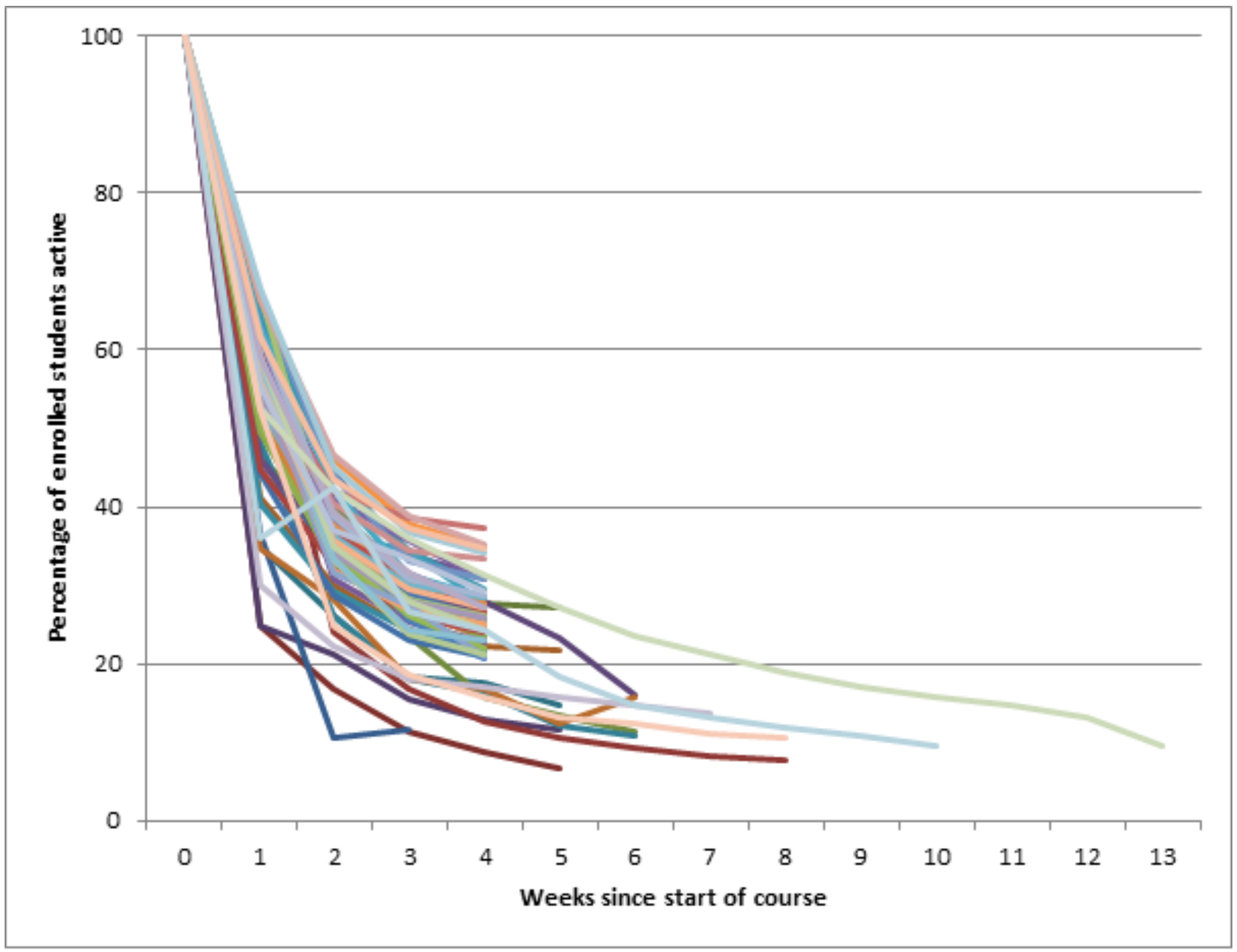
Prediction

- There are times when we want to automate a process in education
- Want to be able to predict what a student might do in the future: next question, next move in a game, drop out

Prediction



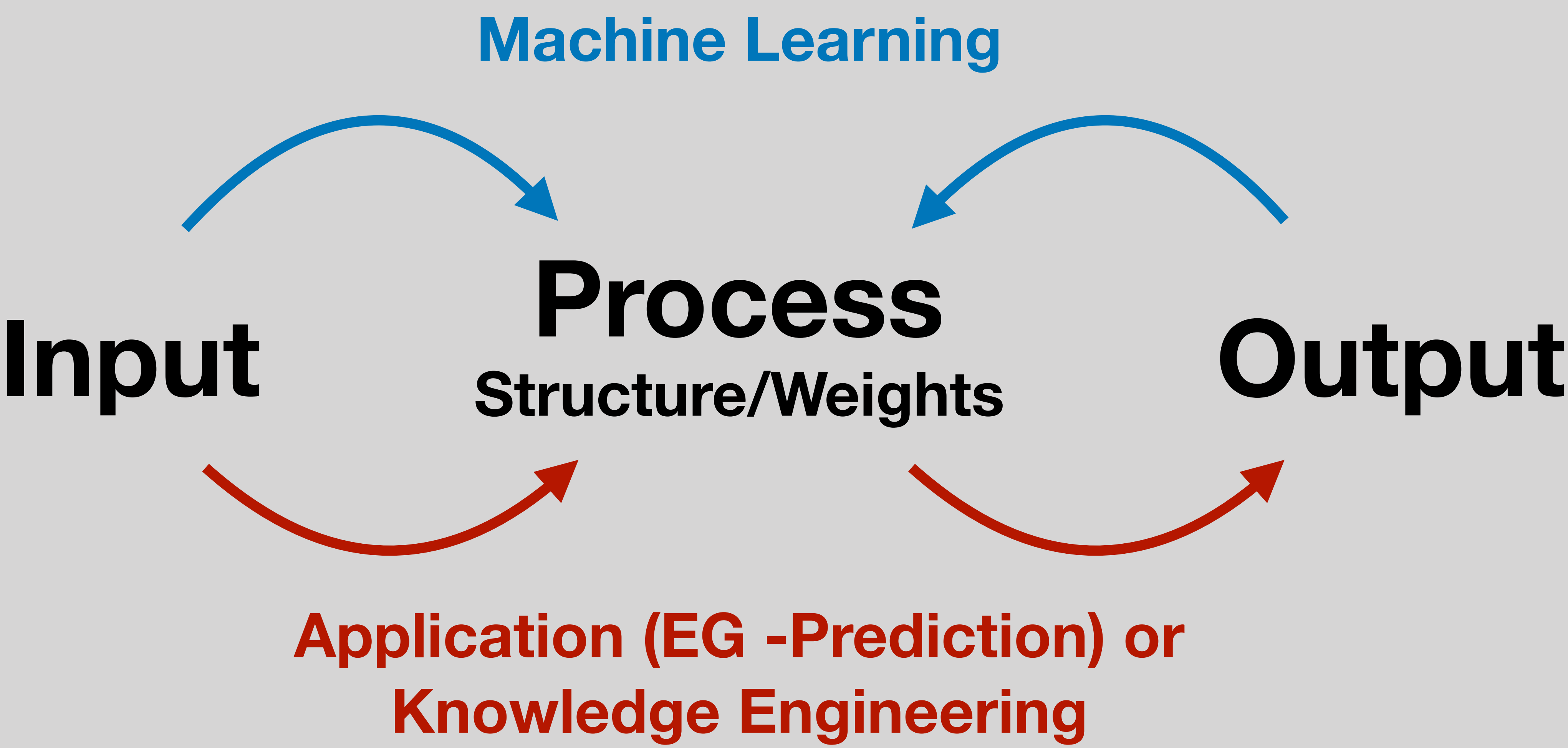
K Jordan, Open University, 2013

summit
public schools

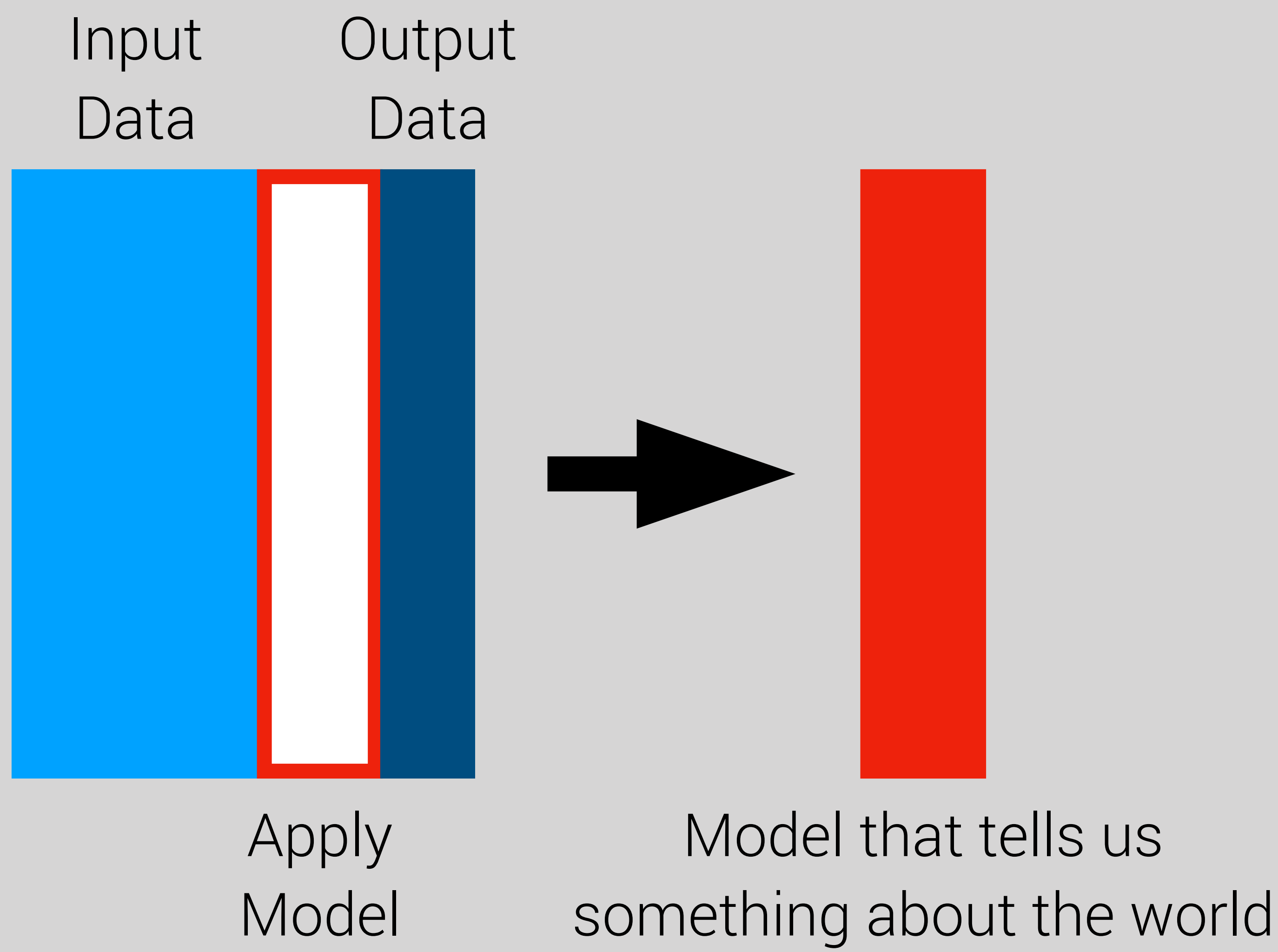
CARNEGIE LEARNING

ASSISTments

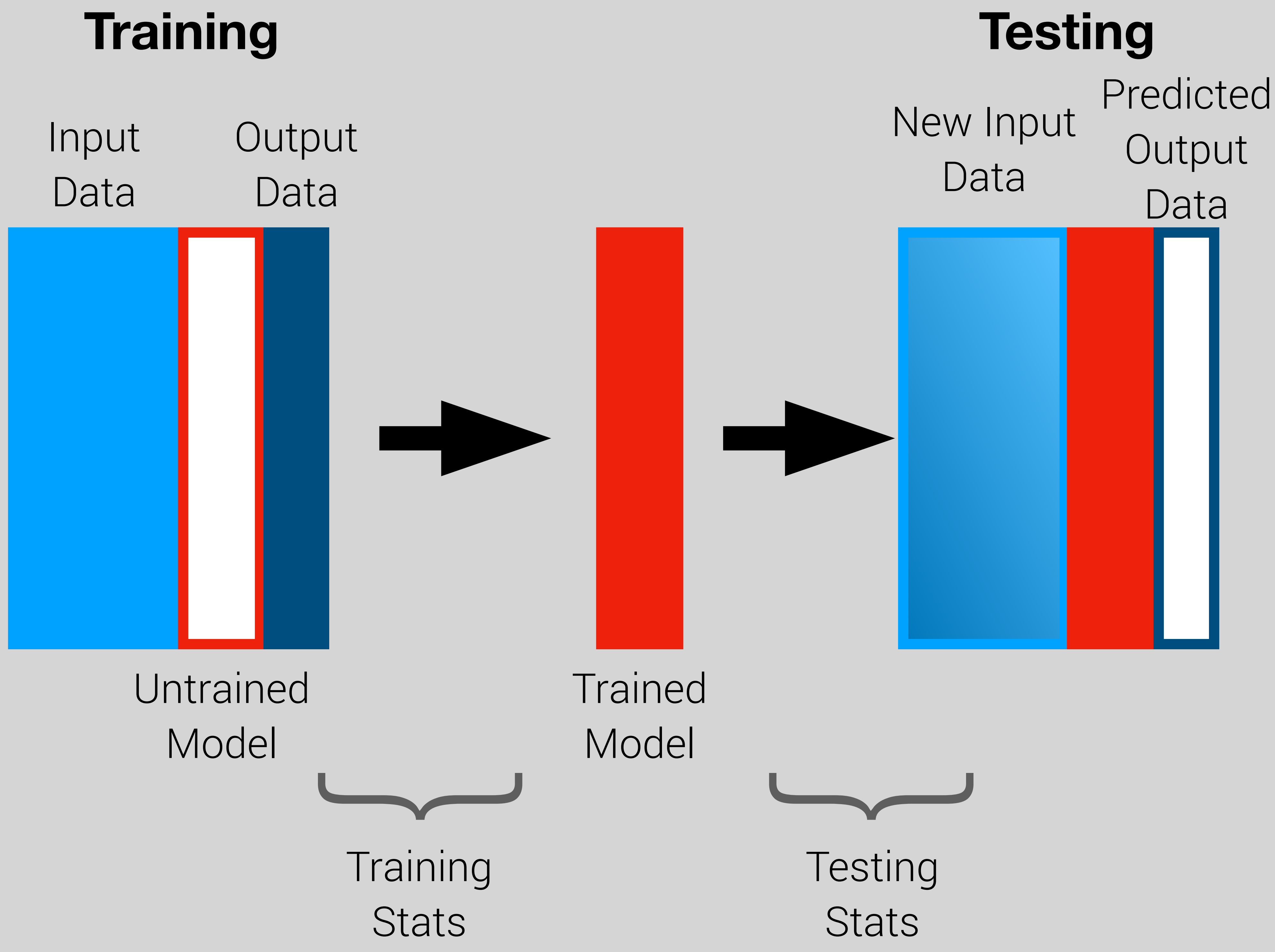
Machine Learning



Educational Statistics



Machine Learning



Classification Confusion Matrix

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Actual Class
(Observations)</td></tr>
<tr><td>P</td><td>N</td></tr>
<tr><td rowspan="2">Predicted Class
(Predictions)</td><td>P</td><td>TP</td><td>FP</td></tr>
<tr><td>N</td><td>FN</td><td>TN</td></tr>
</table>

Classification Confusion Matrix

		Actual Class	
		P	**N**
Predicted Class	**P**	TP	FP
	N	FN	TN

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

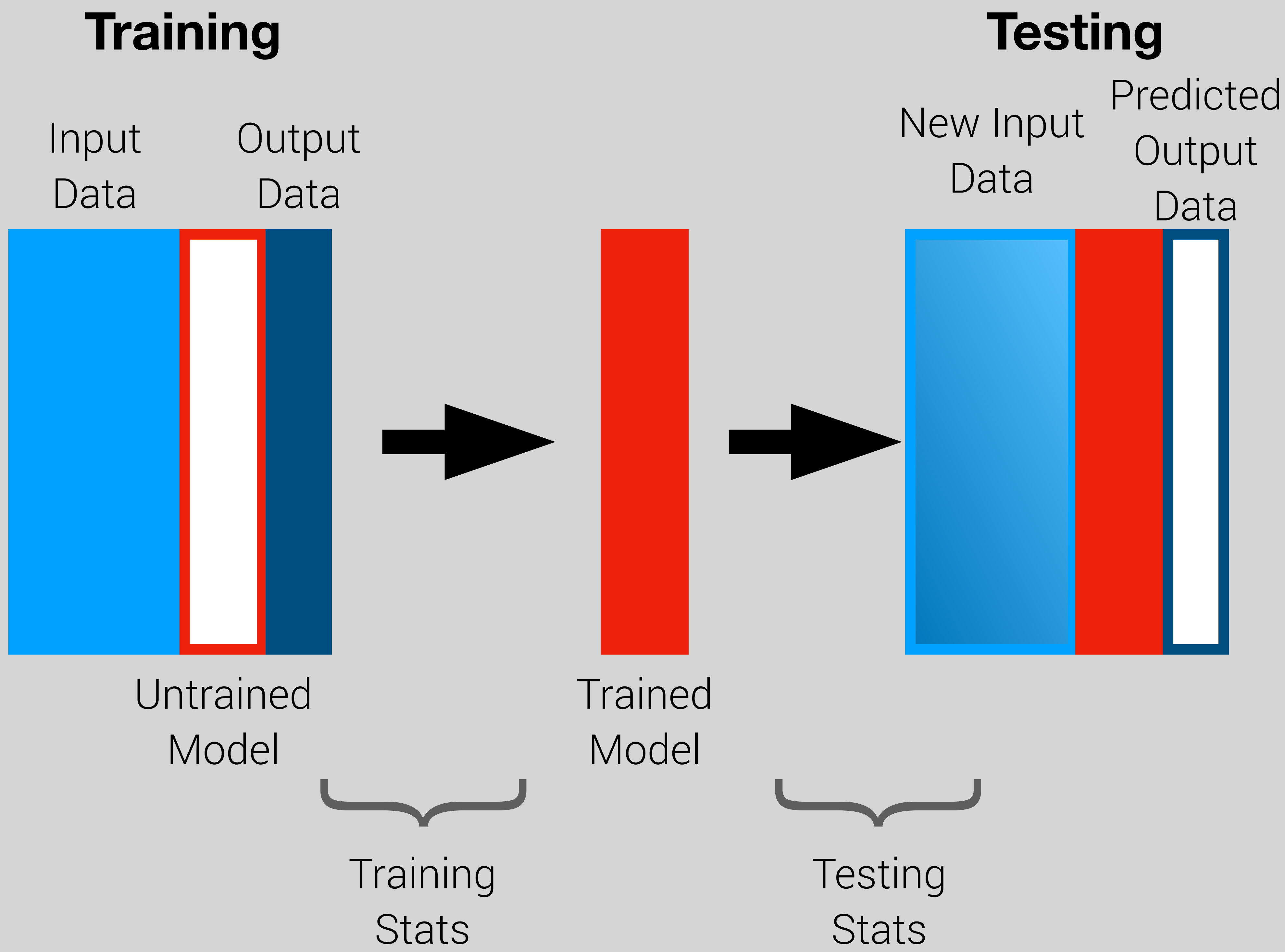
$$\text{Sensitivity/Recall/TPR} = \frac{TP}{TP + FN}$$

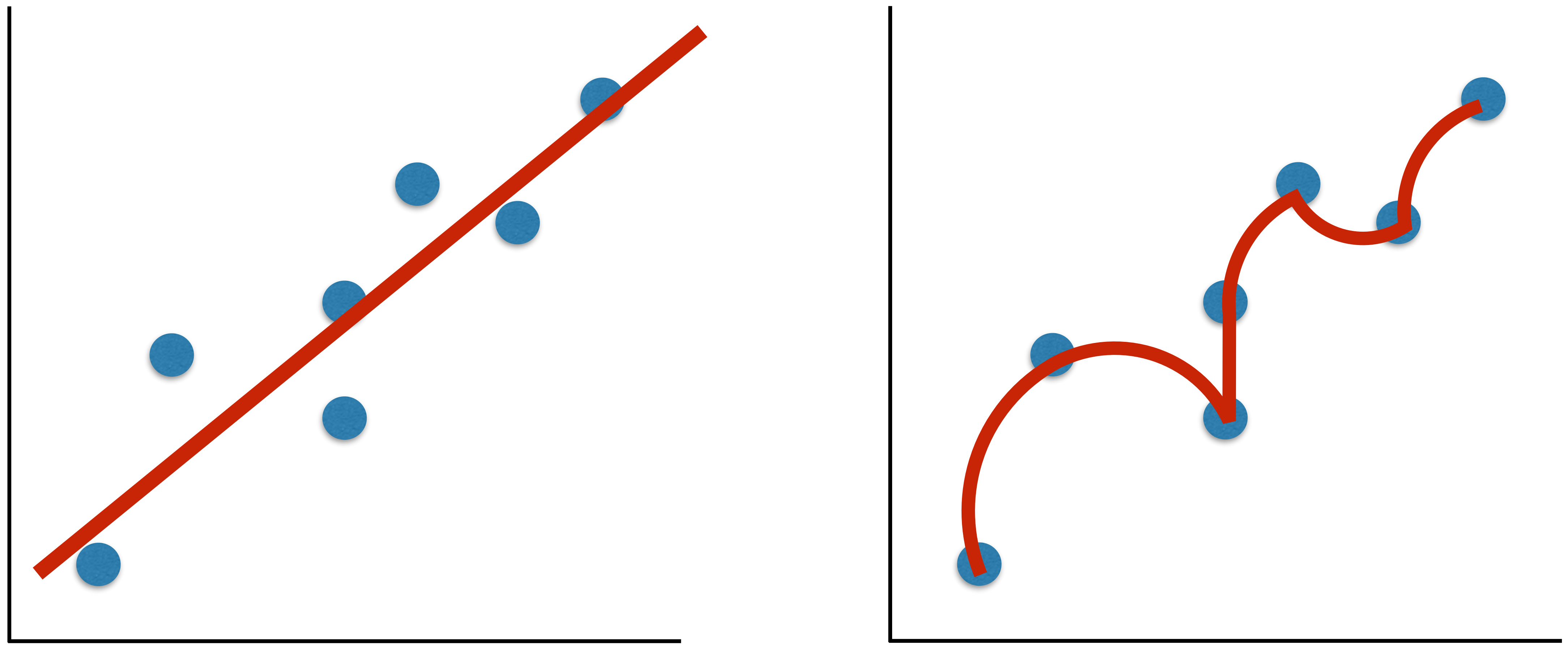
$$\text{Specificity/Selectivity/TNR} = \frac{TN}{TN + FP}$$

$$\text{Precision/Positive Predictive Value (PPV)} = \frac{TP}{TP + FP}$$

$$\text{F1} = \frac{2TP}{2TP + FP + FN}$$

Machine Learning





Which is more “accurate”?

Which is more “useful”?

How can we tell?

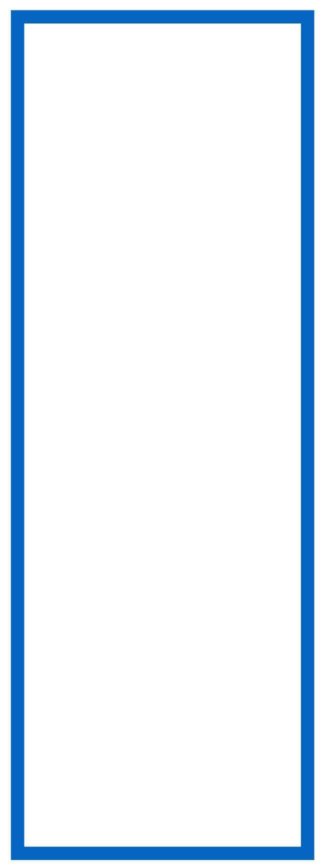
Cross Validation

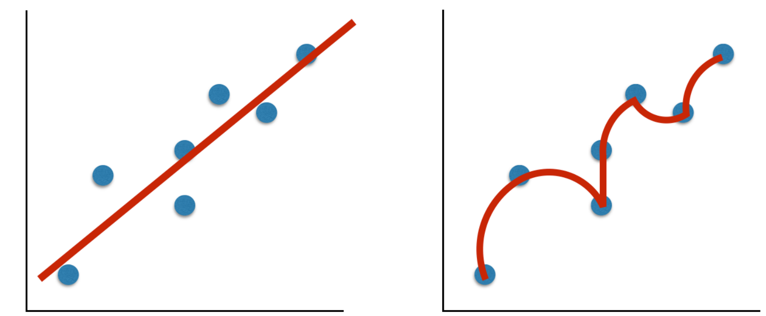
- Estimate how accurately a predictive model will perform in practice
- Give an insight on how the model will generalize to an independent dataset

No Validation

Sample

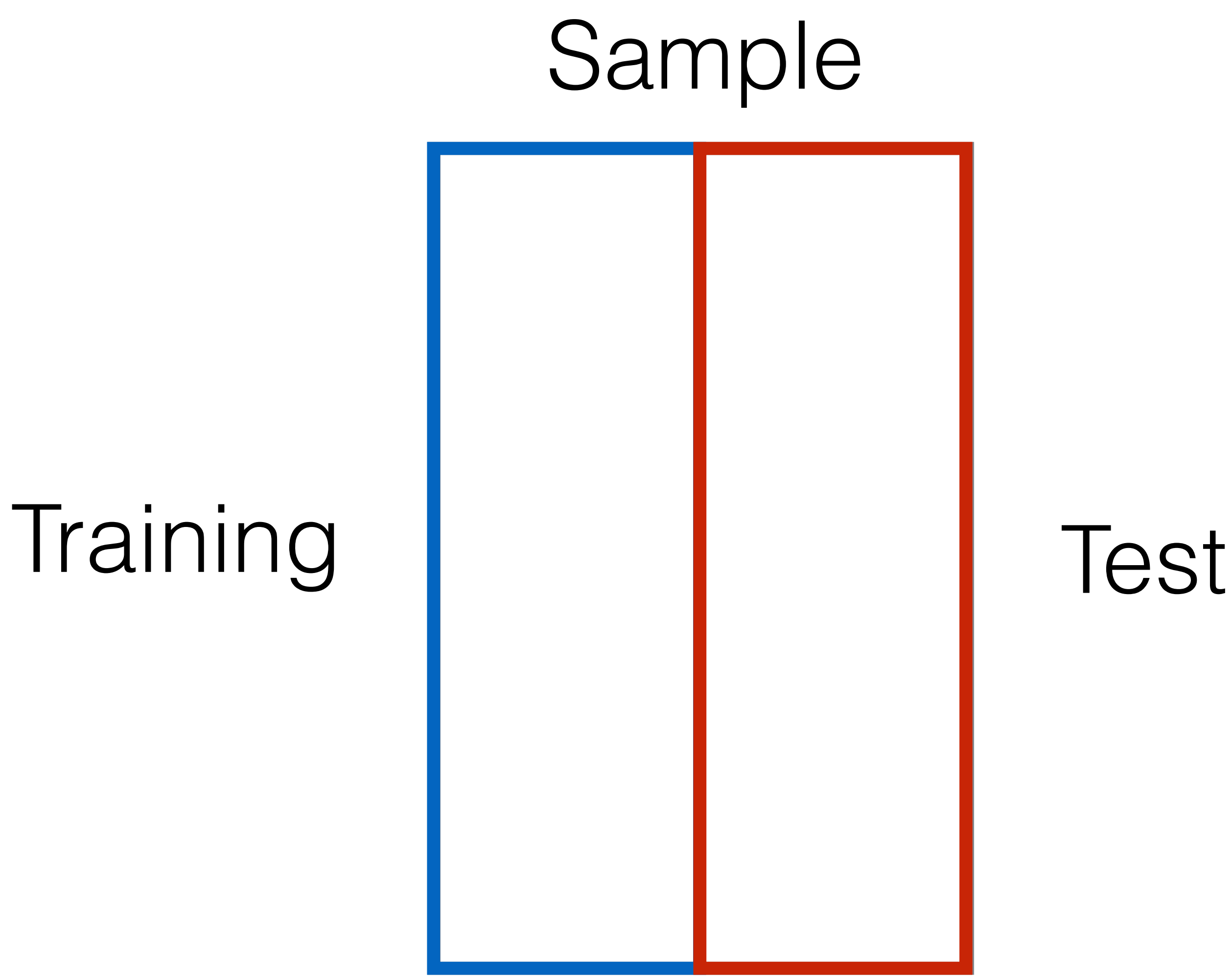
Training

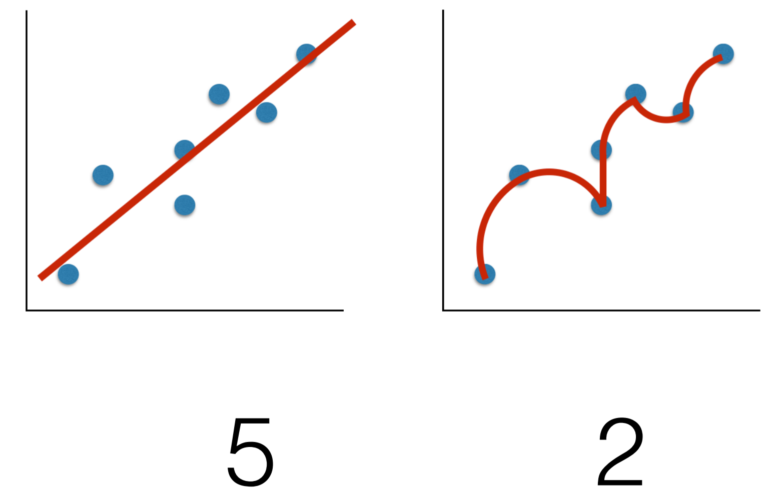




Problem: Can’t compare generalizability of models

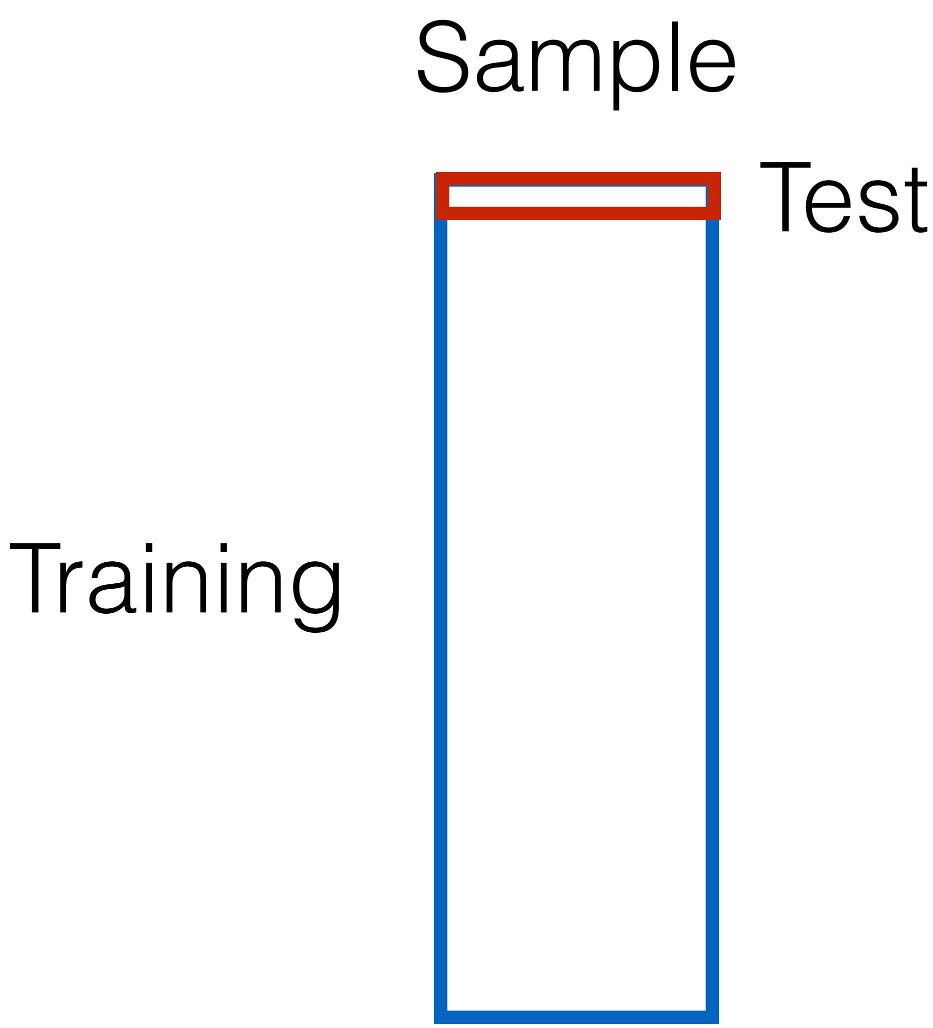
Hold-out Validation

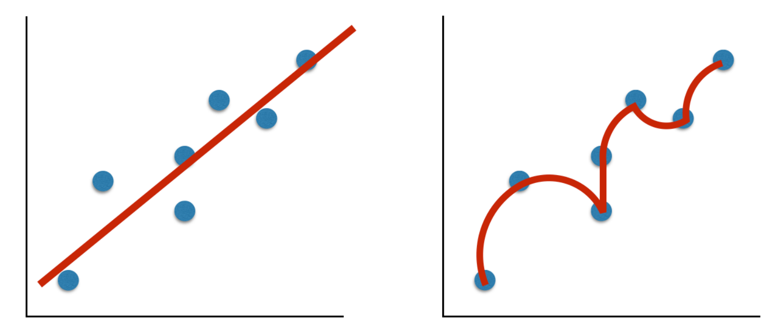




Problem: very dependent on which data are in each group

Hold One-out Validation



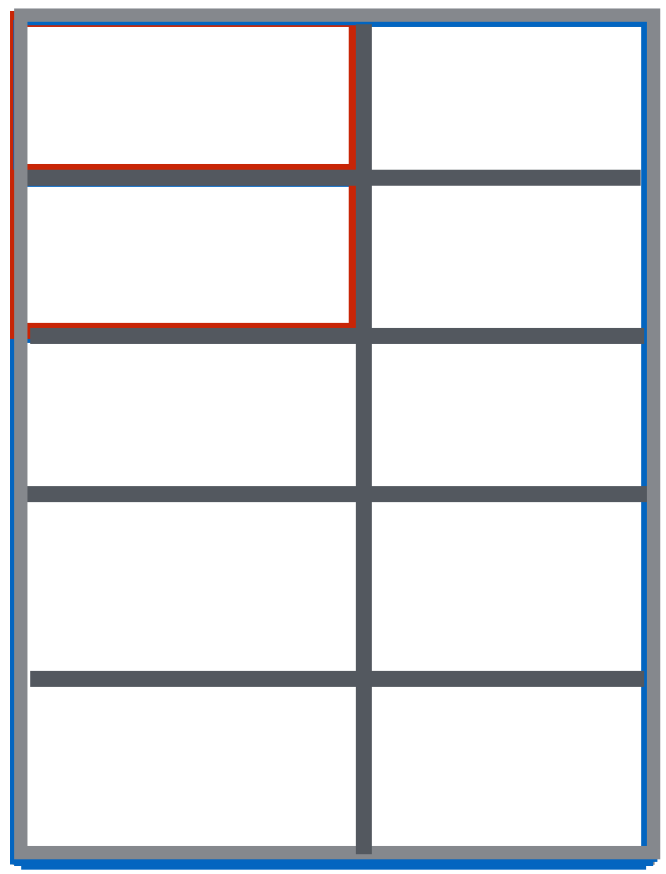


Problem: very computationally expensive

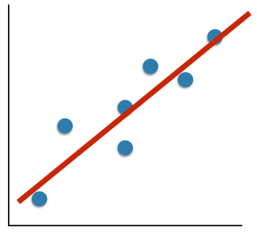
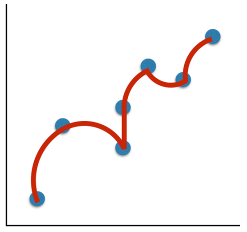
K-Fold Cross Validation

Sample

Test 1
Test 2
Test 3
Test 4
Test 5

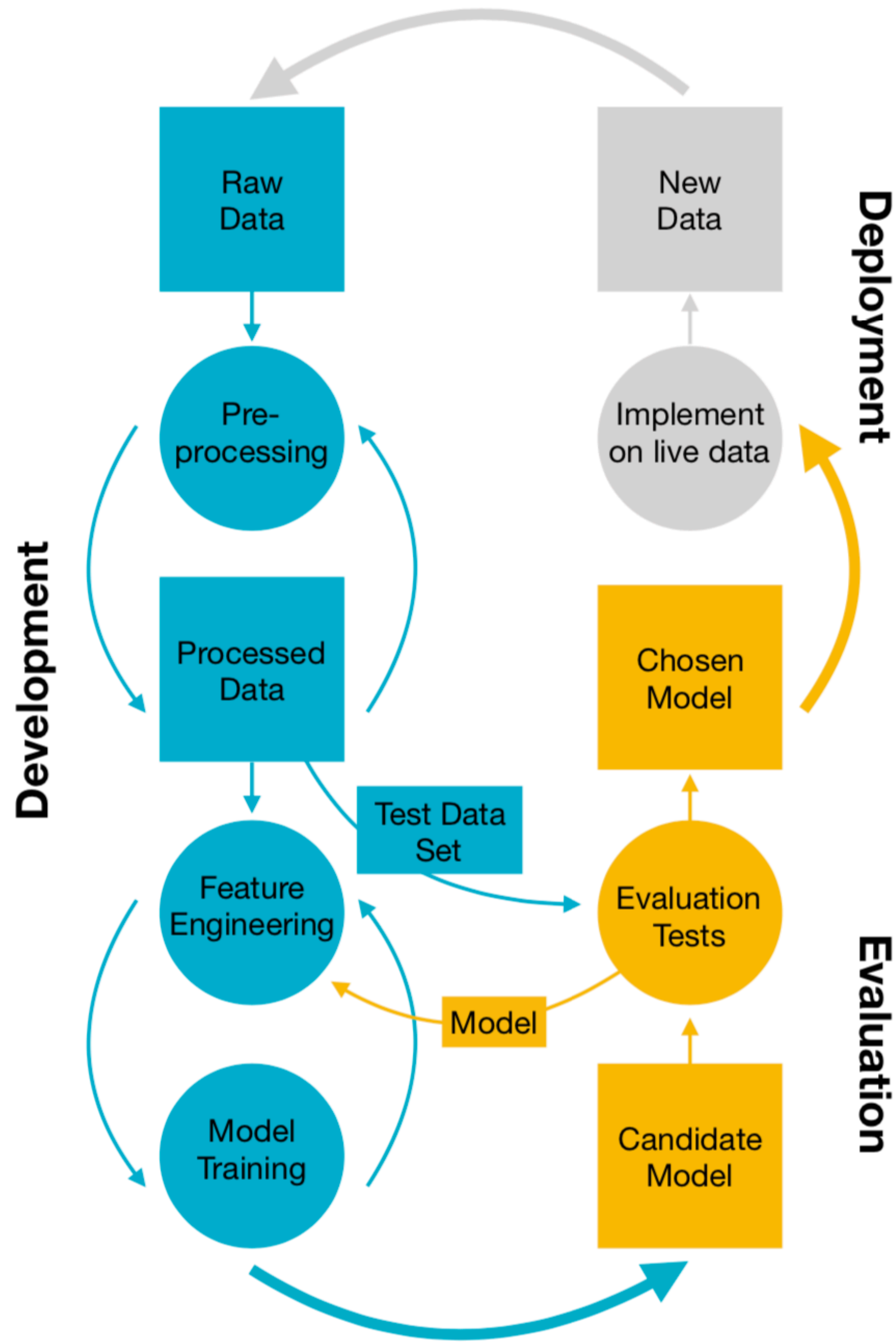


Training 1
Training 2
Training 3
Training 4
Training 5

5	2
4	2
3	1
5	4
4	2
4.2	2.2

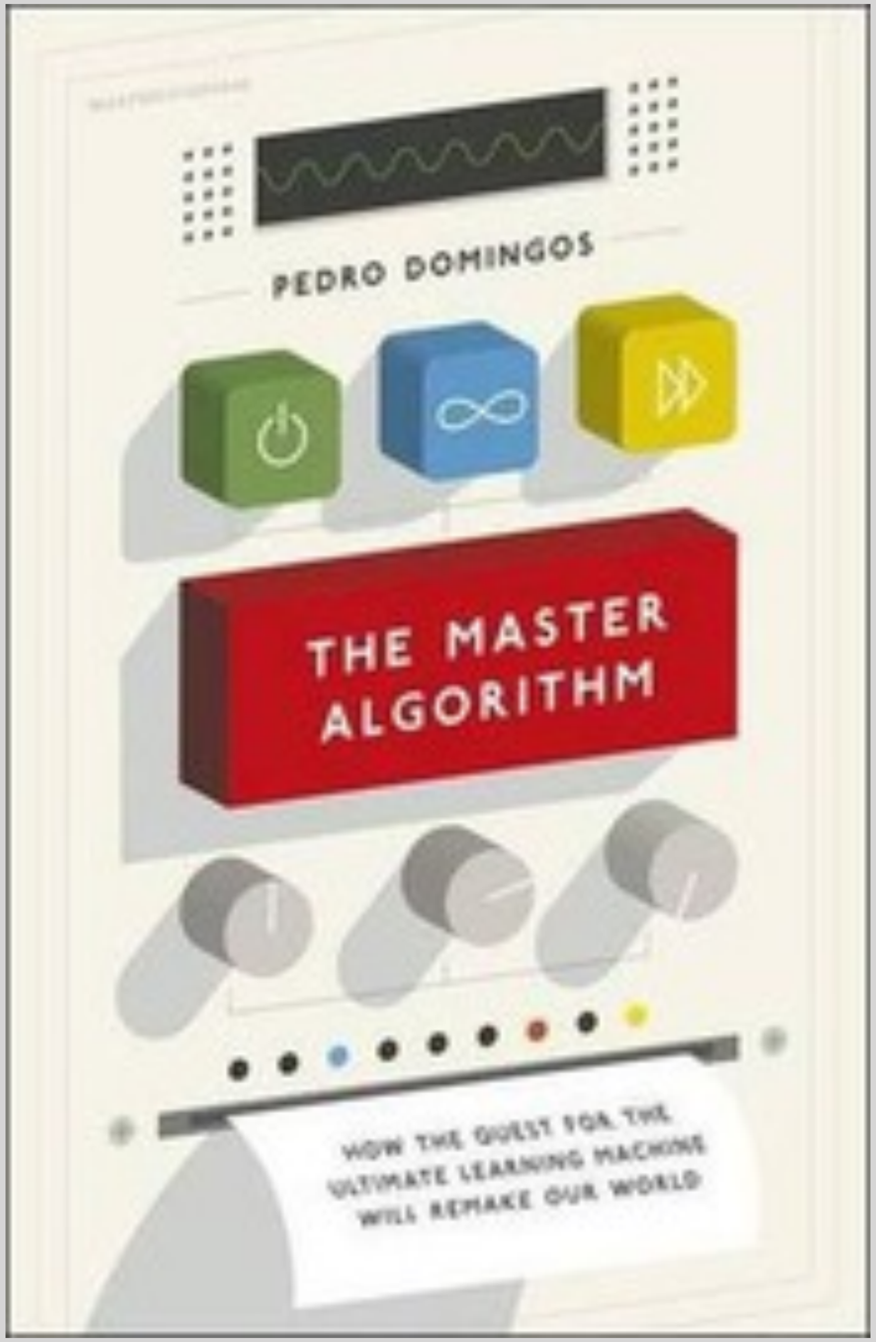
Calculate how accurate we are in each “fold” and average the answer



Raw Data
New Data
Deployment
Pre-processing
Implement on live data
Development
Processed Data
Chosen Model
Test Data Set
Feature Engineering
Evaluation Tests
Model
Evaluation
Model Training
Candidate Model

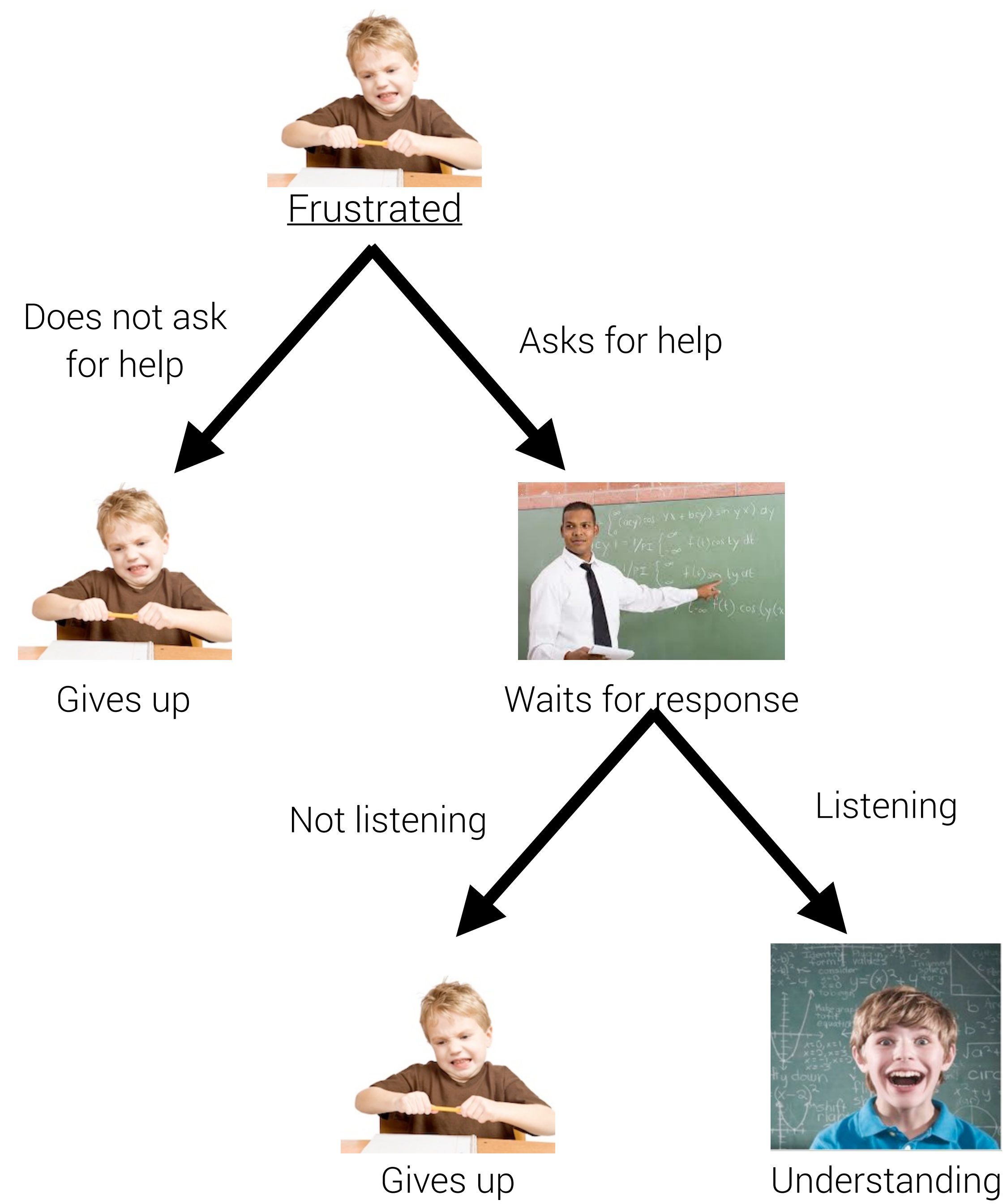
Five Tribes

- Symbolists
- Connectionists
- Evolutionaries
- Bayesians
- Analogizers



Classification Tree

- Decision tree
- Map observations (branches) onto classes (leaves)
- Tree describes the data but can be used as classification
- EG: student states = leaves, student actions = branches



Machine Learning

Input

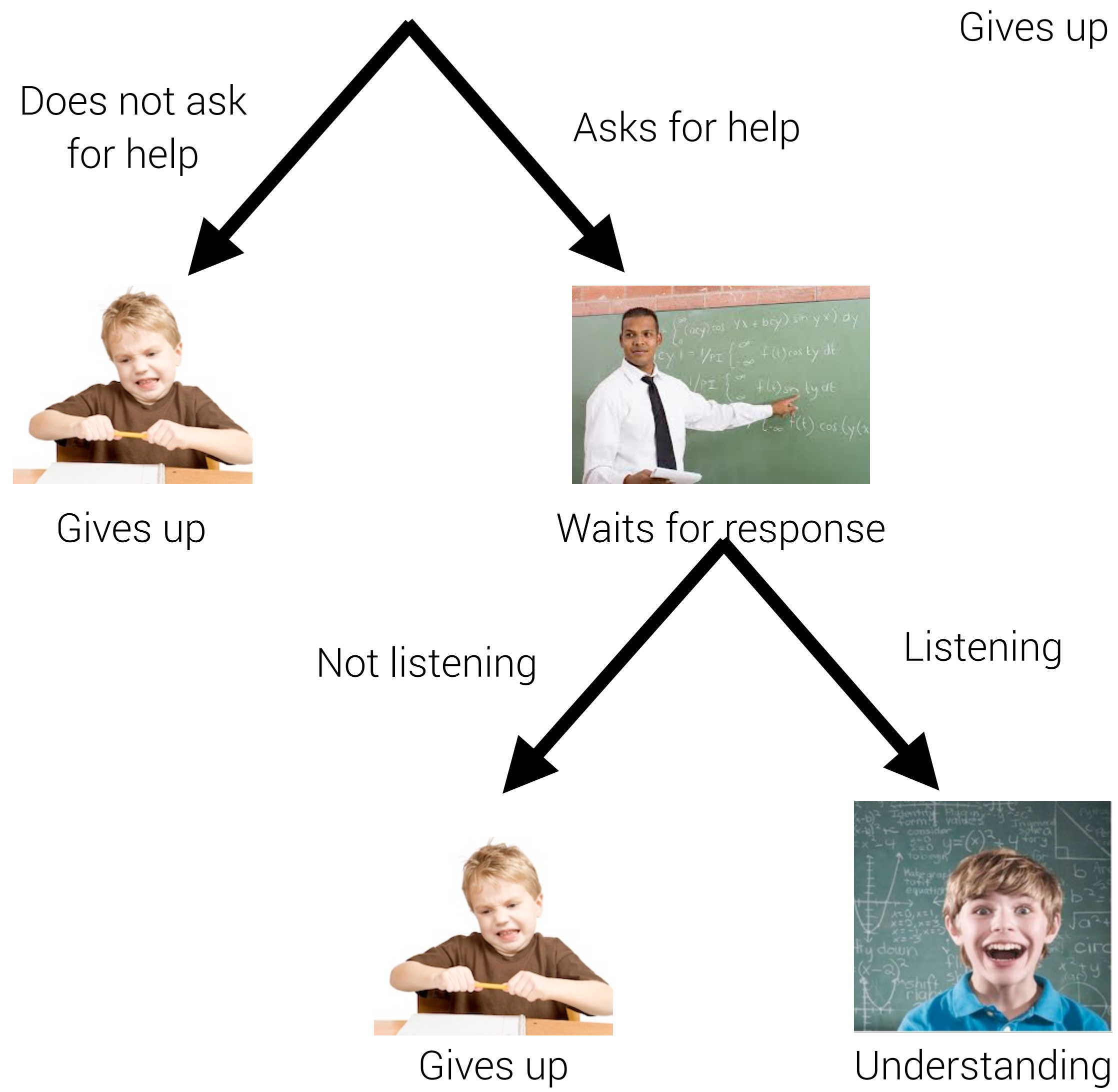
Does not ask
for help

Asks for help

Not listening

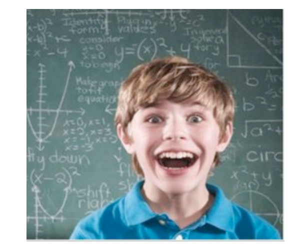
Listening

Process
Structure/Weights



Output

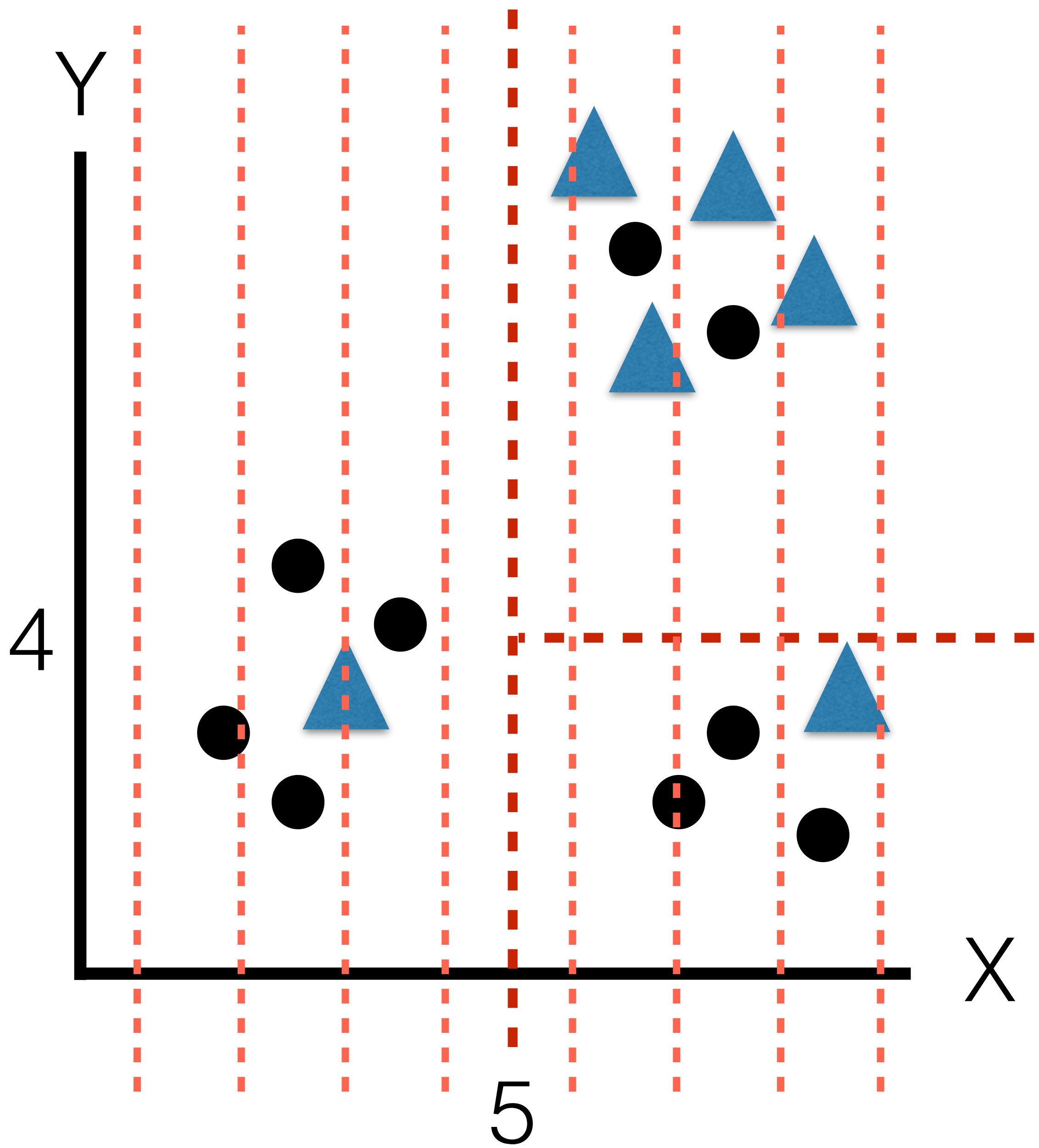
Gives up

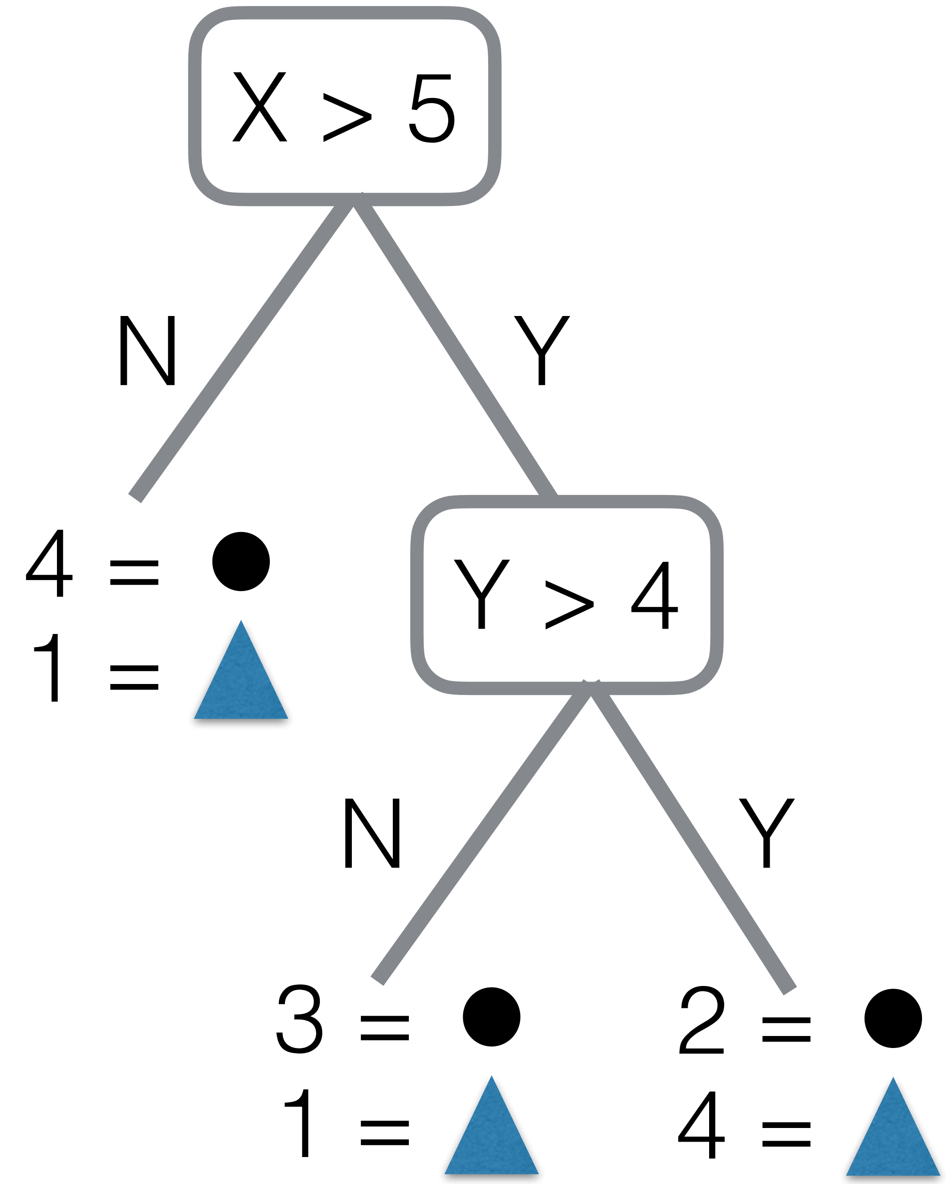


Understanding

Binary Classification Tree

* Minimize the error





caret

- Standard syntax for comparing many models
- Generate training and testing data sets
- Run several model types
- Run resampling algorithms and alter parameters to generate the best model
- Compare using the same diagnostic metrics
- https://topepo.github.io/caret/

caret

Generate Training/Test Data Sets

```
trainData <- createDataPartition(
  y = data$thing, ## the outcome data are needed
  p = .75, ## The percentage of data in the
training set
  list = FALSE)

#Generates a list of index numbers for the sample

training <- DATA[ trainData,]
testing  <-DATA[-trainData,]
```

caret

K-Fold Cross Validation

```
ctrl <- trainControl(method = "cv", repeats = 3)
```

caret

Train Model

```
fit1 <- train(
  thing ~ .,
  data = training,
  method = “model",
  preProc = c("center", “scale")## Center and scale
the predictors for the training set and all future
samples.
  trControl = ctrl #add cross validation specs
  metric = "cp"
)
```

caret

Test Model

```
pred1 <- predict(fit1, newdata = testing)

confusionMatrix(data = pred1, DATA$thing)
```

Project

Train and test three tree-based models (CART, Conditional Inference Trees and C50) using data from the University of Michigan Open Data Set.