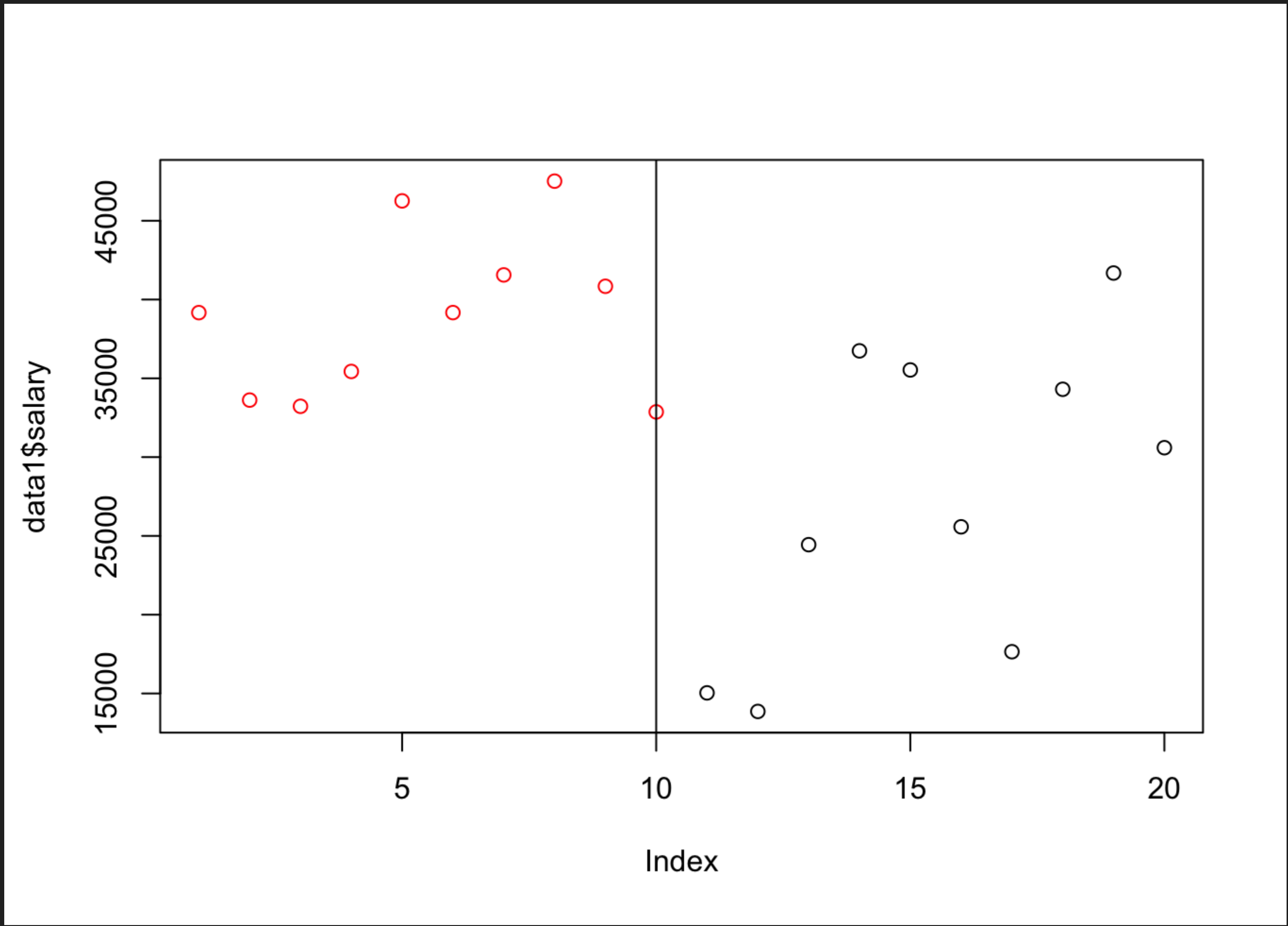
SIMPLE EXAMPLE

- gender prediction
- based on salary

gend	er sal	ary
1	male	39169
2	male	33620
3	male	33225
4	male	35437
11	female	15039
12	female	13861
13	female	24443
14	female	36744


data2$salary
15000
25000
35000
45000
5
10
15
20
Index

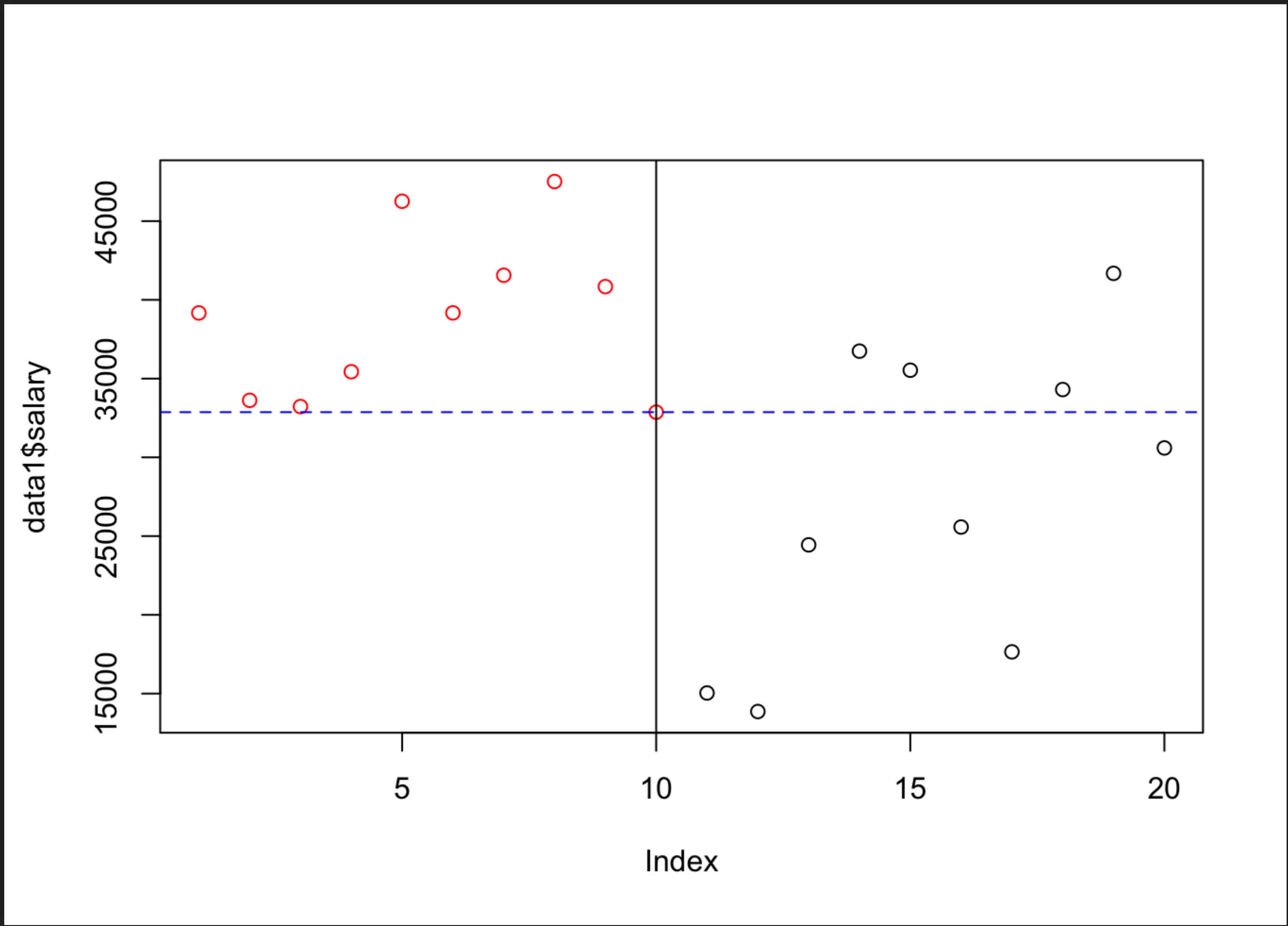
How to best separate the data into two groups?



CORE IDEA

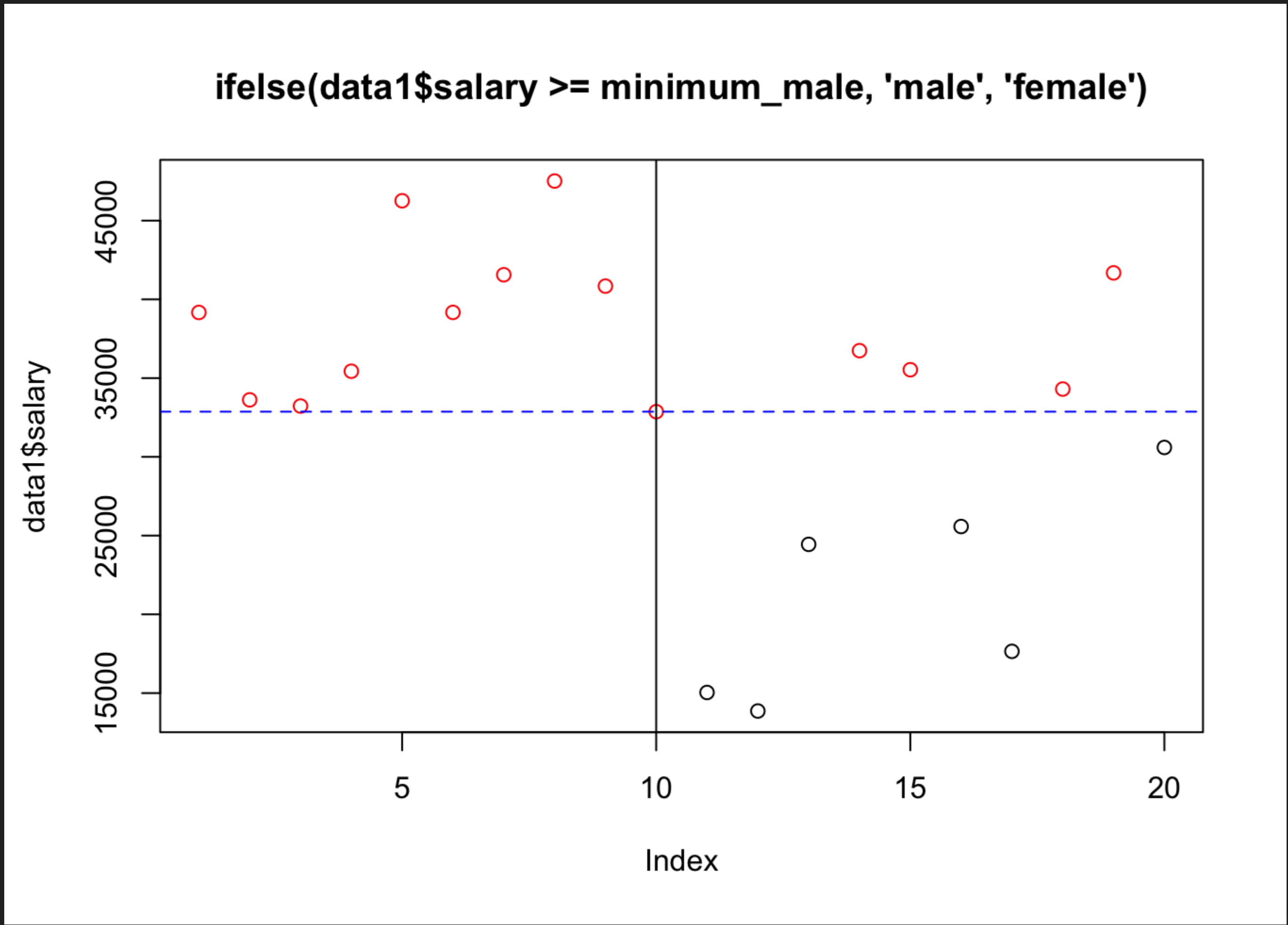
- learn relationship between
  - outcome (target) variable
  - features (predictors)
- “learning” is done through an algorithm
  - simplest algorithm: `if A then B`

IDEA 1: MALE SALARY THRESHOLD

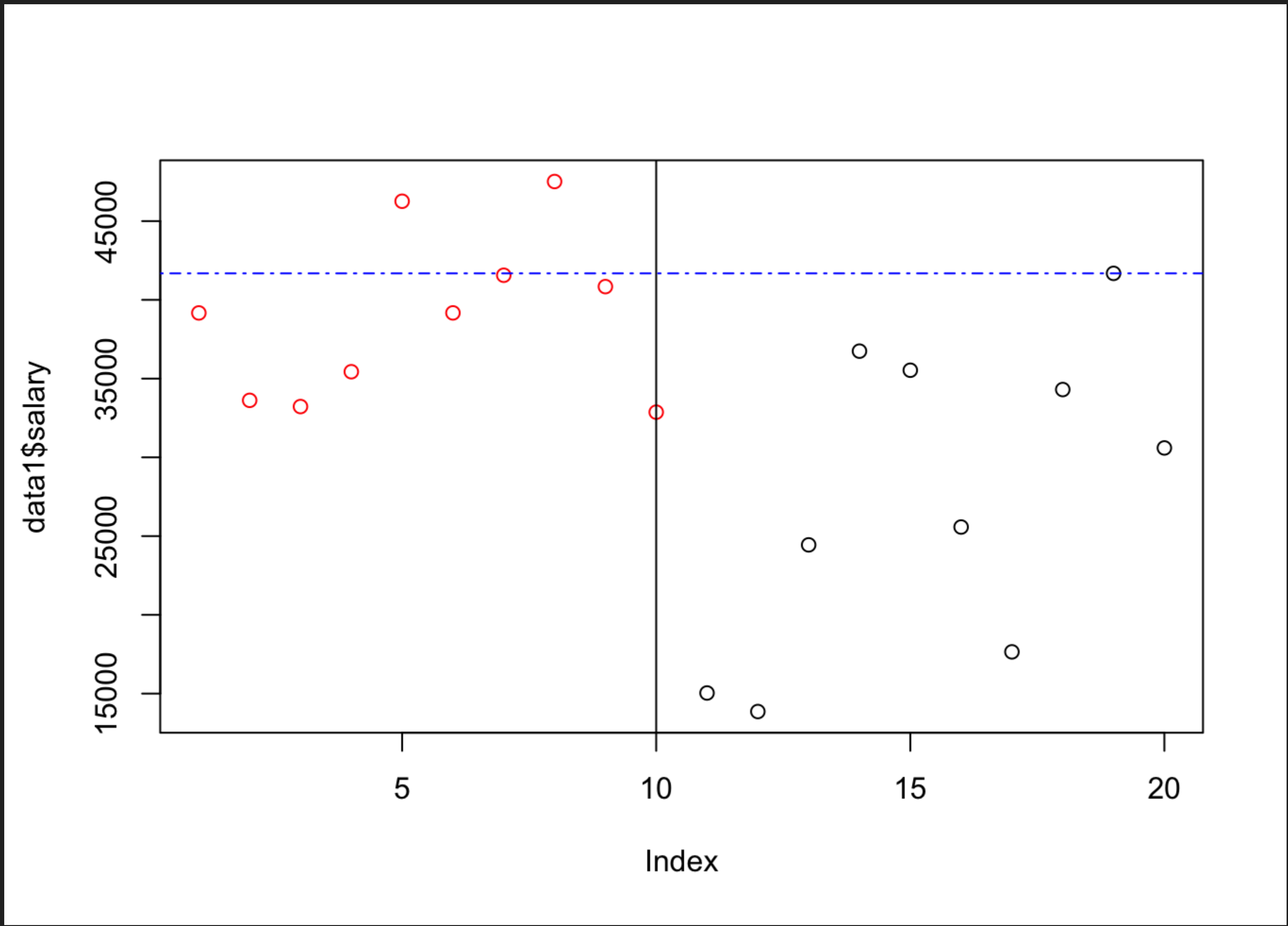


IDEA 1: MALE SALARY THRESHOLD

```
minimum_male = min(data1$salary[data1$gender == 'male']) #32869
data1$my_prediction = ifelse(data1$salary >= minimum_male, 'male', 'f
```

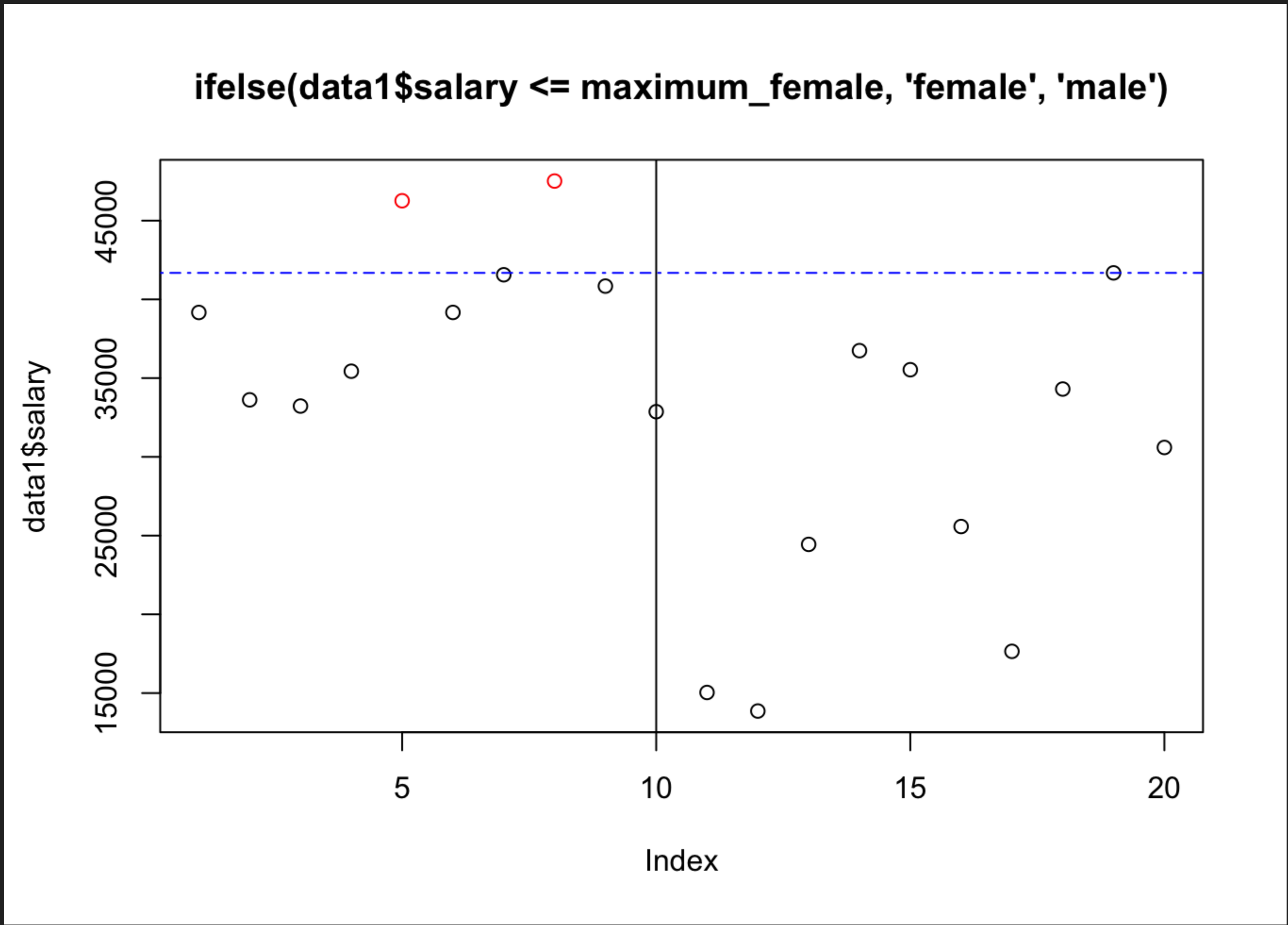


IDEA 2: FEMALE SALARY THRESHOLD



IDEA 2: FEMALE SALARY THRESHOLD

```r
maximum_female = max(data1$salary[data1$gender == 'female']) #41682
data1$my_prediction2 = ifelse(data1$salary <= maximum_female, 'female
```



But this is not learning!

STEPWISE SUPERVISED ML

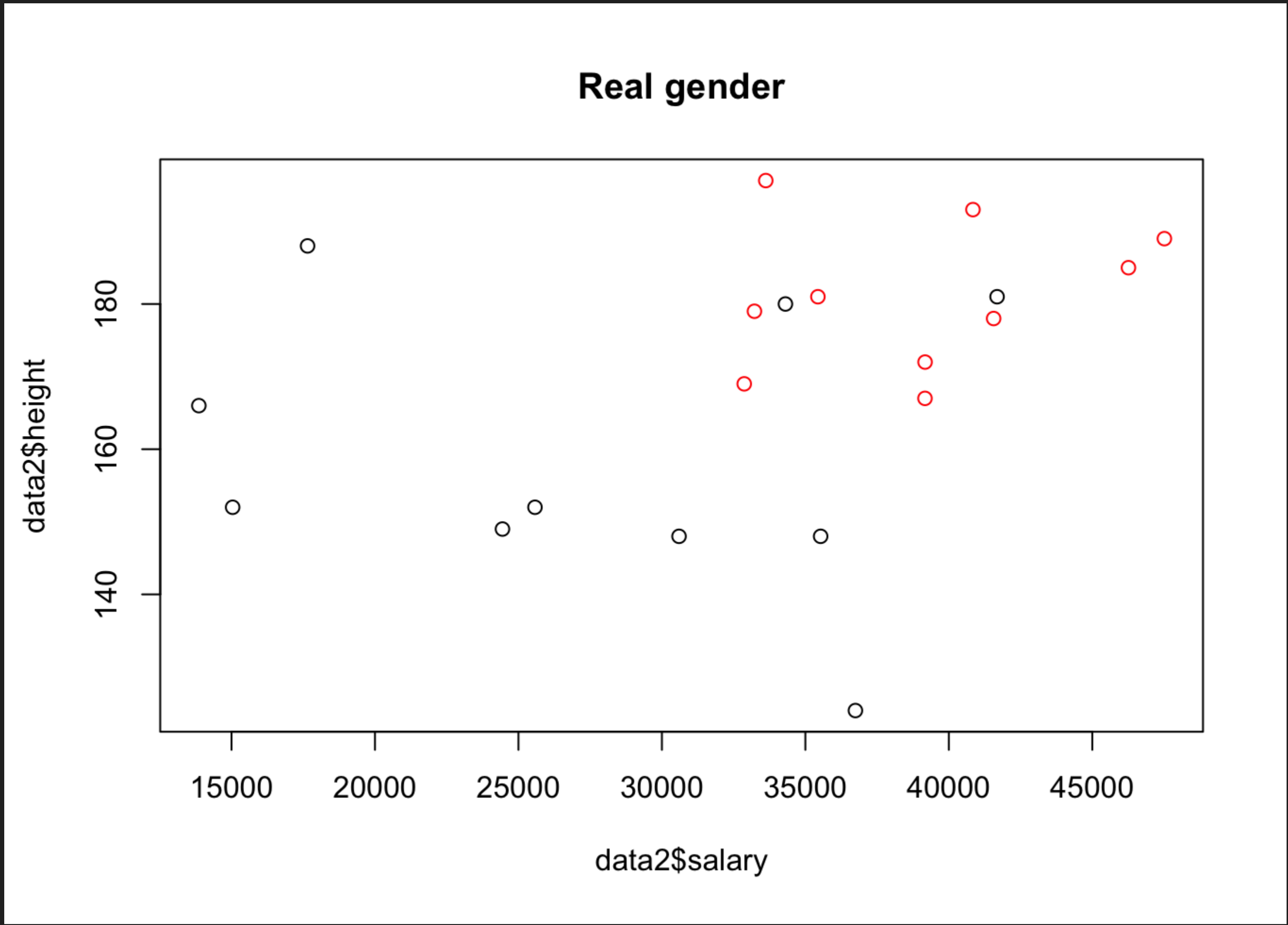
- clarify what `outcome` and `features` are
- determine which classification algorithm to use
- train the model

ENTER: `CARET`

```
library(caret)
```

- excellent package for ML in R
- well-documented website
- common interface for 200+ models

CARET IN PRACTICE



`CARET` IN PRACTICE

```
my_first_model = train(gender ~ .
                       , data = data2
                       , method = "svmLinear"
                       )
```

Now you have trained a model!

- you have taught an algorithm to learn to predict gender from salary & height

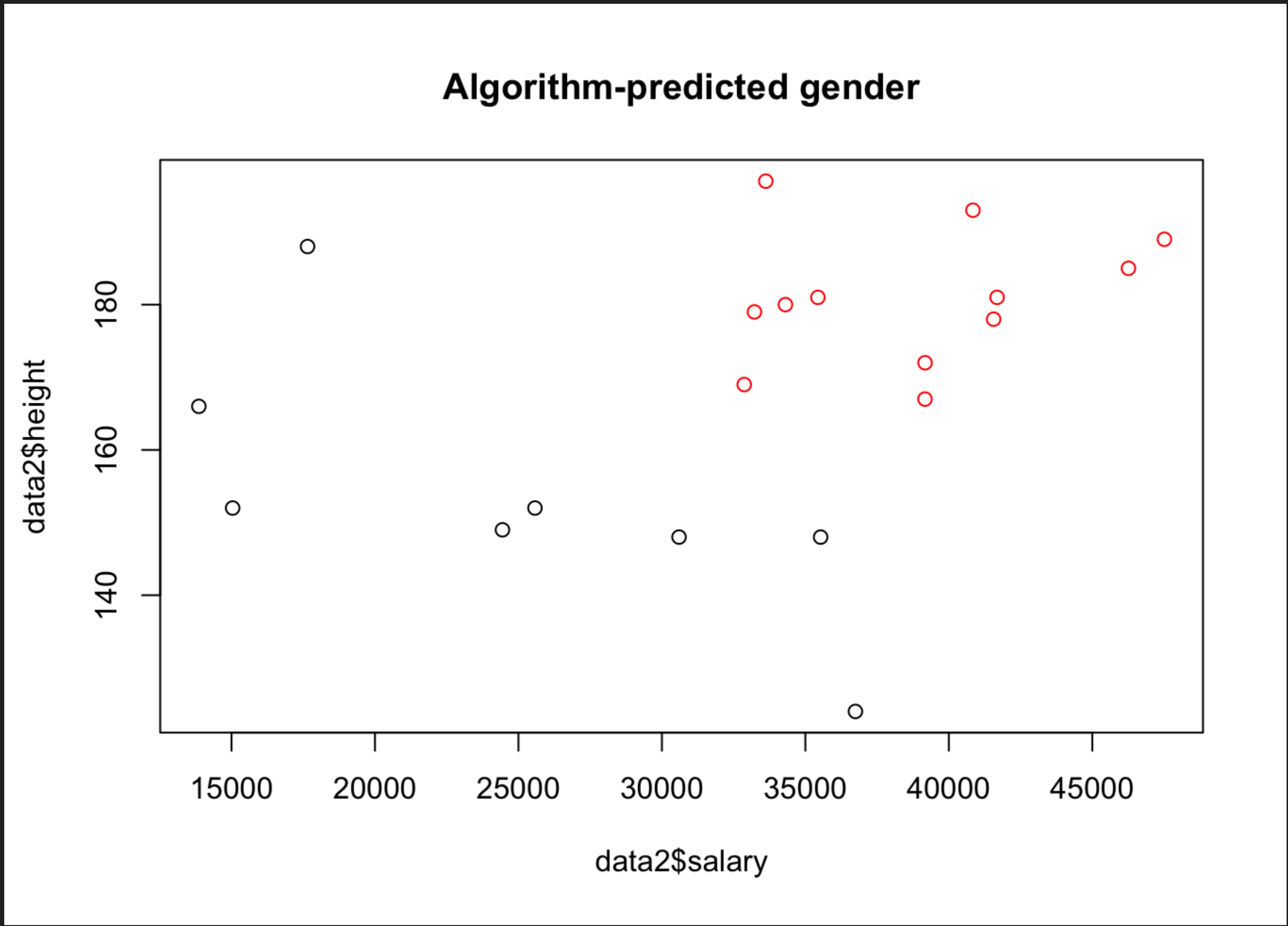
But now what?

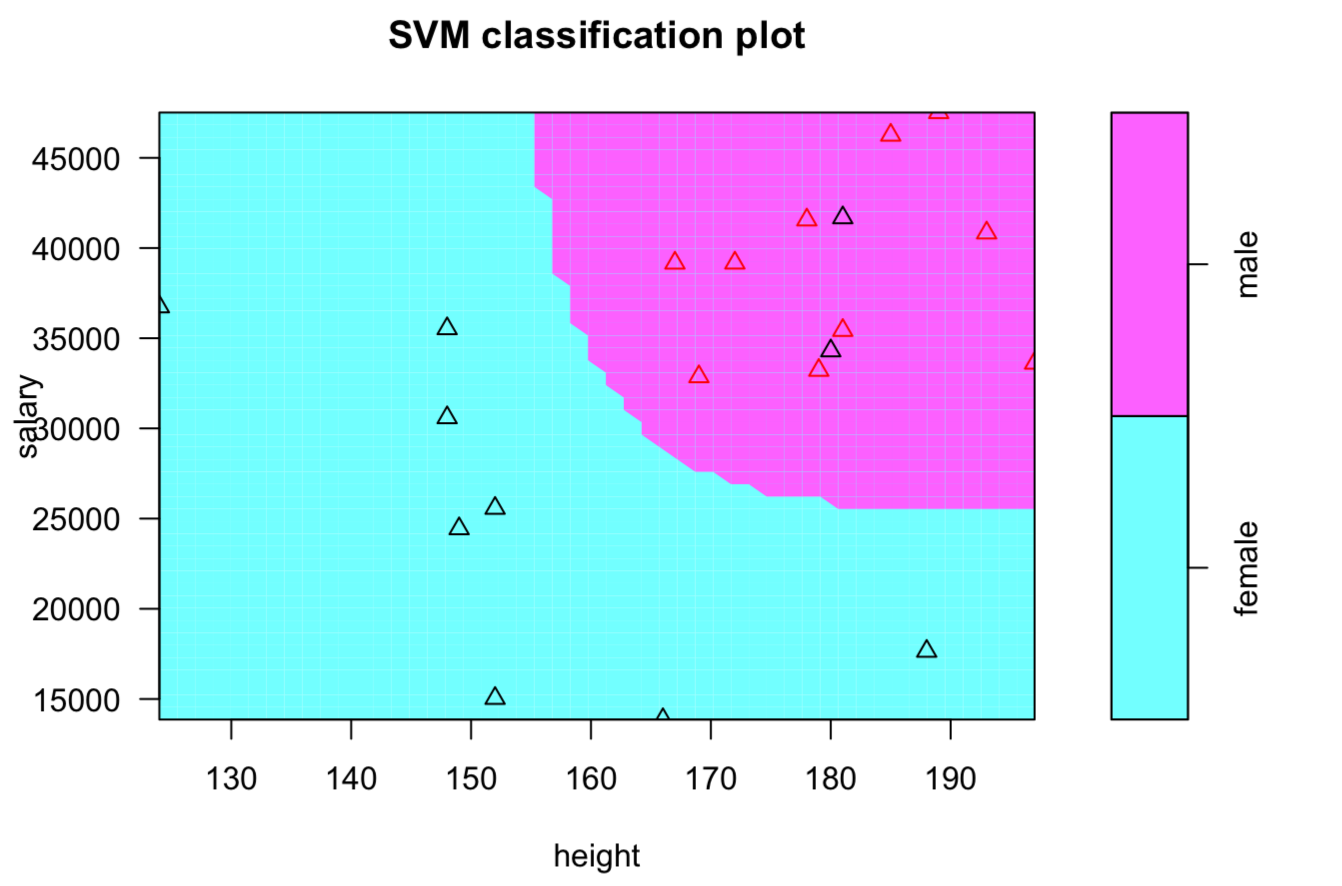
PUT YOUR MODEL TO USE

Make predictions:

```
data2$model_predictions = predict(my_first_model, data2)
```

	female	male
female	8	2
male	0	10


Algorithm-predicted gender
data2$height
140
160
180
15000
20000
25000
30000
35000
40000
45000
data2$salary



SVM classification plot
45000
40000
35000
30000
25000
20000
15000
salary
130
140
150
160
170
180
190
height
male
female

THE KEY CHALLENGE?

Think about what we did…

PROBLEM OF INDUCTIVE BIAS

- remember: we learn from the data
- but what we really want to know is: how does it work on "unseen" data

How to solve this?

KEEP SOME DATA FOR YOURSELF

Train/test split

- split the data (e.g. 80%/20%, 60%/40%)
- use one part as TRAINING SET
- use the other as TEST SET

CARET HELPS!

```
set.seed(1)
in_training = createDataPartition(y = data1$gender
                                  , p = .8
                                  , list = FALSE
                                  )
in_training
```

```
##       Resample1
##  [1,]         1
##  [2,]         2
##  [3,]         4
##  [4,]         5
##  [5,]         6
##  [6,]         7
##  [7,]         8
##  [8,]        10
##  [9,]        12
## [10,]        13
## [11,]        14
## [12,]        15
## [13,]        16
## [14,]        17
## [15,]        18
```

SPLITTING THE DATA

```
training_data = data2[ in_training,]
test_data = data2[-in_training,]
```

	gender	salary	height
3	male	33225	179
9	male	40841	193
11	female	15039	152
20	female	30597	148

PIPELINE AGAIN

- define outcome (DONE)
- define features (DONE)
- build model (DONE)
  - but this time: on the TRAINING SET
- evaluate model
  - this time: on the TEST SET

Teach the SVM:

```
my_second_model = train(gender ~ .
                        , data = training_data
                        , method = "svmLinear"
                        )
```

Fit/test the SVM:

```
model_predictions = predict(my_second_model, test_data)
```

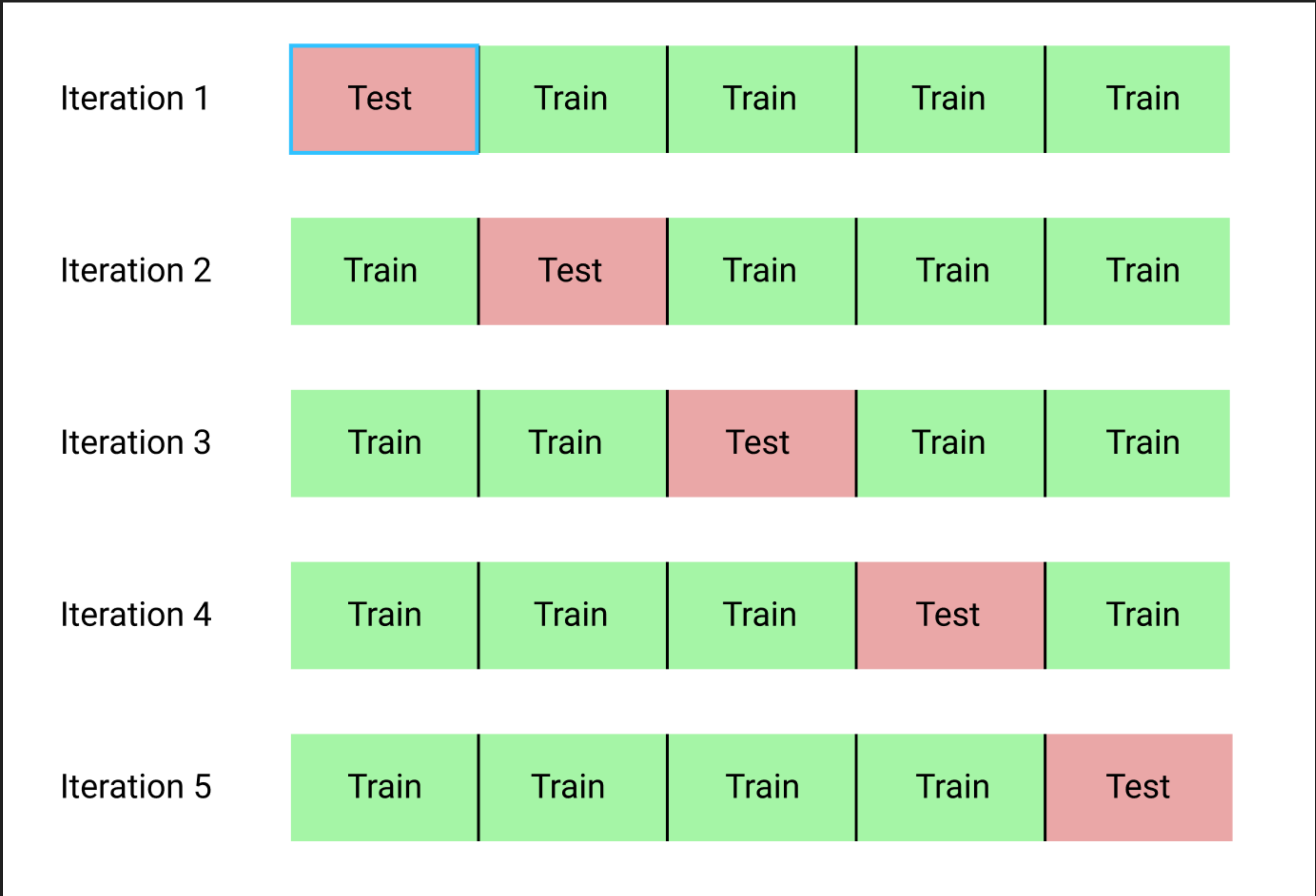
	female	male
female	2	0
male	0	2

BUT!

- our model might be really dependent on the training data
- we want to be more careful
- Can we do some kind of safeguarding in the training data?

CROSS-VALIDATION

K-fold cross-validation


Iteration 1
Test
Train
Train
Train
Train
Iteration 2
Train
Test
Train
Train
Train
Iteration 3
Train
Train
Test
Train
Train
Iteration 4
Train
Train
Train
Test
Train
Iteration 5
Train
Train
Train
Train
Test

Img source

SPECIFYING CV IN `CARET`

```
training_controls = trainControl(method="cv"
                                 , number = 4
                                 )

my_third_model = train(gender ~ .
                       , data = training_data
                       , trControl = training_controls
                       , method = "svmLinear"
                       )
```

```
my_third_model
```

```
## Support Vector Machines with Linear Kernel
##
## 16 samples
##  2 predictor
##  2 classes: 'female', 'male'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold)
## Summary of sample sizes: 12, 12, 12, 12
## Resampling results:
##
##   Accuracy  Kappa
##   0.75      0.5
##
## Tuning parameter 'C' was held constant at a value of 1
```

ASSESS THE CVED MODEL

```
model_predictions = predict(my_third_model, test_data)
```

	female	male
female	2	0
male	0	2

LET’S APPLY THIS!

Fakenews corpus: 1000 fake, 1000 real (data)

	including	ones	information	house	show	security	outcome
1	1	1	1	1	1	1	fake
2	0	0	0	0	0	0	fake
3	0	0	1	2	1	0	fake
1000	0	0	0	0	1	0	fake
1001	1	0	0	0	0	0	real
1002	0	0	0	0	0	0	real
1003	0	0	0	0	0	0	real

PROBLEM

- 1000 fake and 1000 real news items
- only source of information: text
- often fact-checking not available (yet)
- idea: linguistic traces help differentiate fake and real news

```r
dim(fake_news_data)
```

```
## [1] 2000  799
```

STEPWISE ML APPROACH

- the outcome variable?
- the features?
- the algorithm?
- the train/test split?
- the training set cross-validation?

MODEL 1

	Model 1
outcome	fake vs real
features	ngram freqs.
algorithm	Linear SVM
train/test	80/20
Cross-val.	10-fold

STEP 1

Partition the data

```
set.seed(2019)
in_training = createDataPartition(y = fake_news_data$outcome
                                  , p = .8 # <-- split value
                                  , list = FALSE
                                  )
training_data = fake_news_data[ in_training,]
test_data = fake_news_data[-in_training,]
```

Training data

Var1	Freq
fake	800
real	800

STEP 2

Define training controls

```
training_controls = trainControl(method="cv"
                                 , number = 10
                                 )
```

STEP 3

Train the model

```
fakenews_model_1 = train(outcome ~ .
                        , data = training_data
                        , trControl = training_controls
                        , method = "svmLinear"
                        )
```

STEP 4

Fit the model

```
model_1.predictions = predict(fakenews_model_1, test_data)
```

	fake	**real**
fake	159	41
real	42	158

$$(159+158)/400 = 0.73$$

THE STRENGTH OF `CARET`...

Let's see whether we can do better

	Model 1	Model 2
outcome	fake vs real	~
features	ngram freqs.	~
algorithm	Linear SVM	~
train/test	80/20	60/40
Cross-val.	10-fold	5-fold

MODEL 2

Step 1: Splitting the data

```
set.seed(2019)
in_training = createDataPartition(y = fake_news_data$outcome
                                  , p = .6 # <-- split value
                                  , list = FALSE
                                  )
training_data = fake_news_data[ in_training,]
test_data = fake_news_data[-in_training,]
```

Step 2: Define training controls

```
training_controls = trainControl(method="cv"
                                , number = 5
                                )
```

Step 3: Train the model

```
fakenews_model_2 = train(outcome ~ .
                        , data = training_data
                        , trControl = training_controls
                        , method = "svmLinear"
                        )
```

Step 4: Fit the model

```
model_2.predictions = predict(fakenews_model_2, test_data)
```

	fake	**real**
fake	329	71
real	91	309

$$(329+309)/800 = 0.80$$

LOOKING A STEP FURTHER

What's driving the classification?

```r
varImp(fakenews_model_1_)
```

```
## ROC curve variable importance
##
##   only 20 most important variables shown (out of 798)
##
##        Importance
## said       100.00
## first       82.70
## last        77.06
## two         73.30
## year        67.09
## years       63.15
## still       58.76
## also        57.69
## three       53.95
## thats       53.91
## one         53.67
```

IMPORTANT FEATURES

“said”

```r
tapply(training_data$said, training_data$outcome, mean)
```

```
##       1       0
## 0.93875 2.97625
```

“first”

```r
tapply(training_data$first, training_data$outcome, mean)
```

```
##       1       0
## 0.48875 1.16000
```

MAKING FULL USE OF `CARET`

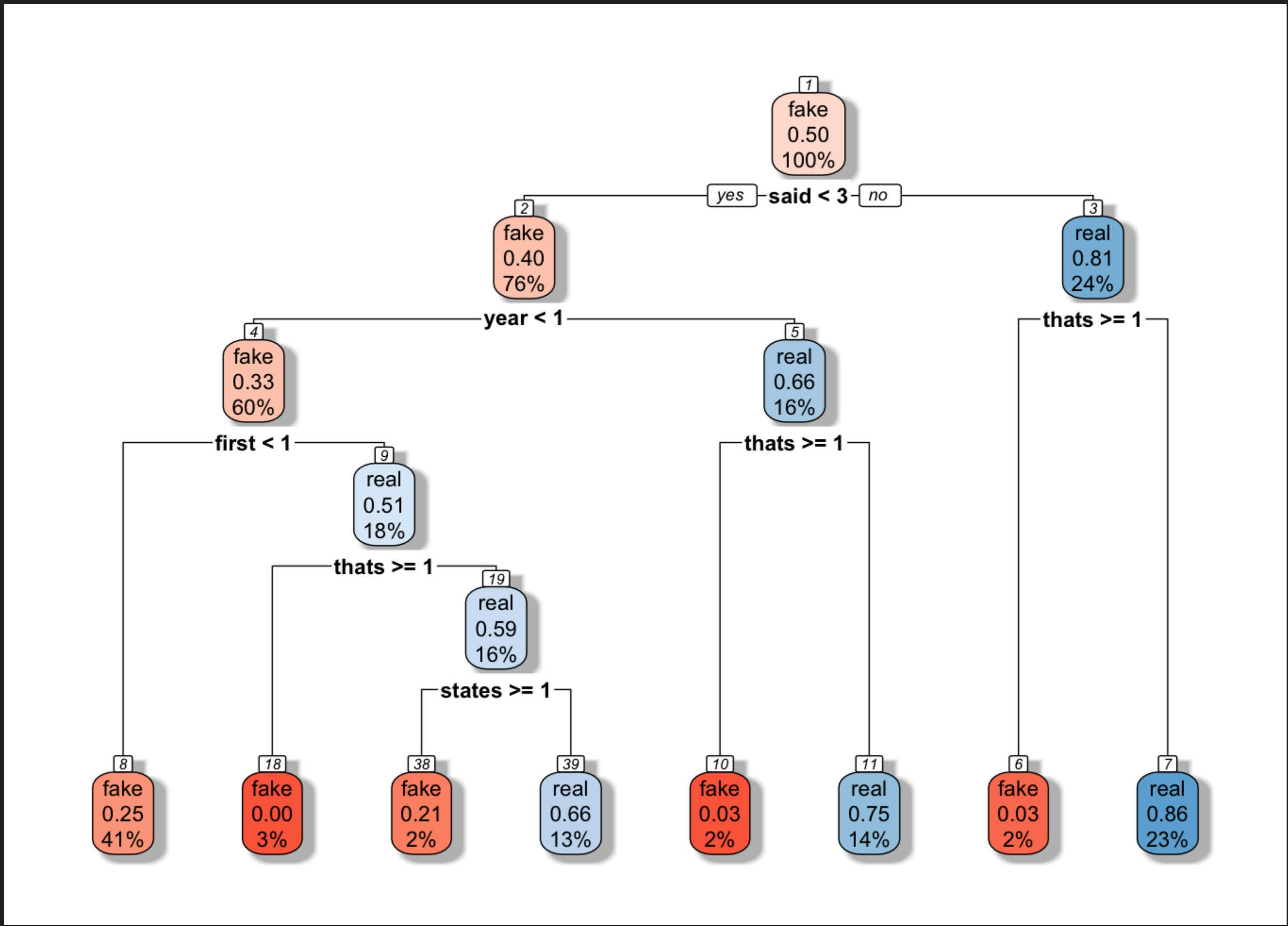
- what if we want to use a different classification algorithm?

Selection of models –> https://topepo.github.io/caret/available-models.html

INTERMEZZO TO DIFFERENT ALGORITHMS

- Support Vector Machine video
- Decision Trees
- Random Forests
- worth knowing:
  - Naive Bayes
  - Logistic regression
  - kNN

DECISION TREES



RANDOM FORESTS

- selects random set of training data
  - builds decision tree
- = many trees = forest
- many random trees = random forest
- averaging the trees (voting)

MODEL 3

	Model 1	Model 2	Model 3
outcome	fake vs real	~	~
features	ngram freqs.	~	~
algorithm	Linear SVM	~	Random Forest
train/test	80/20	60/40	70/30
Cross-val.	10-fold	5-fold	2x Repeated 5-fold

MODEL 3

(skipping data splitting here)

```
training_controls = trainControl(method="repeatedcv"
                                 , number = 5
                                 , repeats = 2
                                 )
```

MODEL 3

```
fakenews_model_3 = train(outcome ~ .
                        , data = training_data
                        , trControl = training_controls
                        , method = "ranger"
                        )
```

MODEL 3

```
## Random Forest
##
## 560 samples
## 798 predictors
##   2 classes: 'fake', 'real'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 448, 448, 448, 448, 448, 448, ...
## Resampling results across tuning parameters:
##
##   mtry  splitrule   Accuracy   Kappa
##     2   gini        0.7464286  0.4928571
##     2   extratrees  0.7366071  0.4732143
##    39   gini        0.8250000  0.6500000
##    39   extratrees  0.7901786  0.5803571
```

MODEL 3

Make predictions

```
model_3.predictions = predict(fakenews_model_3, test_data)
```

	fake	**real**
fake	94	26
real	20	100

(90+108)/240 = 0.83

RECAP

- Types of machine learning
- Supervised ML
- Cross-validation
- Using caret

OUTLOOK

Tutorial tomorrow

Homework: Replication of fake news classification

Week 7: Machine learning 2

Next week: Unsupervised learning + performance metrics

END