CS4225/CS5425 Big Data Systems for Data Science

Basic ML and Spark MLLib

Bryan Hooi
School of Computing
National University of Singapore
bhooi@comp.nus.edu.sg

School of Computing

Announcements

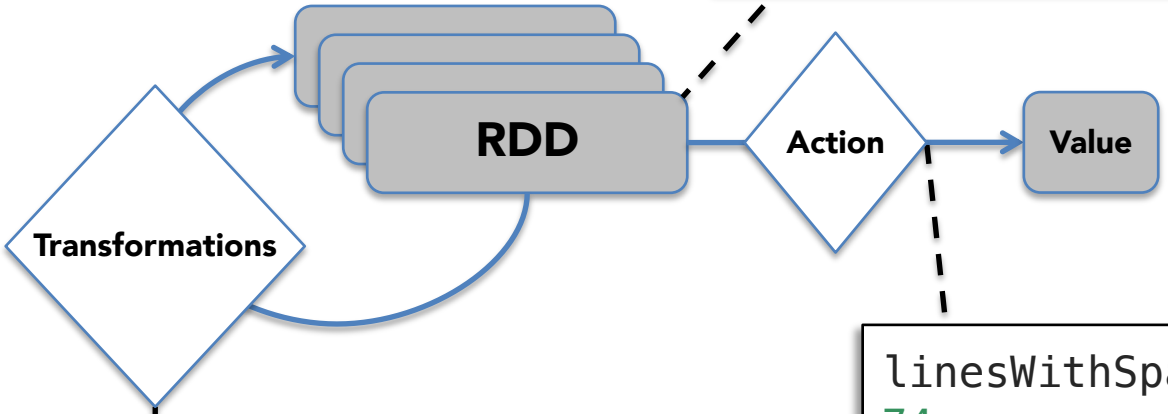
- HW1 is extended to 13 Oct (Wed) 11.59pm.
  - For any issues, feel free to email me / post on LumiNUS / visit office hours
- HW2 is due 31 Oct 11.59pm

Week	Date	Topics	Tutorial	Due Dates
1	12 Aug	Overview and Introduction		
2	19 Aug	MapReduce - Introduction		
3	26 Aug	MapReduce and Relational Databases		
4	2 Sep	MapReduce and Data Mining	Tutorial: Hadoop	
5	9 Sep	NoSQL Overview 1		Assignment 1 released
6	16 Sep	NoSQL Overview 2		
Recess				
7	30 Sep	Apache Spark 1	Tutorial: NoSQL & Spark	Assignment 2 released
8	7 Oct	Apache Spark 2		Assignment 1 due (13 Oct)
9	14 Oct	Large Graph Processing 1	Tutorial: Graph Processing	
10	21 Oct	Large Graph Processing 2		
11	28 Oct	Stream Processing	Tutorial: Stream Processing	Assignment 2 due (31 Oct)
12	4 Nov	Deepavali – No Class		
13	11 Nov	**Test**		

Recap: Working with RDDs

Note: this reads the file on each worker node in parallel, not on the driver node

```
textFile =
sc.textFile("SomeFile.txt")
```




```
linesWithSpark.count()
74

linesWithSpark.first()
# Apache Spark
```

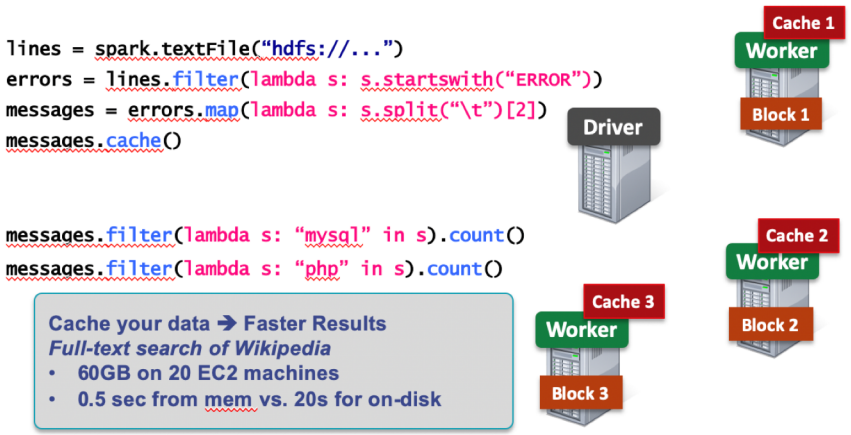
```
linesWithSpark =
textFile.filter(lambda line:
"Spark" in line)
```

Recap: Caching

- `cache()`: saves an RDD to memory (of each worker node).
- `persist(options)`: can be used to save an RDD to memory disk, or off-heap memory
- When should we cache or not cache an RDD?
  - When it is expensive to compute and needs to be re-used multiple times.
  - If worker nodes have not enough memory, they will evict the "least recently used" RDDs. So, be aware of memory limitations when caching.
  - If not enough memory to cache an RDD, it will be ignored. If using persist(), the option 'MEMORY_AND_DISK' will store it in disk in this case (see http://spark.apache.org/faq.html)

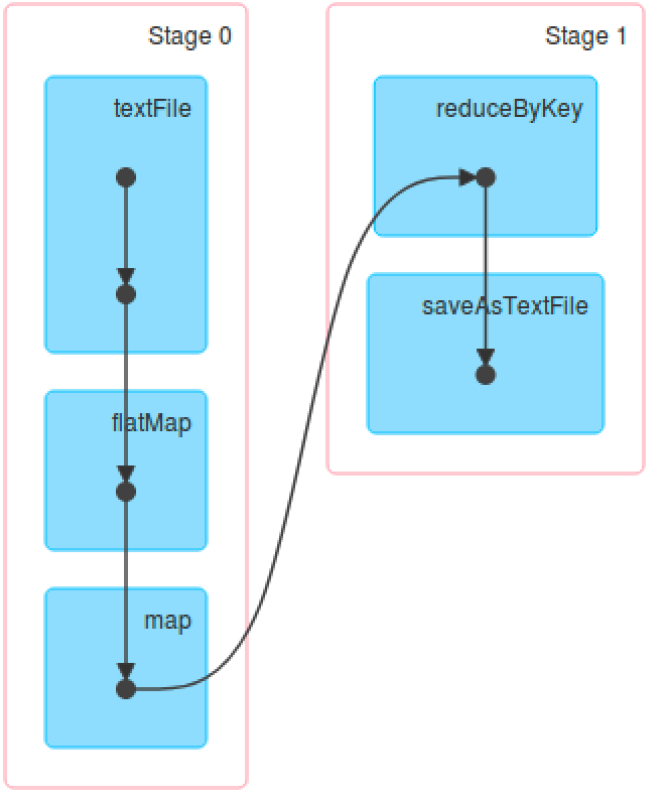
Does my data need to fit in memory to use Spark?

No. Spark's operators spill data to disk if it does not fit in memory, allowing it to run well on any sized data. Likewise, cached datasets that do not fit in memory are either spilled to disk or recomputed on the fly when needed, as determined by the RDD's storage level.



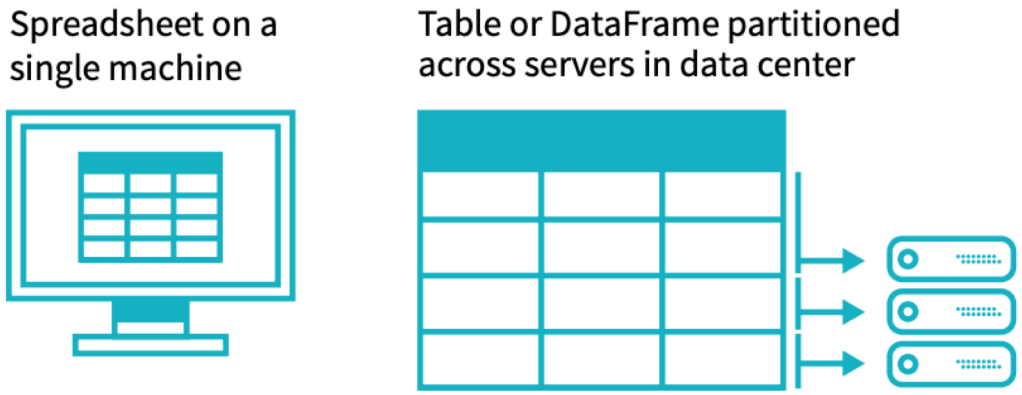
Recap: Lineage and Fault Tolerance

- Unlike Hadoop, Spark does not use replication to allow fault tolerance. Why?
  - Spark tries to store all the data in memory, not disk. Memory capacity is much more limited than disk, so simply duplicating all data is expensive.
- **Lineage approach**: if a worker node goes down, we replace it by a new worker node, and use the graph (DAG) to recompute the data in the lost partition.
  - Note that we only need to recompute the RDDs from the lost partition.



Recap: DataFrames

- A DataFrame represents a table of data, similar to tables in SQL, or DataFrames in pandas.
- Compared to RDDs, this is a higher level interface, e.g. it has transformations that resemble SQL operations.
  - DataFrames (and Datasets) are the recommended interface for working with Spark – they are easier to use than RDDs and almost all tasks can be done with them, while only rarely using the RDD functions.
  - However, all DataFrame operations are still ultimately compiled down to RDD operations by Spark.



Clarification: Confusing Syntax?

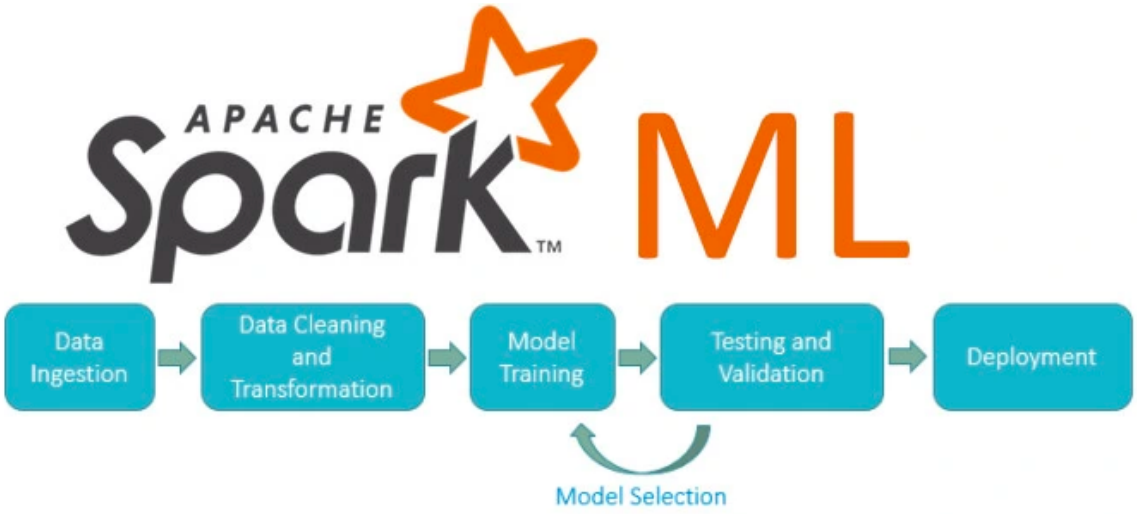
```
flightData2015 = spark\
.read\
.option("inferSchema", "true")\
.option("header", "true")\
.csv("/mnt/defg/flight-data/csv/2015-summary.csv")
```

```
flightData2015\
.groupBy("DEST_COUNTRY_NAME")\
.sum("count")\
.withColumnRenamed("sum(count)", "destination_total")\
.sort(desc("destination_total"))\
.limit(5)
```

- Called “method chaining”, common in Scala
- Works because each method returns the object itself
- Thus, a series of method calls can be chained together

Today's Plan

- **Supervised Machine Learning Basics**
  - Introduction
  - Preprocessing
  - Model Training and Testing
  - Evaluation
- **Spark MLLib**

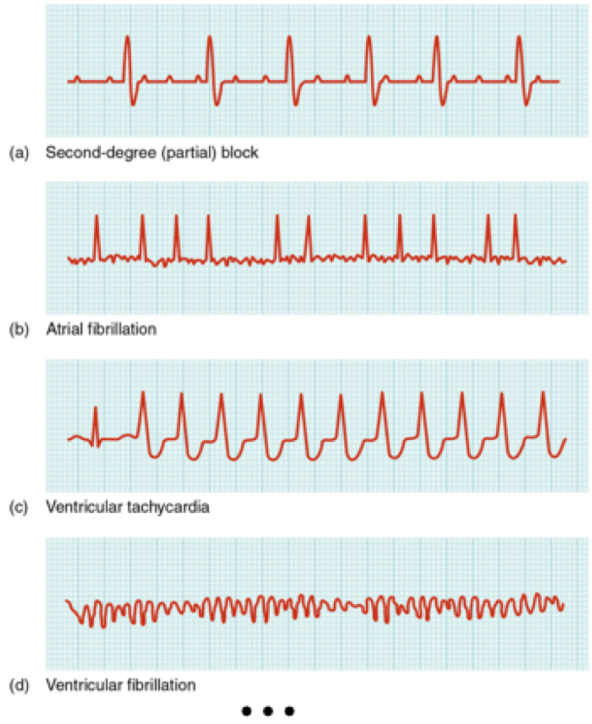


Example: Heart Rhythm Classification

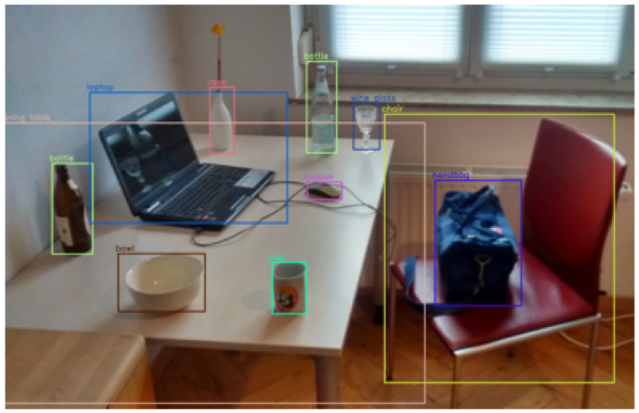
Medical ECG devices

Wearable devices

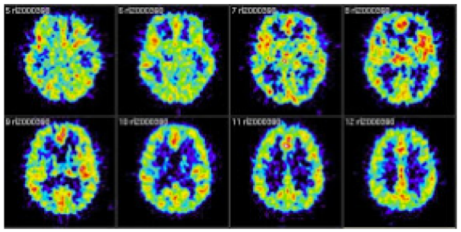
Classification



More Applications



Object Recognition



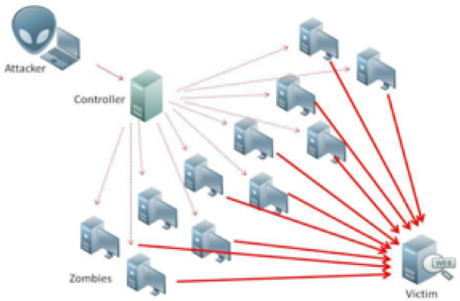
Medical Imaging

PayPal

PayPal Customre Care

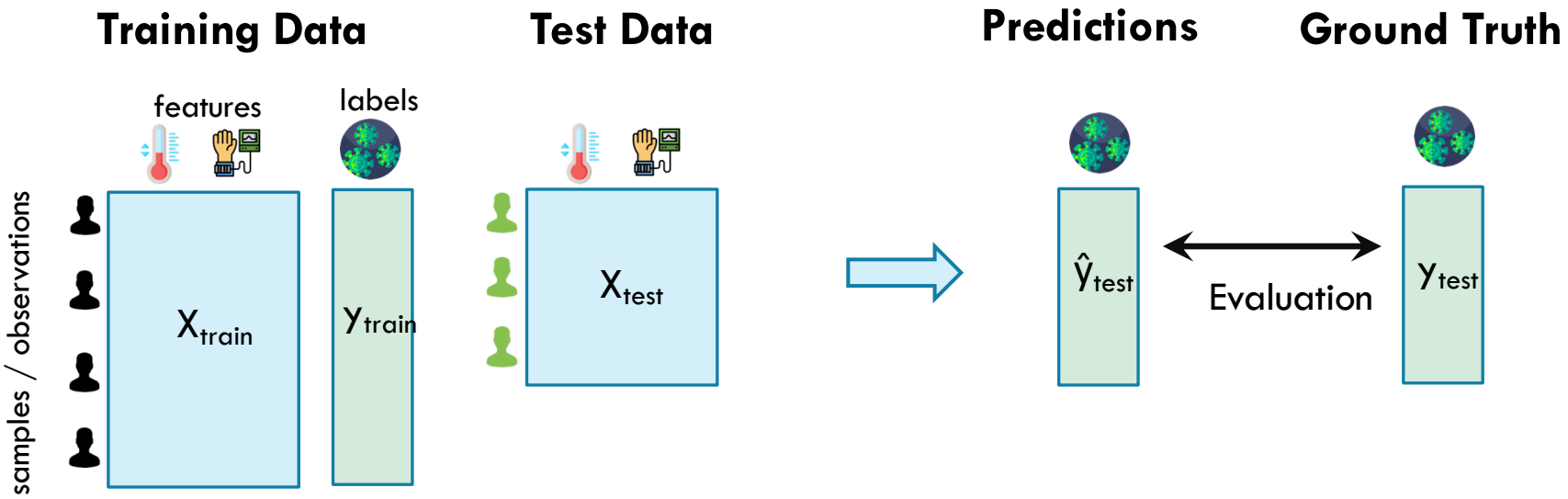
Hi,
Dear customer
At first Thank you for paying attention to PayPal Customer Care.
We contact you for confirming you Paypal account.because of security reason you have to confirm you account in PayPal again.our log on your account show us some illegal usage then we want you to pay some time and Confirm your account again. for confirming just login to PayPal with attached form just from

Spam Detection



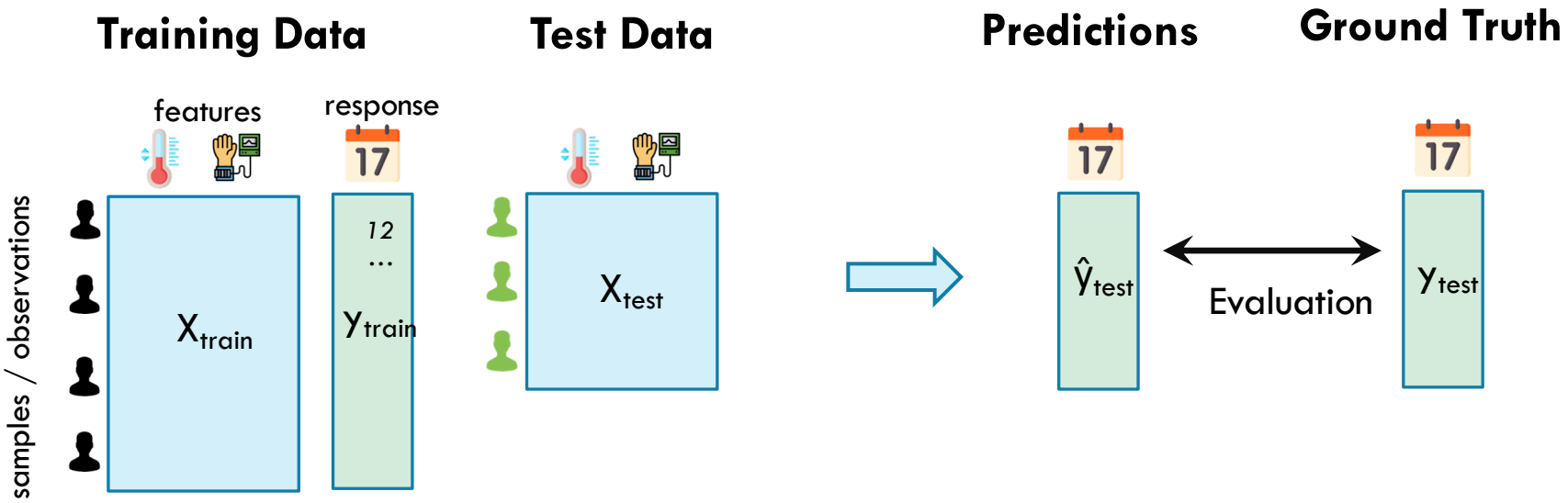
Network Intrusion Detection

Classification: Problem Setup



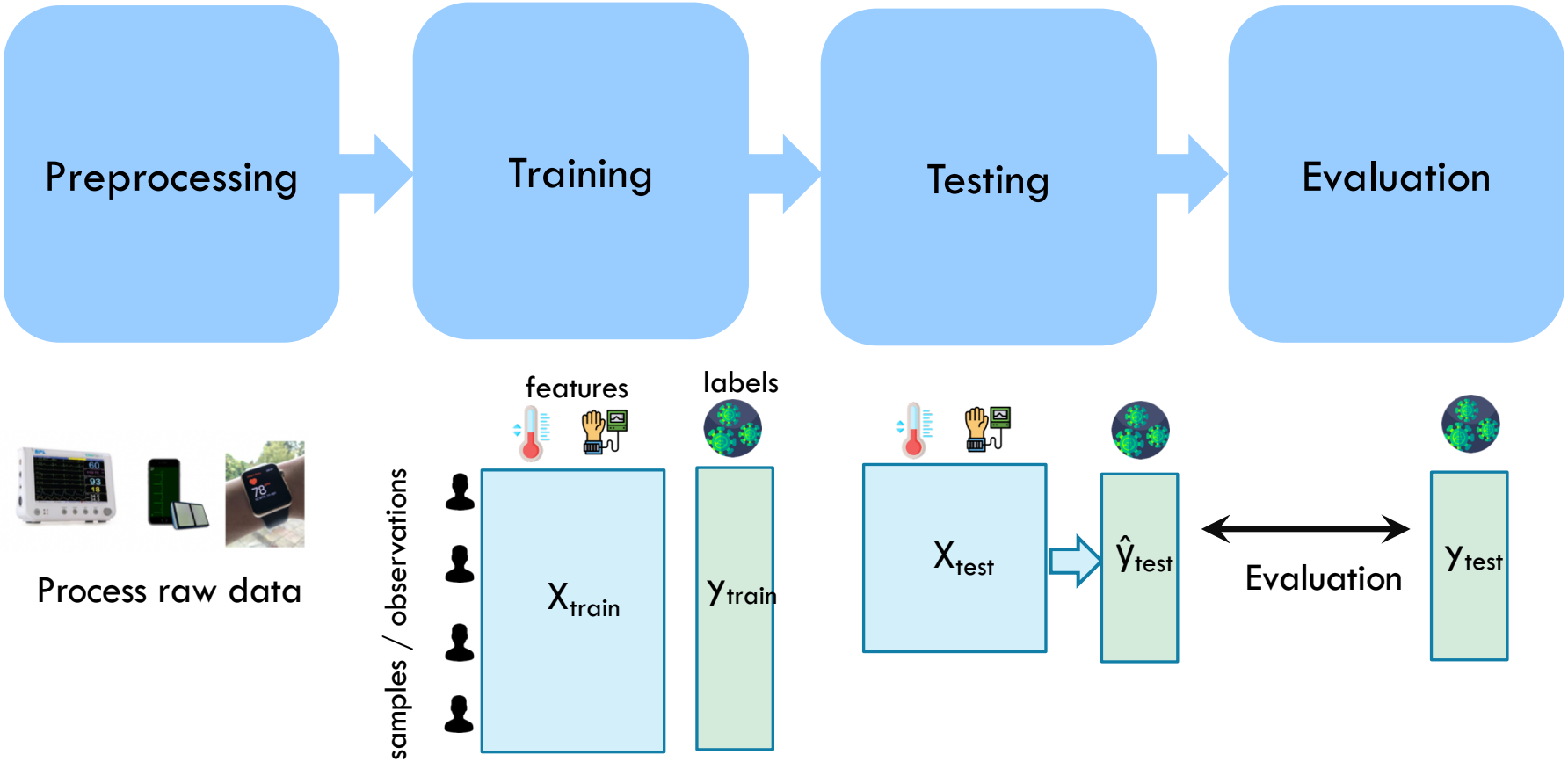
Classification: Categorize samples into *classes*, given training data

Regression: Problem Setup

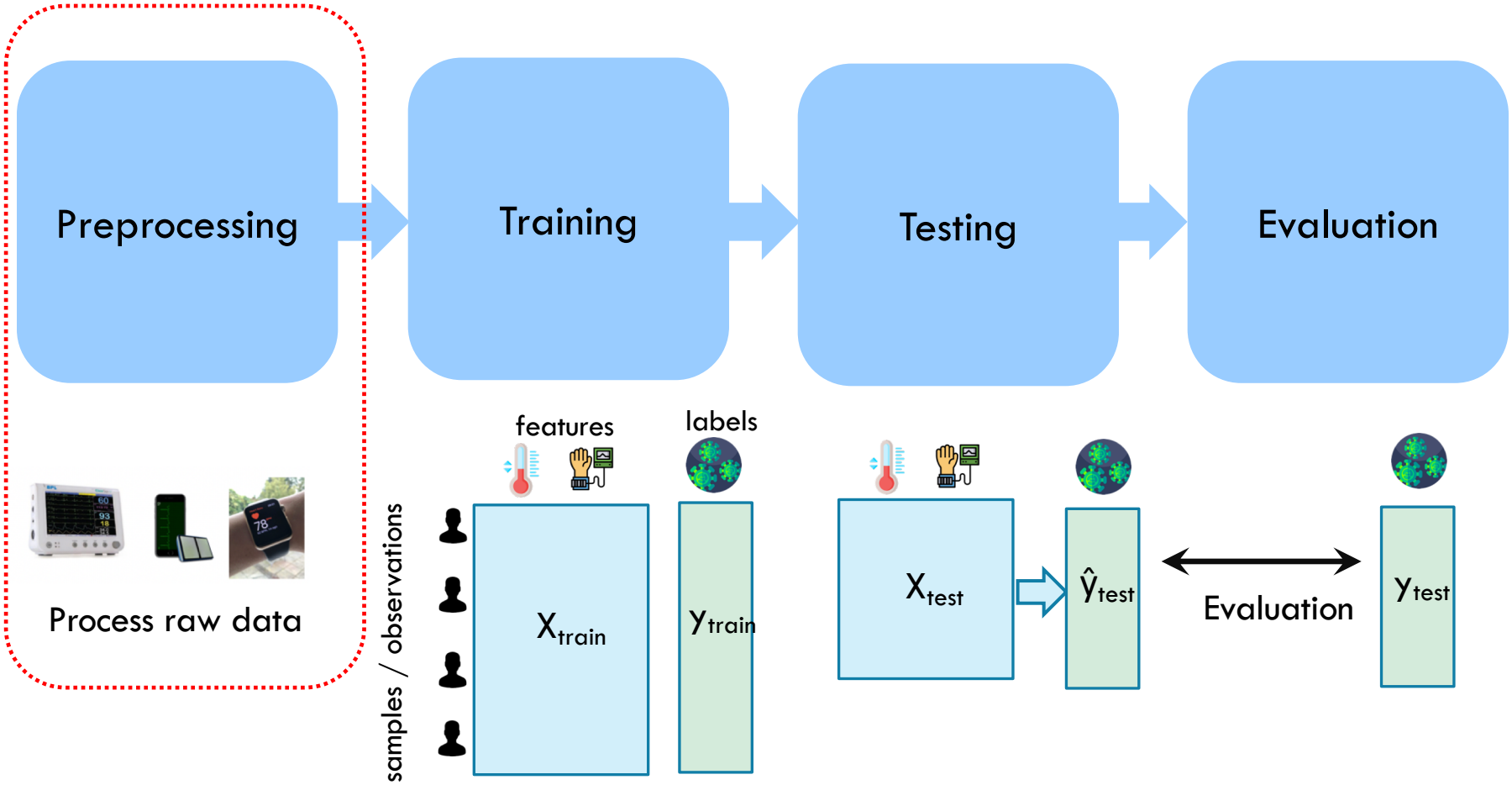


Regression: predict *numeric* labels, given training data

Typical Machine Learning Pipeline



Typical Machine Learning Pipeline



Data Quality

The most important point is that poor data quality is an unfolding disaster.

Poor data quality costs the typical company at least 10% of revenue; 20% is probably a better estimate.

Thomas C. Redman, DM Review, August 2004

Forbes

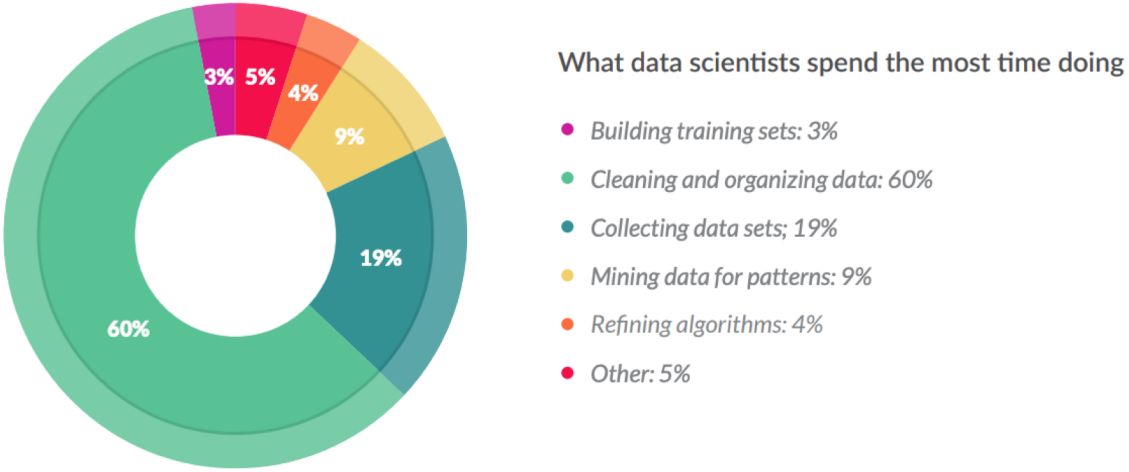
Jun 16, 2021, 05:04pm EDT | 29,167 views

Andrew Ng Launches A Campaign For Data-Centric AI

Data is eating the world so Andrew Ng wants to make sure we radically improve its quality. "Data is food for AI," says Ng, and he is launching a campaign to shift the focus of AI practitioners from model/algorithm development to the quality of the data they use to train the models.

Data Preprocessing

- Often an under-valued part of data science, but very important
  - "Garbage in, garbage out"
  - Google AI (*SIGCHI* 2021): 92% of the analysed high-stakes AI projects practitioners report negative downstream effects from data issues
  - Anecdotally on Kaggle (prediction contest site), feature engineering is often one of the key important factors for winning contests
- NY Times: data scientists spend 50-80% of their time on data preparation



https://www.nytimes.com/2014/08/18/technology/for-big-data-scientists-hurdle-to-insights-is-janitor-work.html
https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf

Data Quality: Missing Values

Why is data missing?

- Information was not collected: e.g. people decline to give weight
- *Missing at random*: missing values are randomly distributed. If data is instead *missing not at random*: then the missingness itself may be important information.

How to handle missing values?

- Eliminate objects (rows) with missing values
- Or: fill in the missing values ("imputation")
  - E.g. based on the **mean** / **median** of that attribute
  - Or: by fitting a **regression** model to predict that attribute given other attributes

UserID	Height (m)	Country	...
1	1.61	SG	...
2	1.50	US	...
3	***NA***	MY	...
...	...	...	...

Median / Regression Imputation

UserID	Height (m)	Country	...
1	1.61	SG	...
2	1.50	US	...
3	***1.55***	MY	...
...	...	...	...

Data Quality: Missing Values

Why is data missing?

- Information was not collected: e.g. people decline to give weight
- *Missing at random*: missing values are randomly distributed. If data is instead *missing not at random*: then the missingness itself may be important information.

How to handle missing values?

- Eliminate objects (rows) with missing values
- Or: fill in the missing values ("imputation")
  - E.g. based on the **mean** / **median** of that attribute
  - Or: by fitting a **regression** model to predict that attribute given other attributes
  - **Dummy variables**: optionally insert a column which is 1 if the variable was missing, and 0 otherwise

UserID	Height (m)	Country	...
1	1.61	SG	...
2	1.50	US	...
3	***NA***	MY	...
...	...	...	...

Dummy Variable Imputation

UserID	Height (m)	Missing?	Country	...
1	1.61	0	SG	...
2	1.50	0	US	...
3	***1.55***	1	MY	...
...	...		...	...

Data Quality: Missing Values

Why is data missing?

- Information was not collected: e.g. people decline to give weight
- *Missing at random*: missing values are randomly distributed. If data is instead *missing not at random*: then the missingness itself may be important information.

How to handle missing values?

- Eliminate objects (rows) with missing values
- Or: fill in the missing values ("imputation")
  - E.g. based on the **mean** / **median** of that attribute
  - Or: by fitting a **regression** model to predict that attribute given other attributes
  - **Dummy variables**: optionally insert a column which is 1 if the variable was missing, and 0 otherwise

UserID	Height (m)	Country	...
1	1.61	SG	...
2	1.50	US	...
3	***NA***	MY	...
...	...	...	...

Dummy Variable Imputation

UserID	Height (m)	Missing?	Country	...
1	1.61	0	SG	...
2	1.50	0	US	...
3	***1.55***	1	MY	...
...	...		...	...

In PySpark:

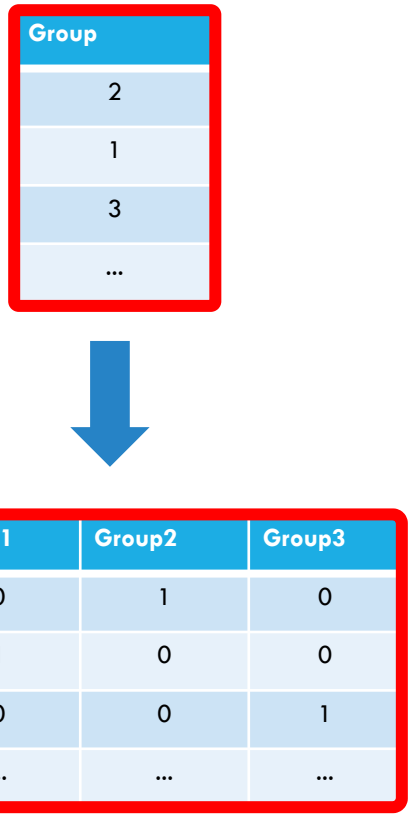
```
imputer = Imputer(inputCols=["a", "b"],
                  outputCols=["out_a", "out_b"])
model = imputer.fit(df)
model.transform(df).show()
```

One Hot Encoding

Convert discrete feature to a series of binary features.

E.g. the first record has group 2, so we set its 2nd binary feature to 1, and all the rest to 0.

This lets us apply algorithms which can handle binary features (e.g. linear regression)

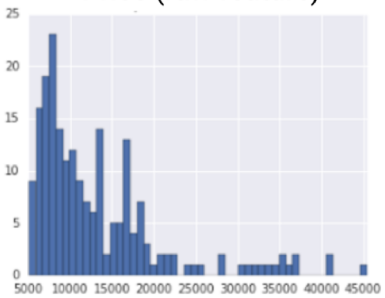


Group
2
1
3
...

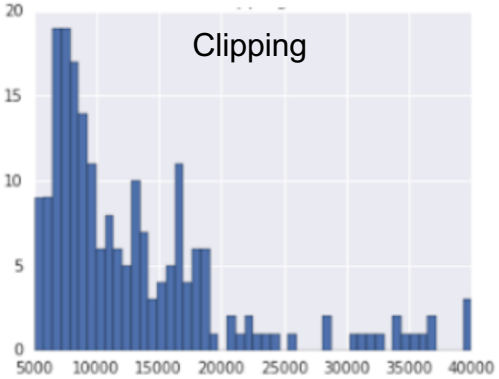
Group1	Group2	Group3
0	1	0
1	0	0
0	0	1
...	...	...

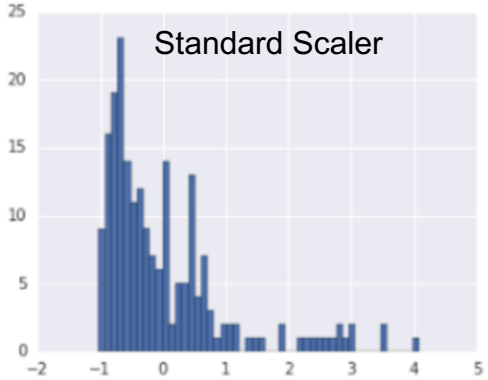
Normalization

Price (raw feature)



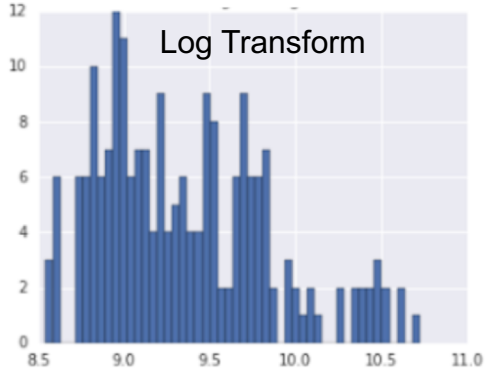
$\text{clip}(x)$

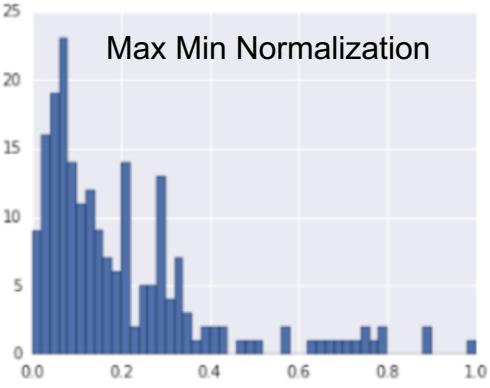




$$\frac{x - mean(x)}{std(x)}$$

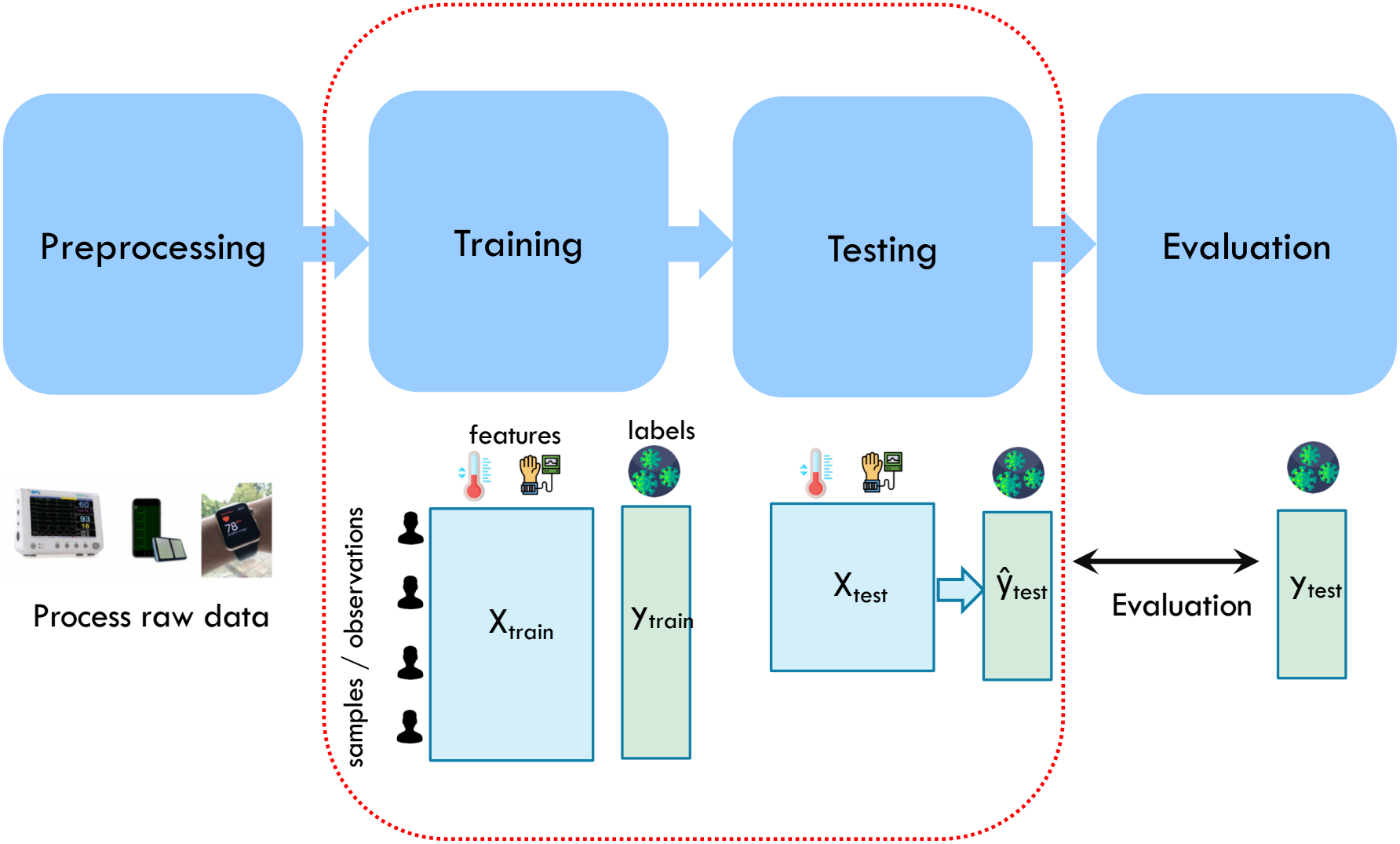
$\log(1 + x)$





$$\frac{x - \min(x)}{\max(x) - \min(x)}$$

Typical Machine Learning Pipeline

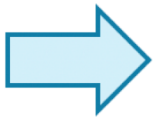


Running Example: Spam Classification

Features

Number of '$'	Number of '!'
5	2

$$x = \begin{pmatrix} 5 \\ 2 \end{pmatrix}$$



(Predicted probability of the email being spam)

?

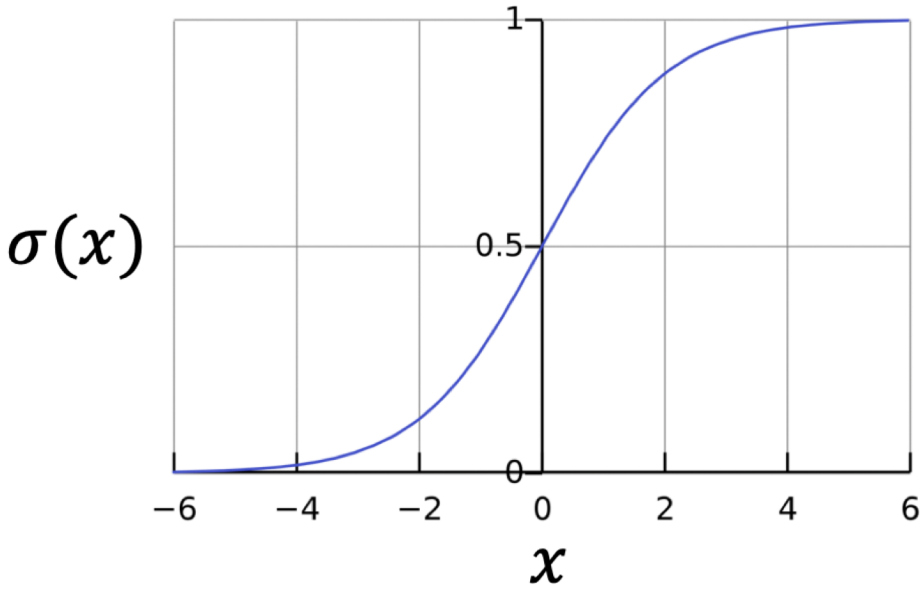
$\hat{y}$

Logistic Regression: Sigmoid Function

The sigmoid function $\sigma(x)$ maps the real numbers to the range $(0, 1)$

It is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Logistic Regression: Sigmoid Function

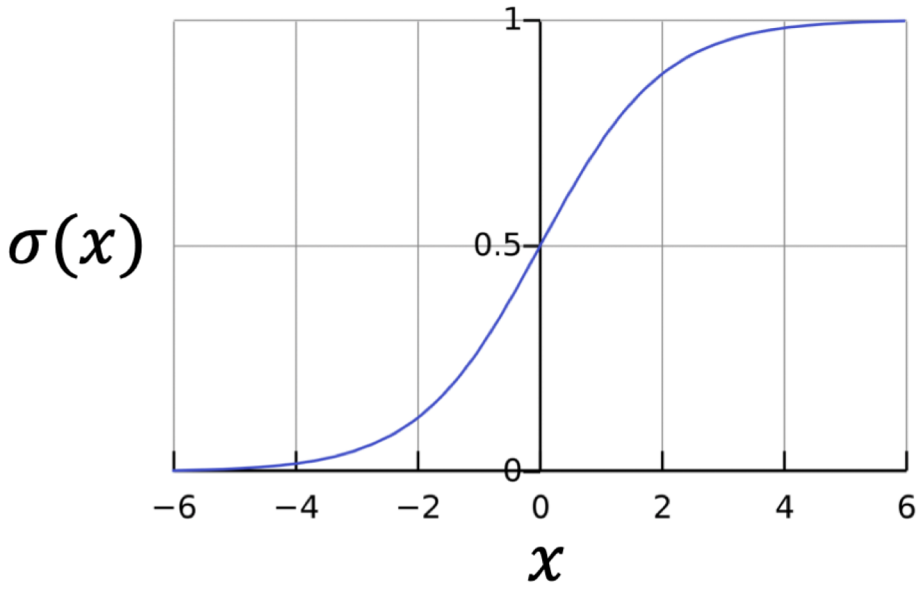
The sigmoid function $\sigma(x)$ maps the real numbers to the range $(0, 1)$

It is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Check:

As $x \to -\infty, \sigma(x) \to \frac{1}{1+e^{\infty}} = 0$



Logistic Regression: Sigmoid Function

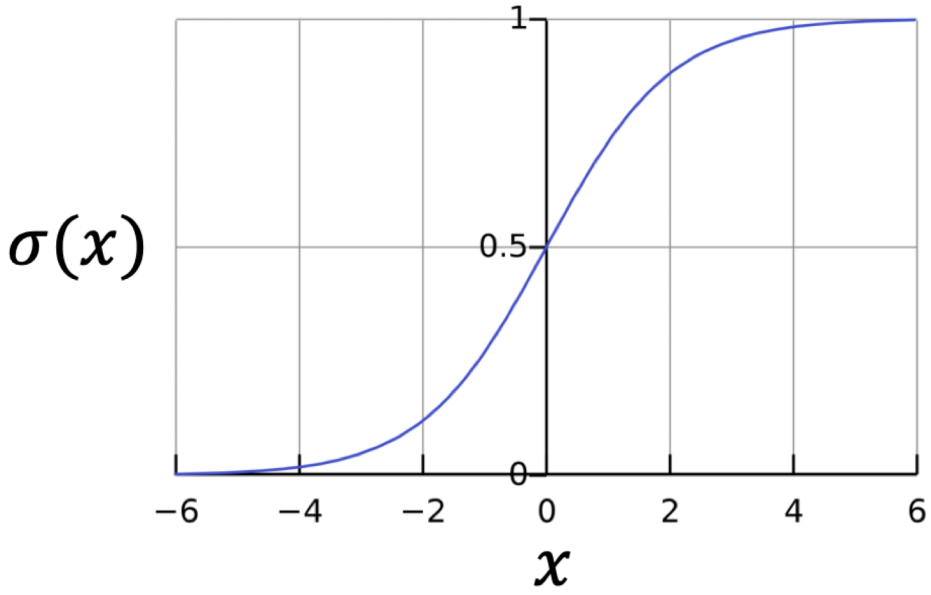
The sigmoid function $\sigma(x)$ maps the real numbers to the range $(0, 1)$

It is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Check:

$$\sigma(0) = \frac{1}{1+e^{0}} = 0.5$$



Logistic Regression: Sigmoid Function

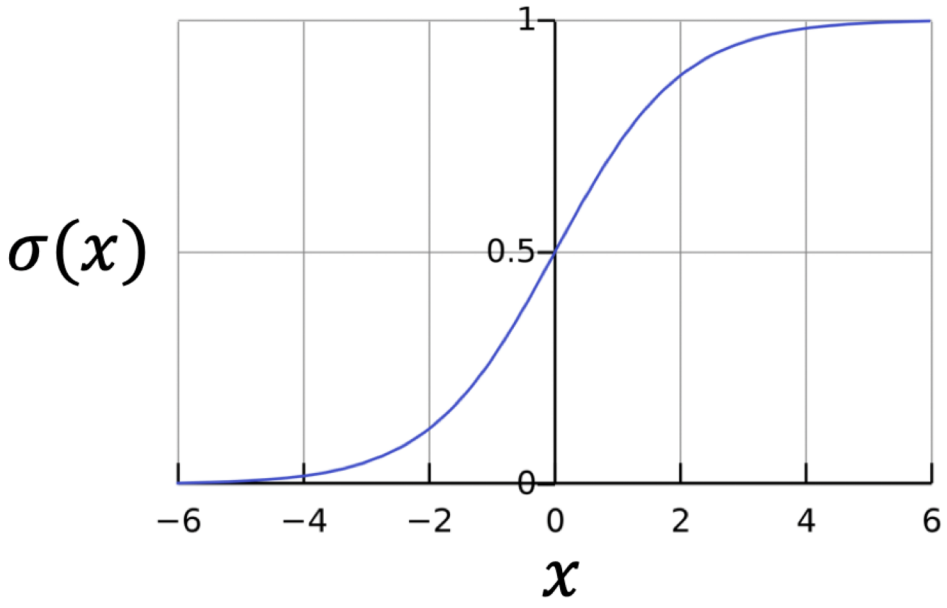
The sigmoid function $\sigma(x)$ maps the real numbers to the range $(0, 1)$

It is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Check:

As $x \to \infty, \sigma(x) \to \frac{1}{1+e^{-\infty}} = 1$



Logistic Regression: Parameters

Features

Number of '$'	Number of '!'
5	2

$x = \begin{pmatrix} 5 \\ 2 \end{pmatrix}$

(Predicted probability of the email being spam)

? $\hat{y}$

Parameters:

Weights $w = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

1	2

Bias $b = -5$

-5

Logistic Regression: Prediction

Features

Number of '$'	Number of '!'
5	2

$x = \begin{pmatrix} 5 \\ 2 \end{pmatrix}$

(Predicted probability of the email being spam)

? $\hat{y}$

Parameters:

Weights $w = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ Bias $b = -5$

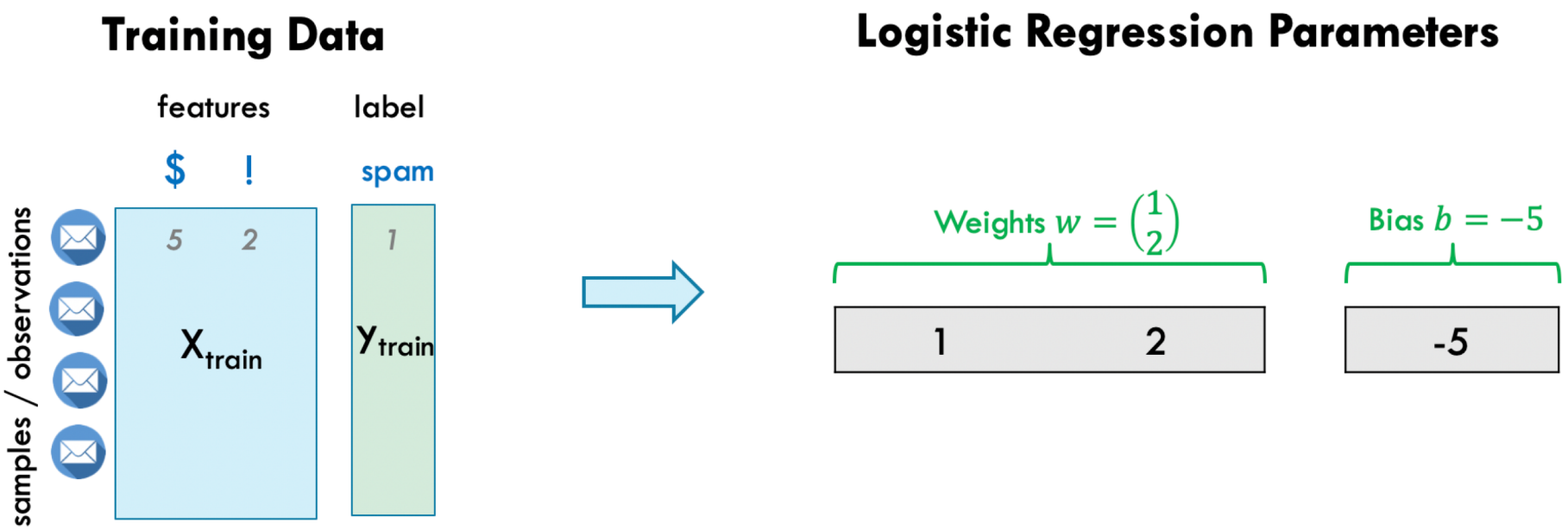
1	2	-5

Prediction:

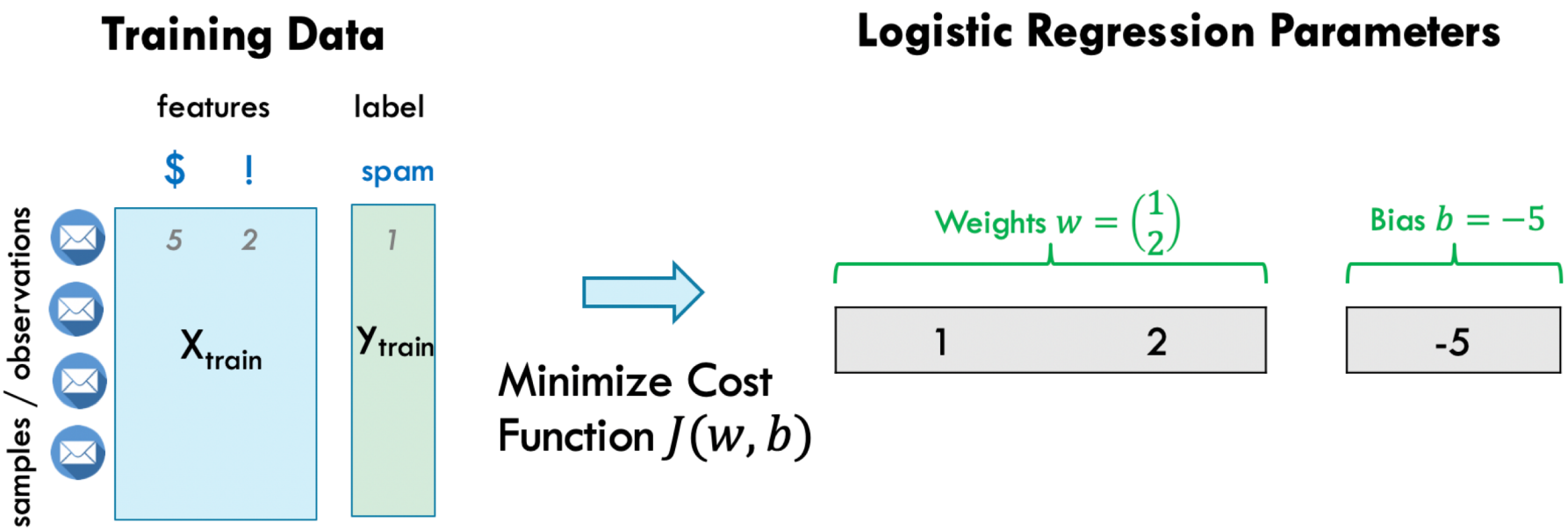
Sigmoid function, Dot product

$$\hat{y} = \sigma(x \cdot w + b) = \sigma\left(\begin{pmatrix} 5 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} - 5\right) = \sigma(9 - 5) = \frac{1}{1 + e^{-4}} = 0.982$$

Training Logistic Regression



Training Logistic Regression

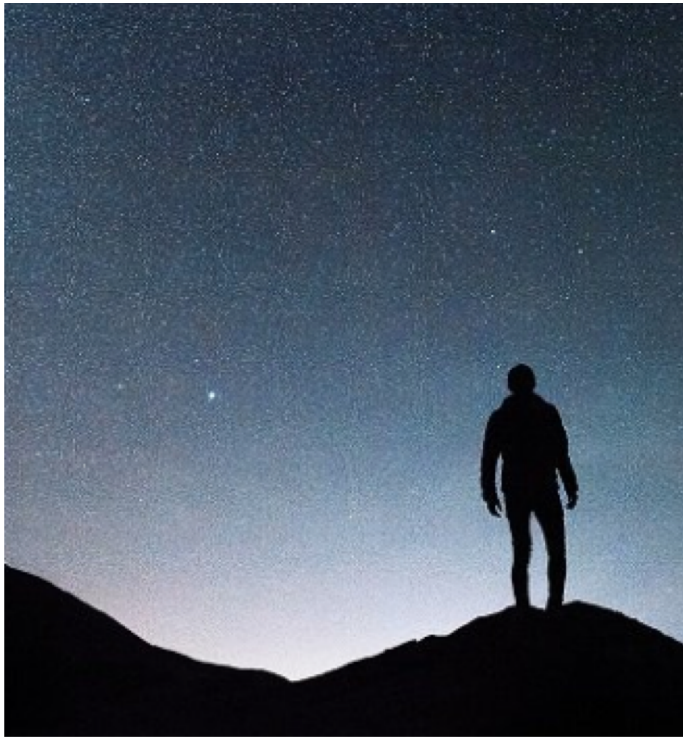


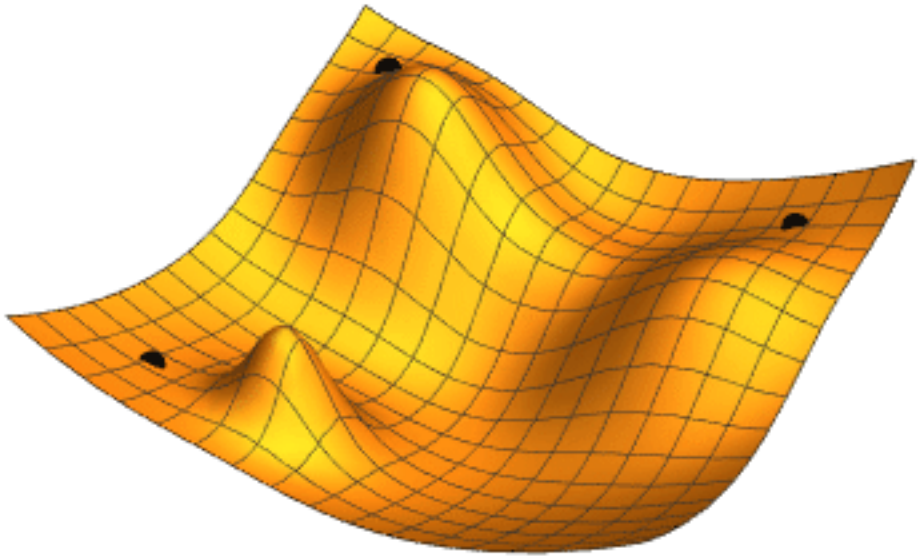
Big Picture: machine learning involves fitting the ***parameters*** of a model (here *w*, *b*). This is often done by minimizing a ***loss function***.

Here, the cost function J is ***Cross Entropy Loss*** (intuitively: think of the model's predictions as a probability. The higher the probability of the data given this model, the better our model is)

Minimizing cost function J using gradient descent

Minimizing cost function J using gradient descent

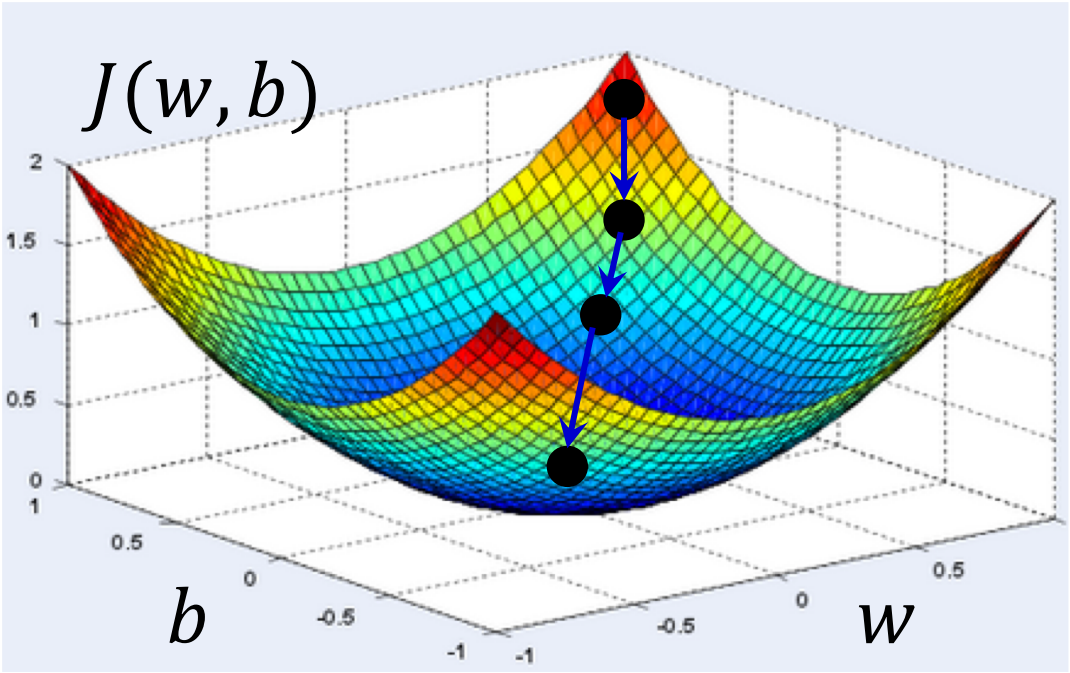




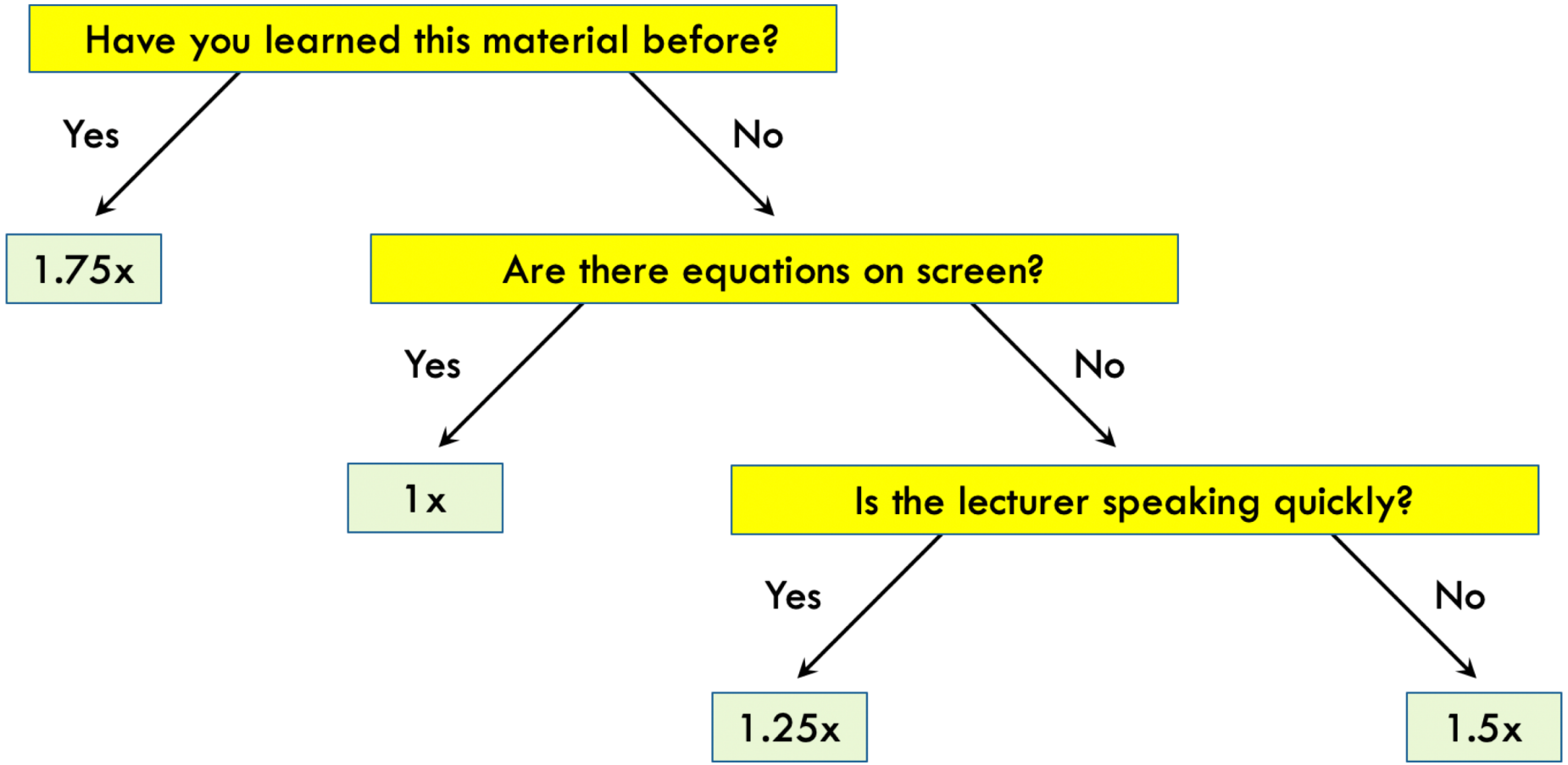
Minimizing cost function J using gradient descent

Optional

- We want to find w, b to minimize $J(w, b)$
- Start at an arbitrary point, then move following the steepest downward slope ('*gradient*')
- Continue until convergence
  - Stop when improvement in J is below a fixed threshold

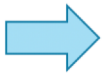


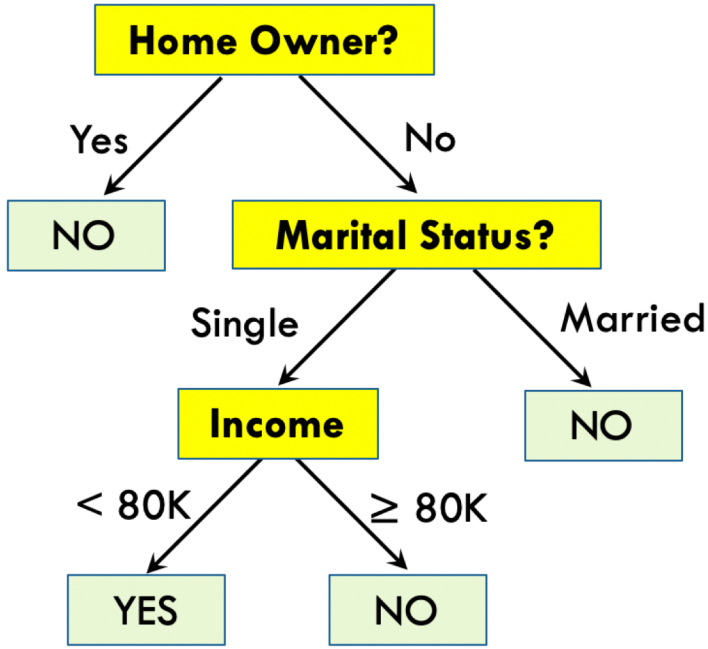
Other Classifiers: Decision Trees



What is a Decision Tree?

	categorical	categorical	continuous	response
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

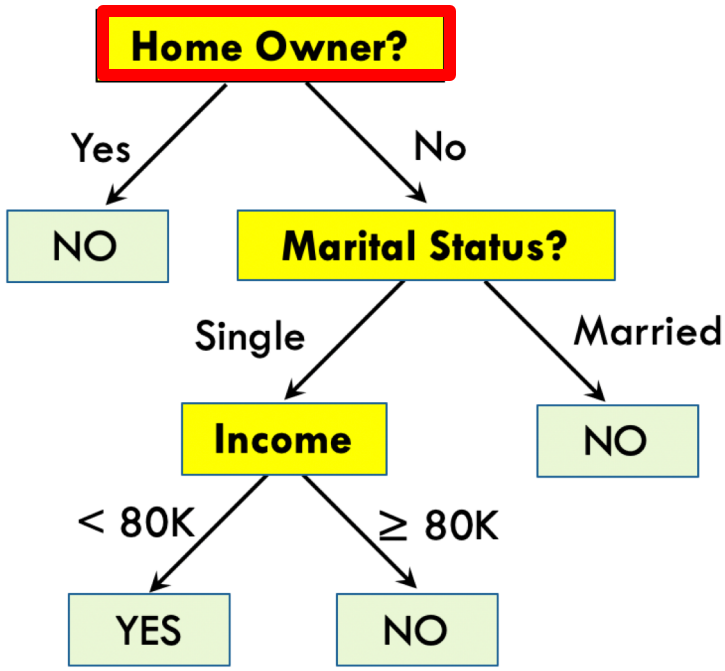




Applying a Decision Tree

ID	Home Owner (categorical)	Marital Status (categorical)	Annual Income (continuous)	Defaulted Borrower? (response)
1	No	Married	205K	?

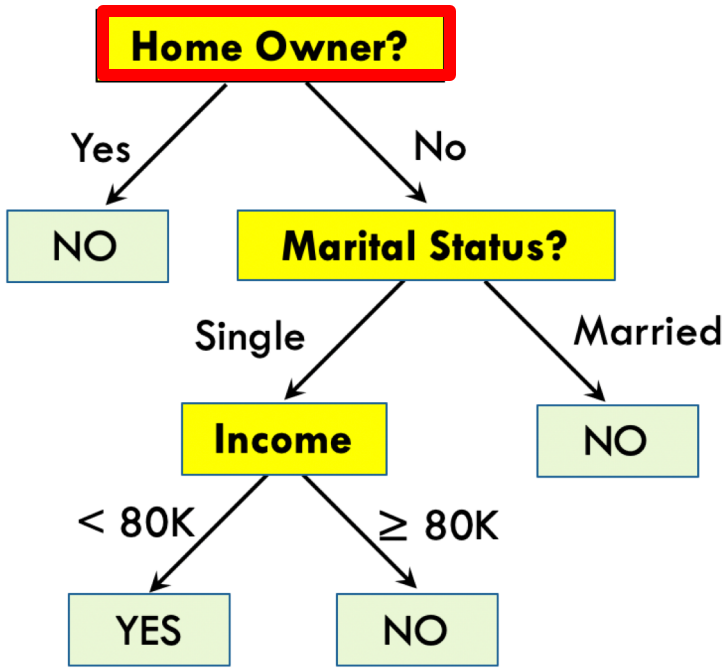
Test Record



Applying a Decision Tree

ID	Home Owner (categorical)	Marital Status (categorical)	Annual Income (continuous)	Defaulted Borrower? (response)
1	No	Married	205K	?

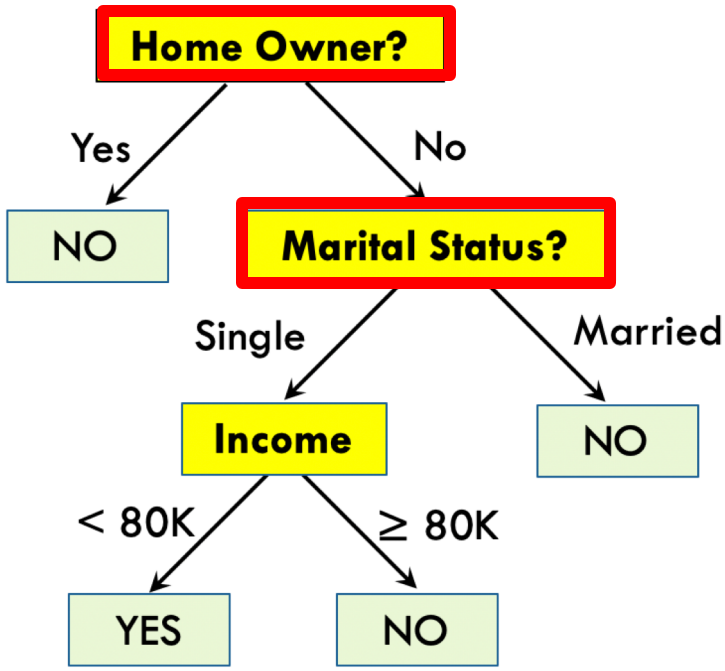
Test Record



Applying a Decision Tree

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower?
	categorical	categorical	continuous	response
1	No	Married	205K	?

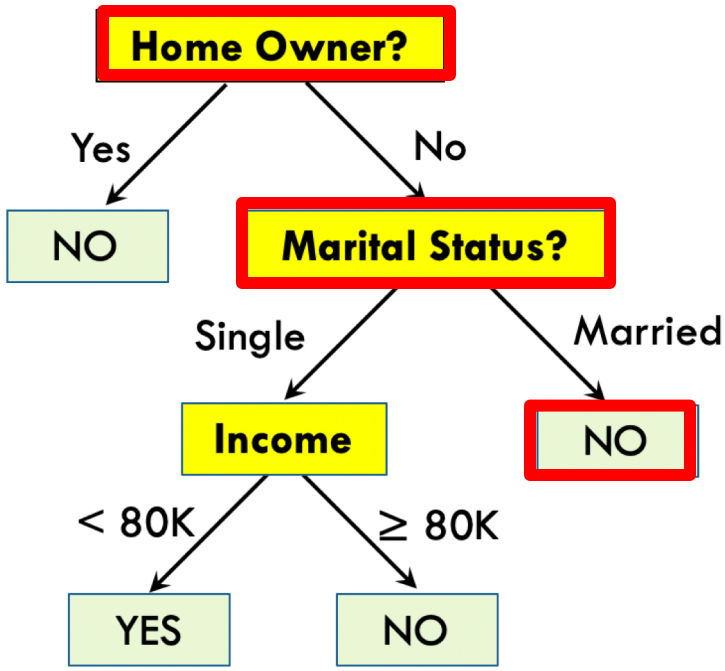
Test Record



Applying a Decision Tree

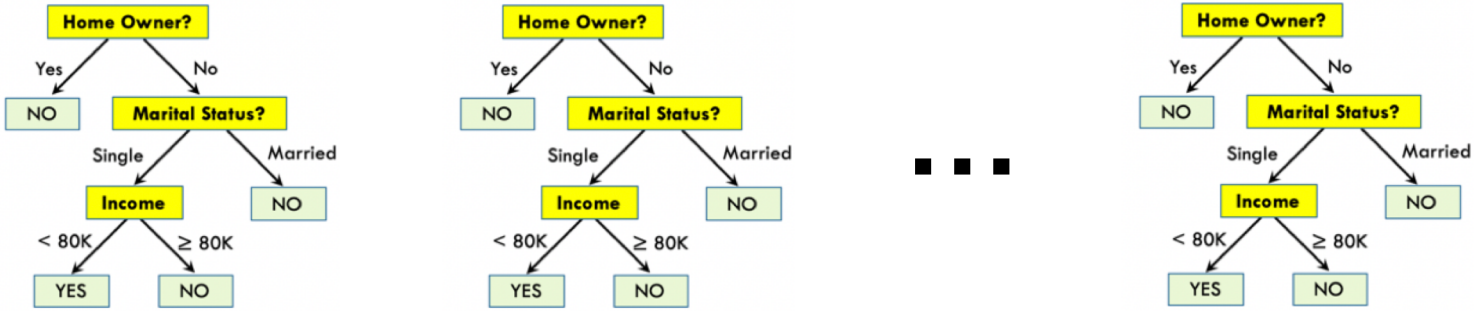
ID	Home Owner (categorical)	Marital Status (categorical)	Annual Income (continuous)	Defaulted Borrower? (response)
1	No	Married	205K	No

Test Record



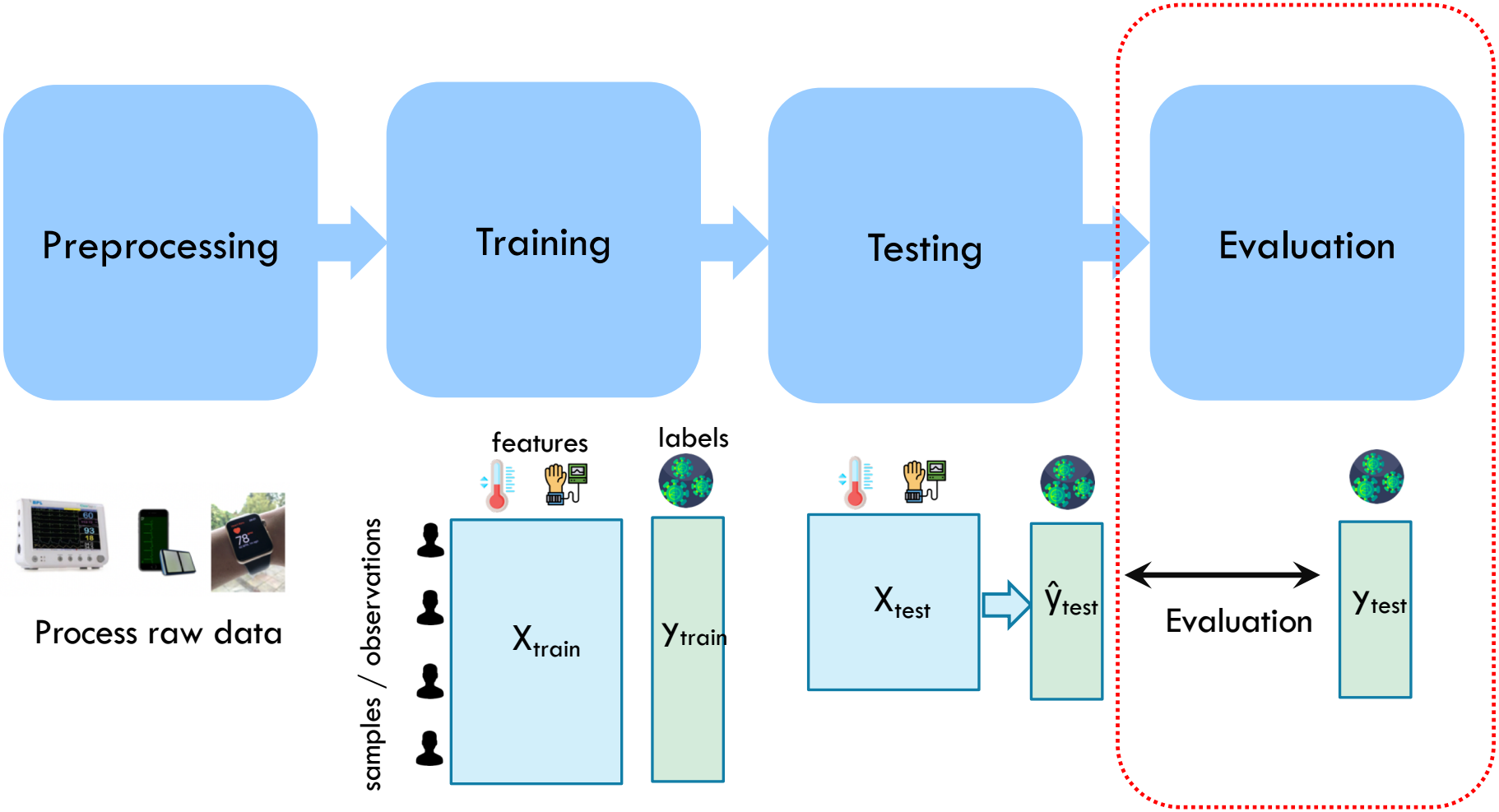
From Decision Trees to Random Forests and Gradient Boosted Trees

- Decision trees are simple, interpretable and fast, but suffer from poor accuracy, and are not robust to small changes
  - For applications where interpretability is especially important, learning "optimal decision trees" is still an active area of research
- Random Forests and Gradient Boosted Trees are very popular approaches which combine a large number of decision trees
- On tabular data, their accuracy is still highly competitive with neural networks, and are faster, easier to tune, and more interpretable



Model 1 Model 2 Model N

Typical Machine Learning Pipeline



THE STRAITS TIMES

Fast Covid-19 tests to be added to Singapore's testing arsenal: How do antigen rapid tests work?

The Health Ministry will be turning to Covid-19 antigen rapid tests to complement existing polymerase chain reaction tests – the gold standard for testing – as Singapore further opens its economy.

Clara Chong

PUBLISHED OCT 21, 2020, 5:00 AM SGT

Benefits

• Faster (less than 30 minutes) and hence more feasible for pre-event testing.

• Cheaper.

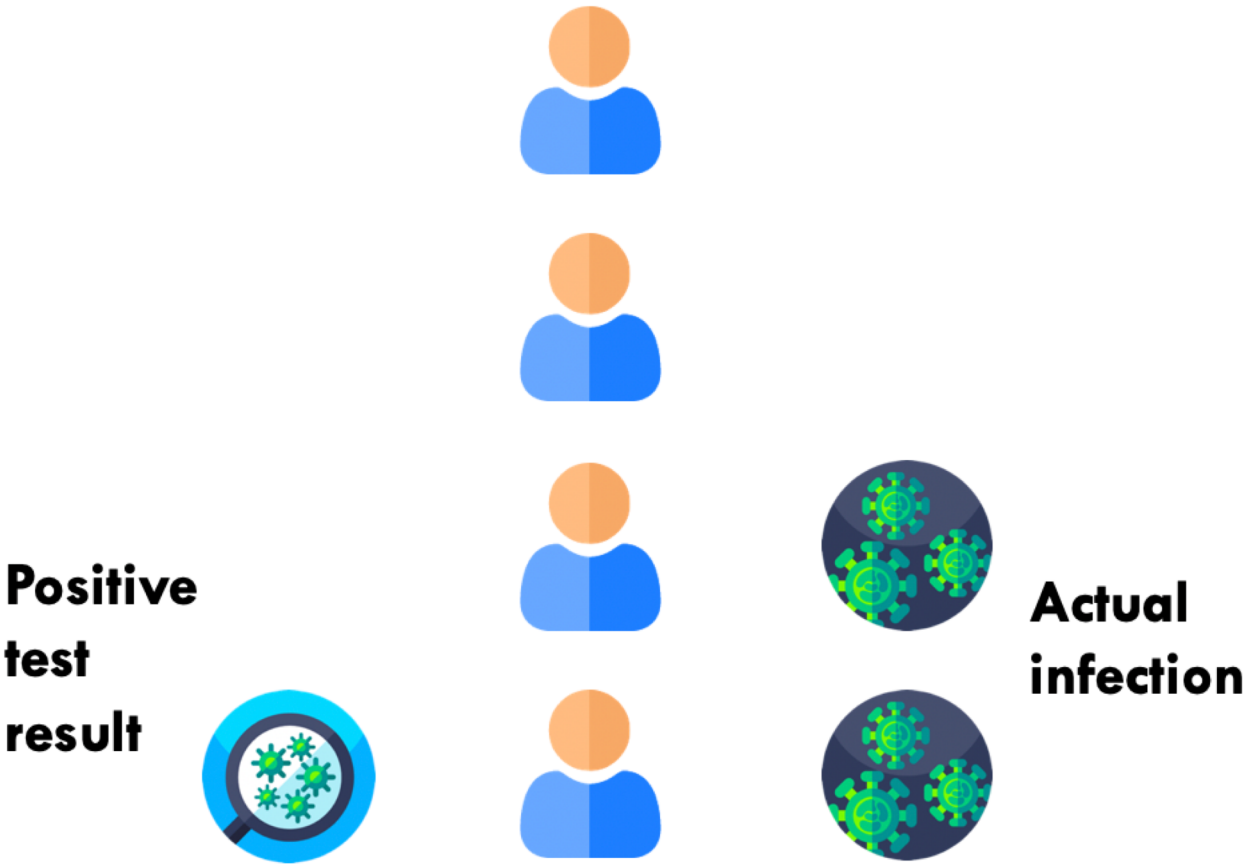
• Easier to administer.

Drawbacks

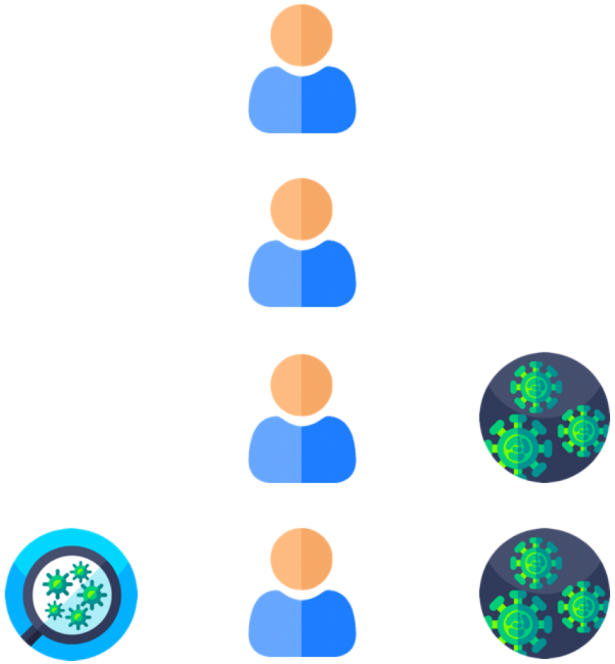
• Tests have lower sensitivity and specificity, and may carry a higher risk of false positives and false negatives.

• The World Health Organisation recommends at least 80 per cent sensitivity and 97 per cent specificity. This means that at least 80 per cent of those infected are identified, and at most 3 per cent of healthy subjects are tested false positives.

Binary Classification Setting



Binary Classification Setting



Predicted Label ($\hat{y}$)	Ground Truth Label (y)
0	0
0	0
0	1
1	1

Confusion Matrix

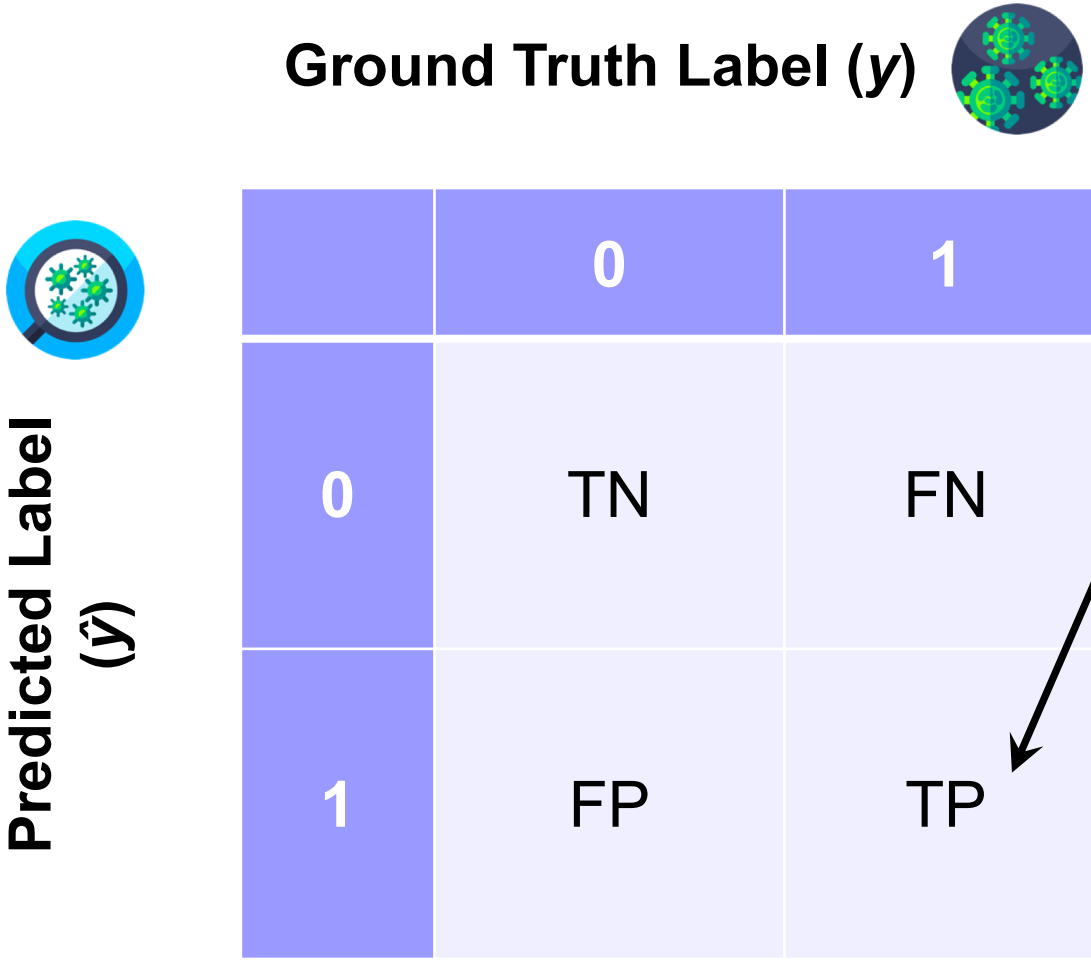
Predicted Label ($\hat{y}$)	Ground Truth Label (y)
0	0
0	0
0	1
1	1

Ground Truth Label (y)

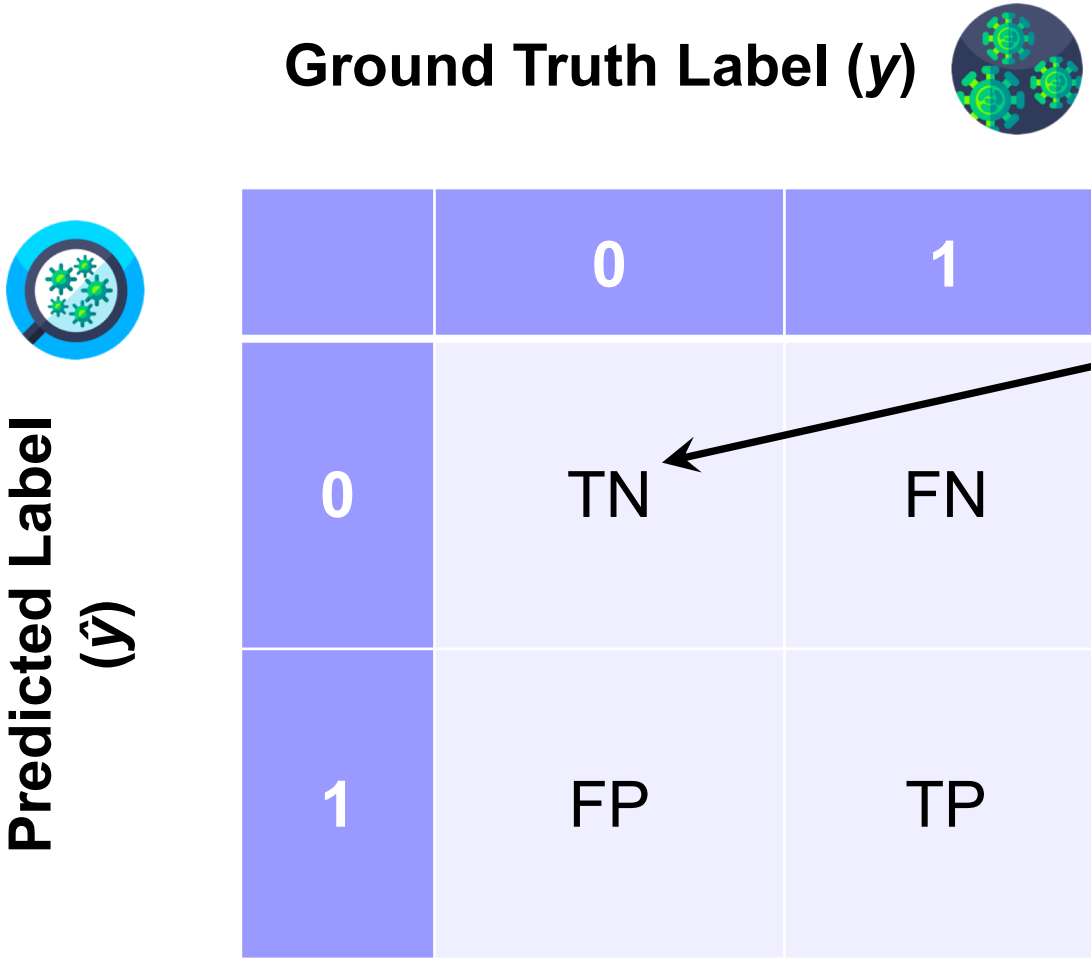
Predicted Label ($\hat{y}$)

	0	1
0	2	1
1	0	1

True/False Positives/Negatives



True/False Positives/Negatives



True/False Positives/Negatives

Predicted Label ($\hat{y}$)

Ground Truth Label (y)

	0	1
0	TN	FN
1	FP	TP

True Positive: test correctly output positive
True Negative: test correctly output negative
False Positive: test wrongly outputs positive

True/False Positives/Negatives

Predicted Label ($\hat{y}$)

Ground Truth Label (y) 

	0	1
0	TN	FN
1	FP	TP

True Positive: test correctly output positive
True Negative: test correctly output negative
False Positive: test wrongly outputs positive
False Negative: test wrongly outputs negative

Accuracy

Ground Truth Label (*y*)

	0	1
0	TN	FN
1	FP	TP

Accuracy: fraction of correct predictions

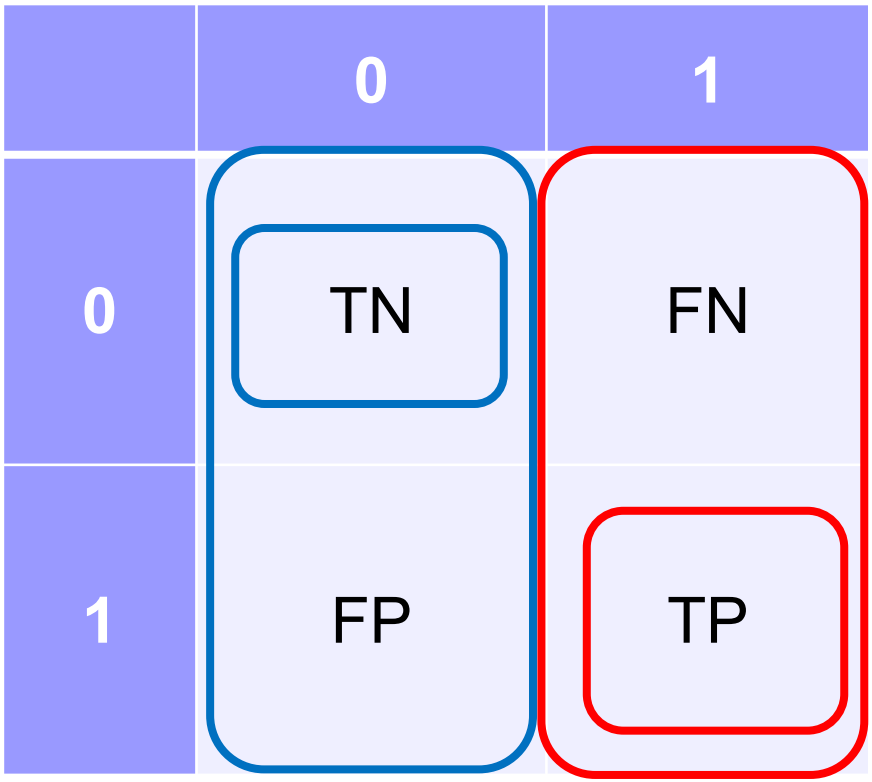
Sensitivity / Specificity

Ground Truth Label (*y*)

	0	1
0	TN	FN
1	FP	TP

Sensitivity: fraction of positive cases that are detected

Sensitivity / Specificity



Sensitivity: fraction of positive cases that are detected

Specificity: fraction of actual negatives that are correctly identified

Sensitivity / Specificity

Ground Truth Label (*y*)

	0	1
0	TN	FN
1	FP	TP

Sensitivity: fraction of positive cases that are detected

Specificity: fraction of actual negatives that are correctly identified

Q: Assume specificity = 97%; if I tested negative, does it mean my probability of being negative is 97%?

Sensitivity / Specificity

Ground Truth Label (*y*)

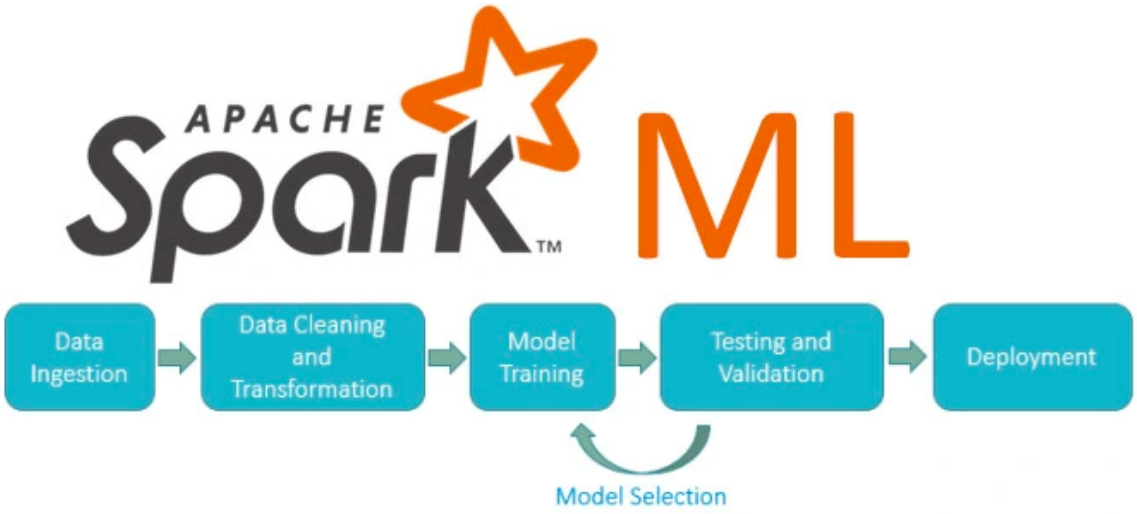
	0	1
0	TN	FN
1	FP	TP

Sensitivity: fraction of positive cases that are detected
Specificity: fraction of actual negatives that are correctly identified

Q: Assume specificity = 97%; if I tested negative, does it mean my probability of being negative is 97%?
A: no (that is the wrong direction).

Today's Plan

- **Supervised Machine Learning Basics**
  - Introduction
  - Preprocessing
  - Model Training and Testing
  - Evaluation
- **Spark MLLib**



Spark MLLib: Simple Logistic Regression Model

```
from pyspark.ml.classification import LogisticRegression

training = spark.read.format("libsvm").load("data/mllib/sample_libsvm_data.txt")

lr = LogisticRegression(maxIter=10)

lrModel = lr.fit(training)

print("Coefficients: " + str(lrModel.coefficients))
print("Intercept: " + str(lrModel.intercept))
```

Pipelines

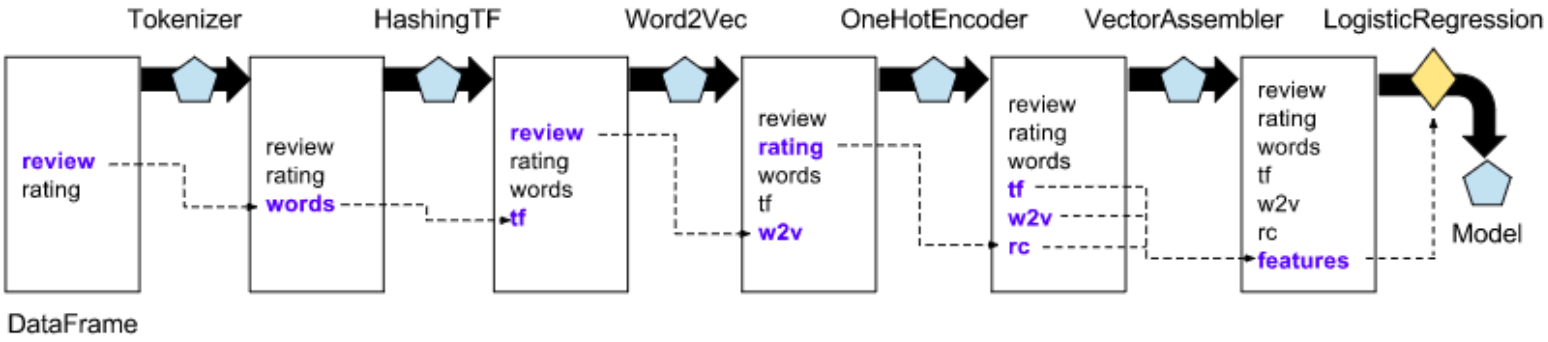
Idea: building complex pipeline out of simple building blocks
(Note: scikit-learn pipelines are basically the same as Spark MLLib ones)

Pipelines

Idea: building complex pipeline out of simple building blocks: e.g. normalization, feature transformation, model fitting.

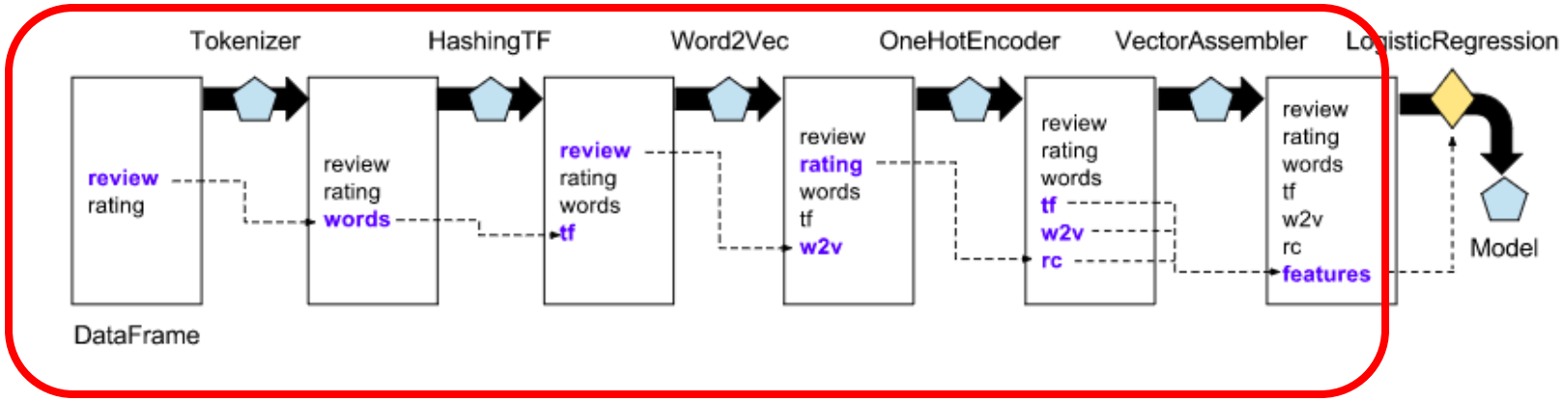
Why?

- Better code reuse: without pipelines, we would repeat a lot of code, e.g. between the training and test pipelines, cross-validation, model variants, etc.
- Easier to perform cross validation, and hyperparameter tuning.



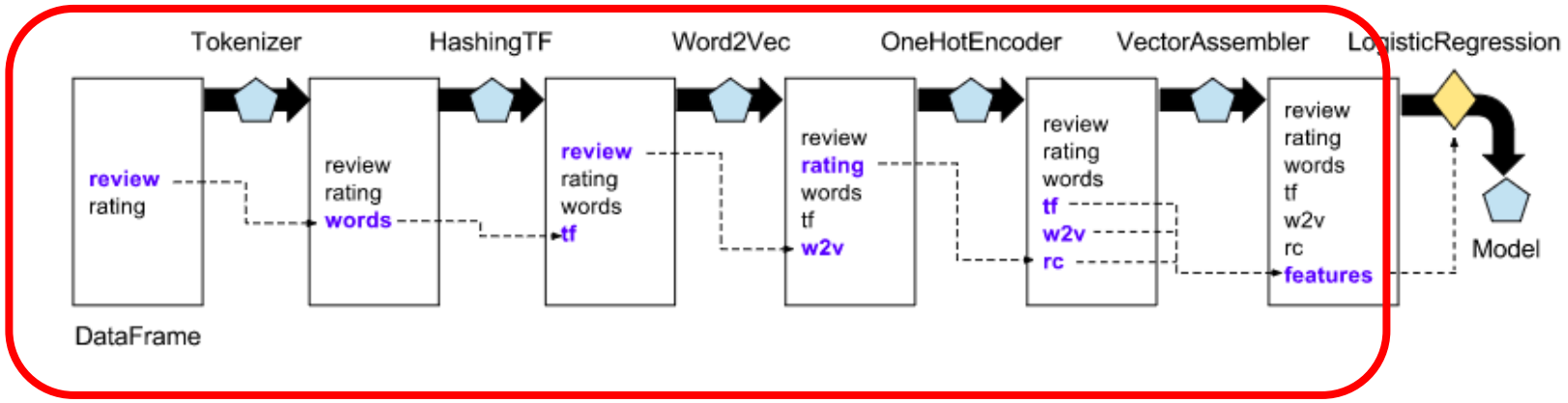
Building Blocks: Transformers

- **Transformers** map DataFrames to DataFrames
- Example: one-hot encoding, tokenization
  - Generally, these transformers output a new DataFrame which ***append*** their result to the original DataFrame.
  - Similarly, a fitted model (e.g. logistic regression) is a Transformer that transforms a DataFrame into one with the predictions appended.



Building Blocks: Transformers

- **Transformers** map DataFrames to DataFrames
- Example: one-hot encoding, tokenization
  - Generally, these transformers output a new DataFrame which ***append*** their result to the original DataFrame.
  - Similarly, a fitted model (e.g. logistic regression) is a Transformer that transforms a DataFrame into one with the predictions appended.
  - They have a transform() method, which performs their transformation.



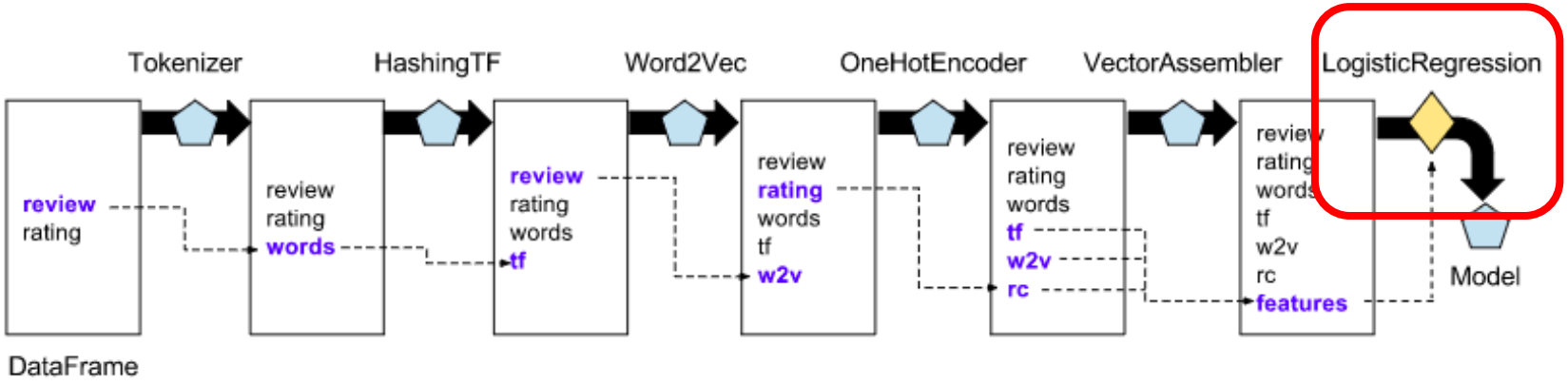
Building Blocks: Estimator

- **Estimator** is an algorithm which takes in data, and outputs a model. For example, a learning algorithm (the LogisticRegression object) can be fit to data, producing the trained logistic regression model.
- They have a fit() method, which returns a Transformer.

```
from pyspark.ml.classification import LogisticRegression

training = spark.read.format("libsvm").load("data/mllib/sample_libsvm_data.txt")

lr = LogisticRegression(maxIter=10)

lrModel = lr.fit(training)

print("Coefficients: " + str(lrModel.coefficients))
print("Intercept: " + str(lrModel.intercept))
```

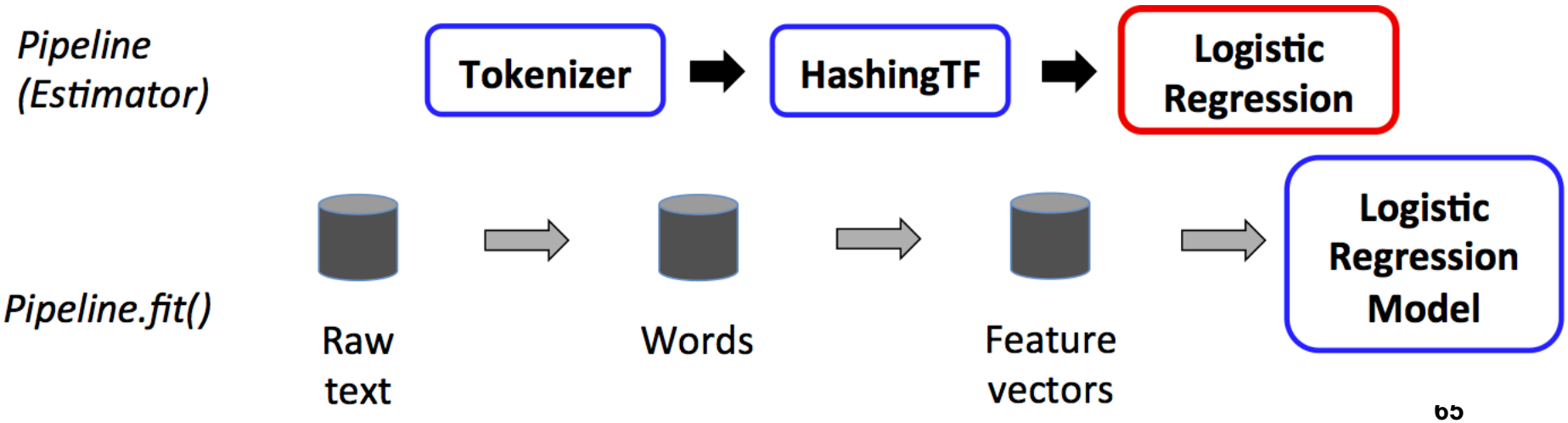
Building Blocks: Estimator

- **Estimator** is an algorithm which takes in data, and outputs a model. For example, a learning algorithm (the LogisticRegression object) can be fit to data, producing the trained logistic regression model.
- They have a fit() method, which returns a Transformer.



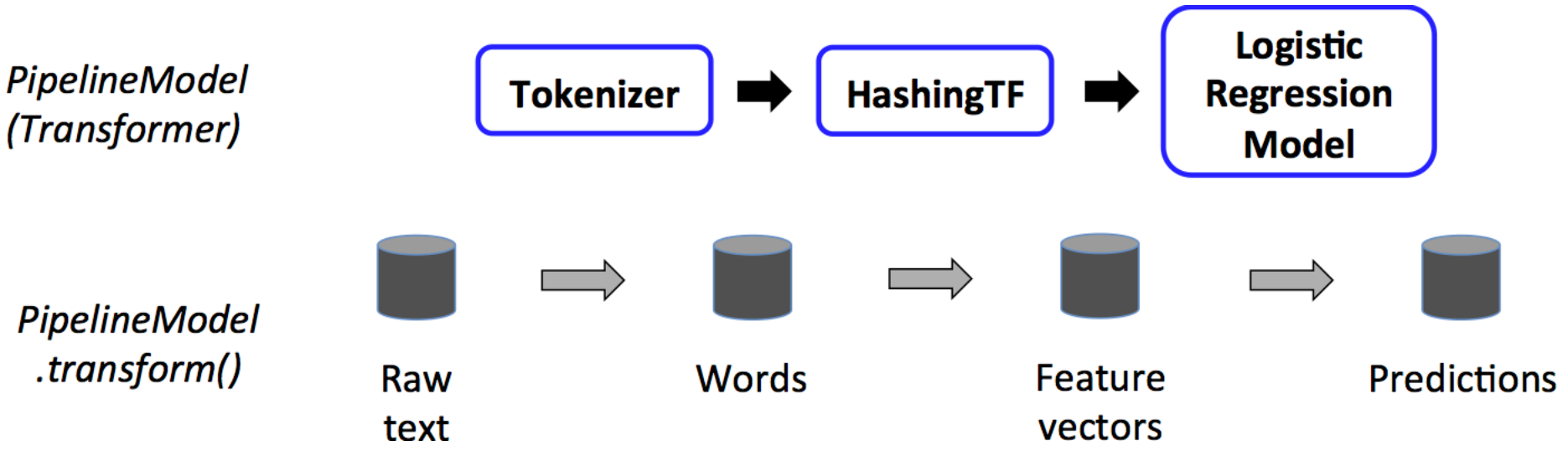
Pipeline: Training Time

- A pipeline chains together multiple Transformers and Estimators to form an ML workflow.
- It is an Estimator. When Pipeline.fit() is called:
  - Starting from the beginning of the pipeline:
  - For Transformers, it calls transform()
  - For Estimators, it calls fit() to fit the data, then transform() on the fitted model



Pipeline: Test Time

- The output of Pipeline.fit() is the estimated pipeline model (of type PipelineModel).
  - It is a transformer, and consists of a series of Transformers.
  - When its transform() is called, each stage's transform() method is called.



Example: Pipeline

```
training = spark.createDataFrame([
    (0, "a b c d e spark", 1.0),
    (1, "b d", 0.0),
    (2, "spark f g h", 1.0),
    (3, "hadoop mapreduce", 0.0)
], ["id", "text", "label"])

# Configure an ML pipeline, which consists of three stages:
tokenizer, hashingTF, and lr.
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(),
outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])

# Fit the pipeline to training documents.
model = pipeline.fit(training)
```

Acknowledgements

- https://www.kdnuggets.com/2020/04/data-transformation-standardization-normalization.html
- https://blog.insightdatascience.com/spark-pipelines-elegant-yet-powerful-7be93afcdd42
- https://spark.apache.org/docs/latest/ml-pipeline.html
- https://www.forbes.com/sites/gilpress/2021/06/16/andrew-ng-launches-a-campaign-for-data-centric-ai/#:~:text=Landing%20AI%2C%20the%20startup%20Ng,the%20quality%20of%20the%20data.
- https://www.straitstimes.com/singapore/fast-covid-19-tests-to-be-added-to-singapores-testing-arsenal-how-do-antigen-rapid-tests