Implementing Predictive Solutions with Hadoop and HDInsight

01 | Supervised Learning

Richard Conway | Microsoft Azure MVP, Elastacloud

- 01 | Introduction to Data Science with Apache Spark
- 02 | Building Machine Learning models
- 03 | Building Real-Time Machine Learning Solutions
- 04 | Course Exam

Hands-On Labs

- Microsoft Azure Subscription
  - Free trial available in some regions
- Client computer
  - Windows
  - Linux
  - Mac OS X

- What is machine learning? How does machine learning work?
- Is Machine Learning fast?
- How to … Machine Learning in Apache Spark
- How do I sample data?
- What is Quantization (Binning)? How do I reduce dimensions?
- What is normalization?

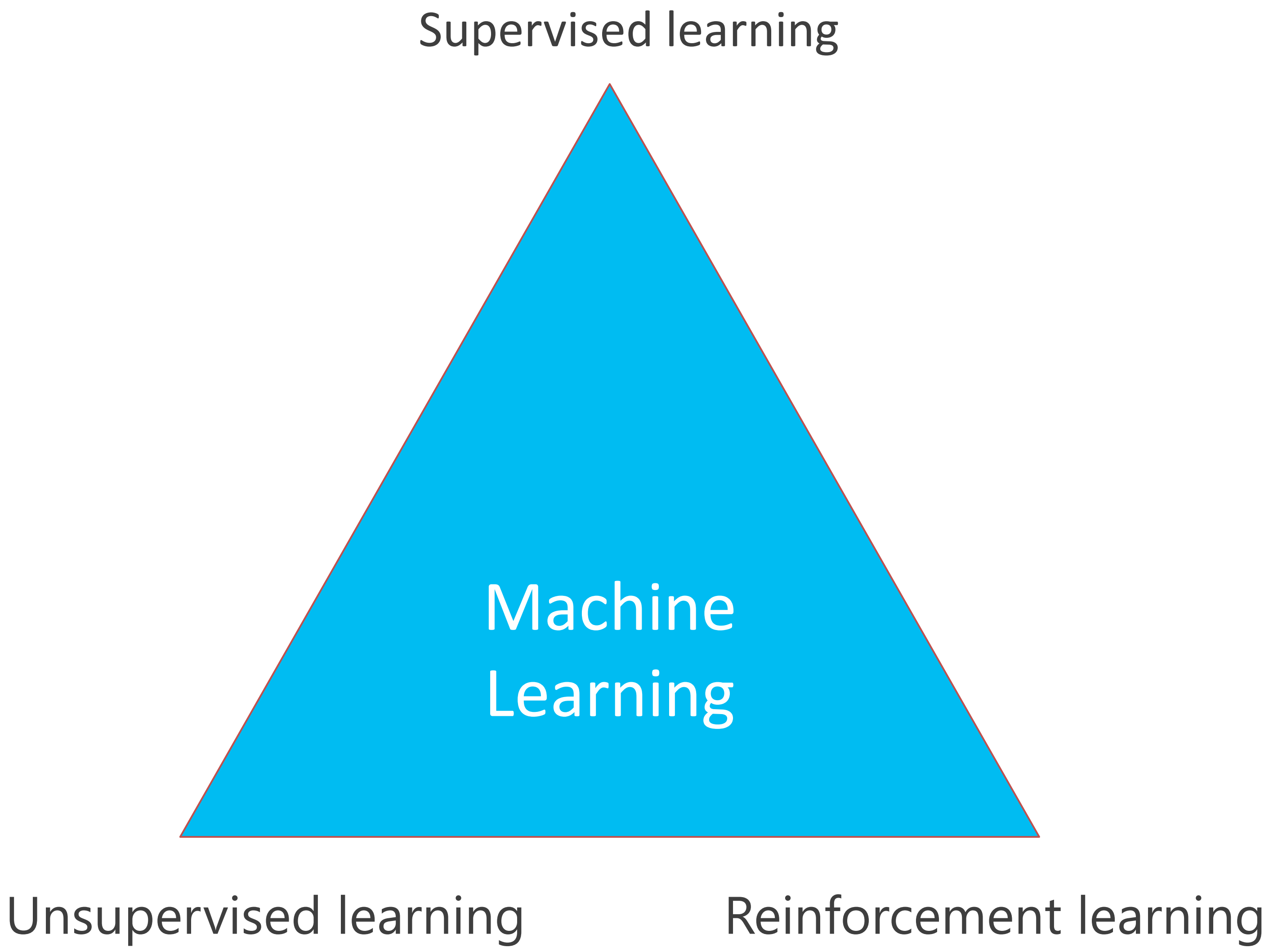
What is Machine Learning?

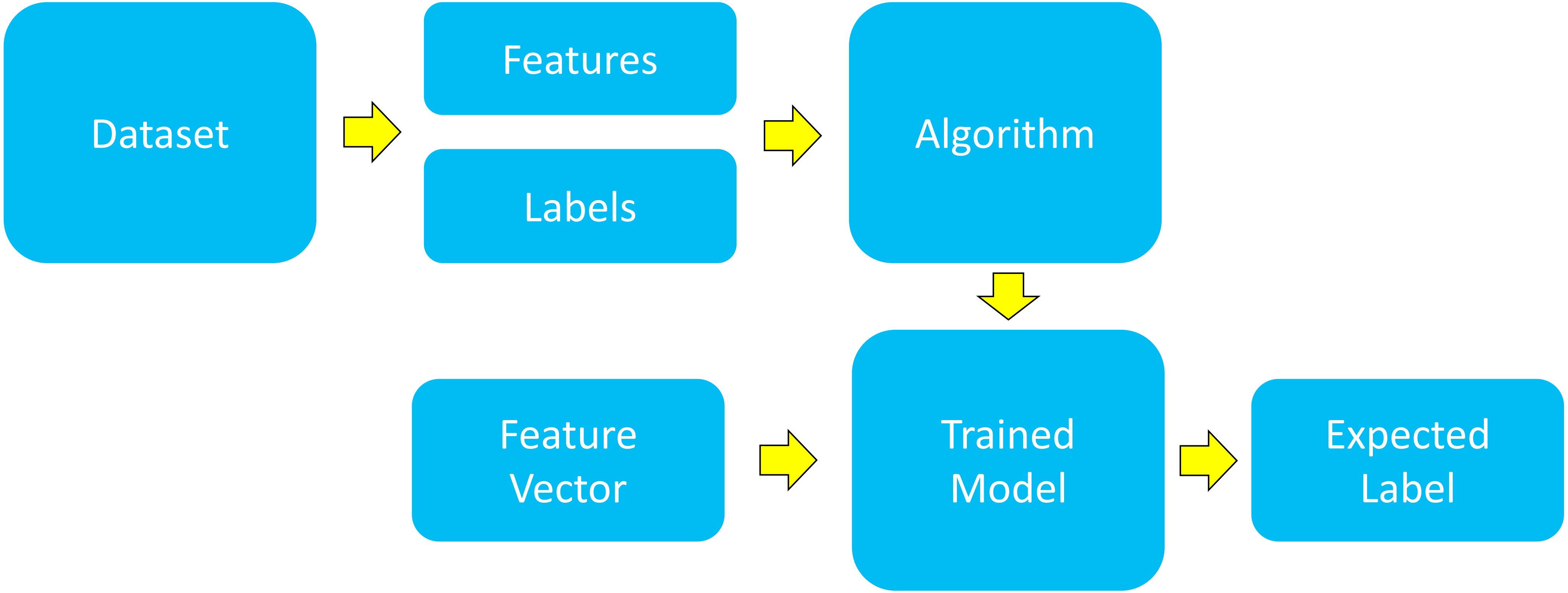
- Formal definition: “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E” - *Tom M. Mitchell*
- Another definition: “The goal of machine learning is to program computers to use example data or past experience to solve a given problem.” – *Introduction to Machine Learning, 2nd Edition, MIT Press*
- ML often involves two primary techniques:
  - Supervised Learning: Finding the mapping between inputs and outputs using correct values to “train” a model
  - Unsupervised Learning: Finding patterns in the input data (similar to *Density Estimates* in Statistics)

- Evolved from pattern recognition, computation and Artificial Intelligence
- Uses *algorithms* to make predictions on data
- Uses *models* to understand this particular data
- Uses feedback to learn how to make better predictions

How does Machine Learning work?

- Data labels
- Direct feedback
- Predict outcome





Dataset

Features

Labels

Algorithm

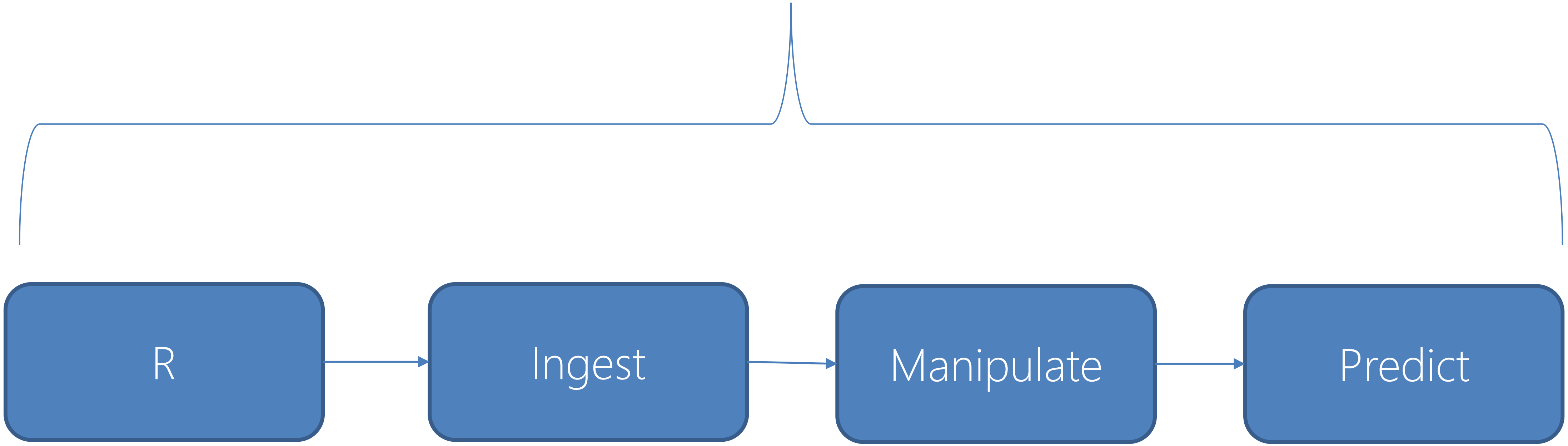
Trained Model

Feature Vector

Expected Label

Is Machine Learning fast?

Single Threaded

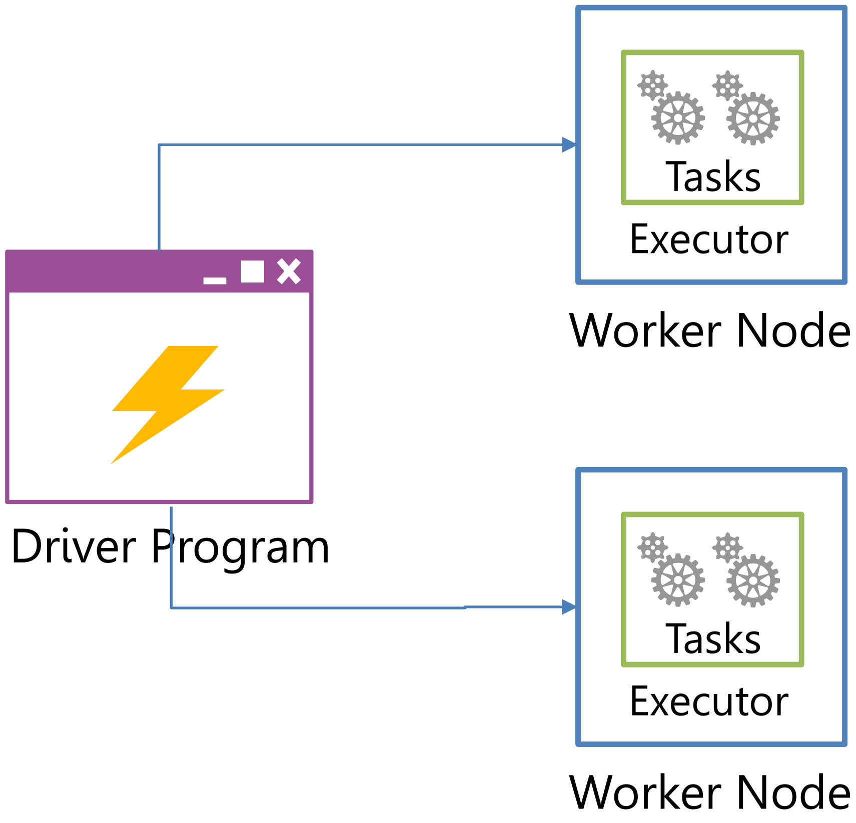

R

Ingest

Manipulate

Predict

```
iris <- read.csv(“C:\MicrosoftR\files\iris.csv”, HEADER= true)
irisframe <- as.data.frame(iris)
fit <- lm(y ~ x, data=irisframe)
summary(fit)
```


Tasks
Executor
Worker Node
Driver Program
Tasks
Executor
Worker Node

```
val rdd = sc.textFile(“wasb:///iris.csv”)
val model = DecisionTree.trainClassifier(trainingData,
numClasses, categoricalFeaturesInfo, impurity, maxDepth,
maxBins)
val labelAndPreds = testData.map { point =>
  val prediction = model.predict(point.features)
  (point.label, prediction)
}
```

How to .. Machine Learning in Apache Spark

- All primitives in Spark Machine Learning are *Vectors*
- *Features* are represented by a Vector
- Vectors can contain other Vectors and so be Dense or Sparse
- Spark uses *LabeledPoints* to encapsulate a Vector and a Label
- RDDs are transformed into Vectors through map functions

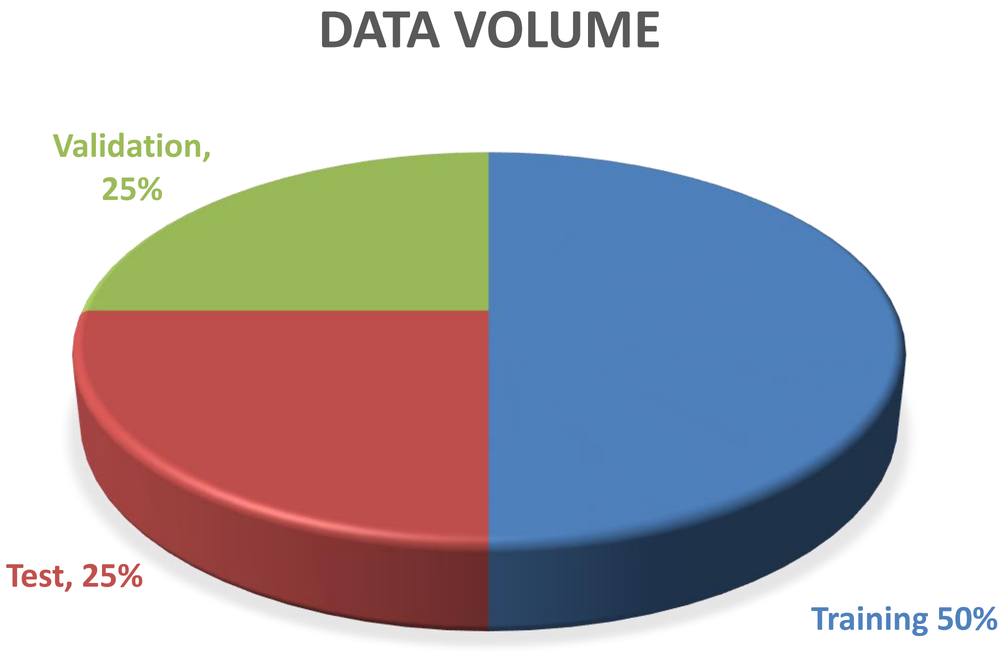
Umbrellas sold	Wind Speed / mph	Rainfall / inches	Temperature / F
10	8	0.2	65.1
56	12	2.1	64.6
70	7	3.0	67.3
21	5	1.5	65.3
4	4	0.1	65.1

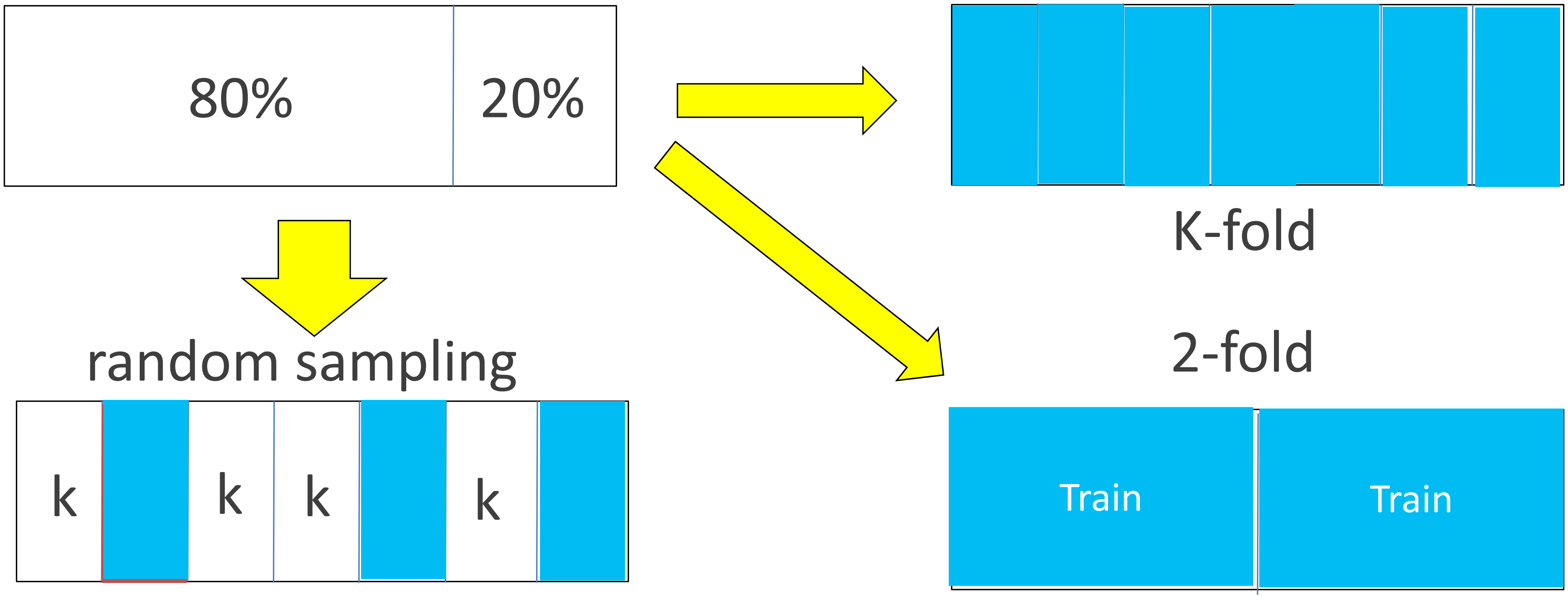
Label	Feature	Feature
1.0	A	We
0.0	B	Are
1.0	C	No
1.0	A	Yes

```
categoricalFeaturesInfo = Map[Int, Int]((1,3),(2,4))
val model = DecisionTree.trainClassifier(trainingData, numClasses,
categoricalFeaturesInfo, impurity, maxDepth, maxBins)
```

How do I sample data?

- 3 types of data for ML
  - Training : train your model over this dataset
  - Validation: use this data to validate the model
  - Testing: assess the generalization of the model





80%

20%

random sampling

k

k

k

k

K-fold

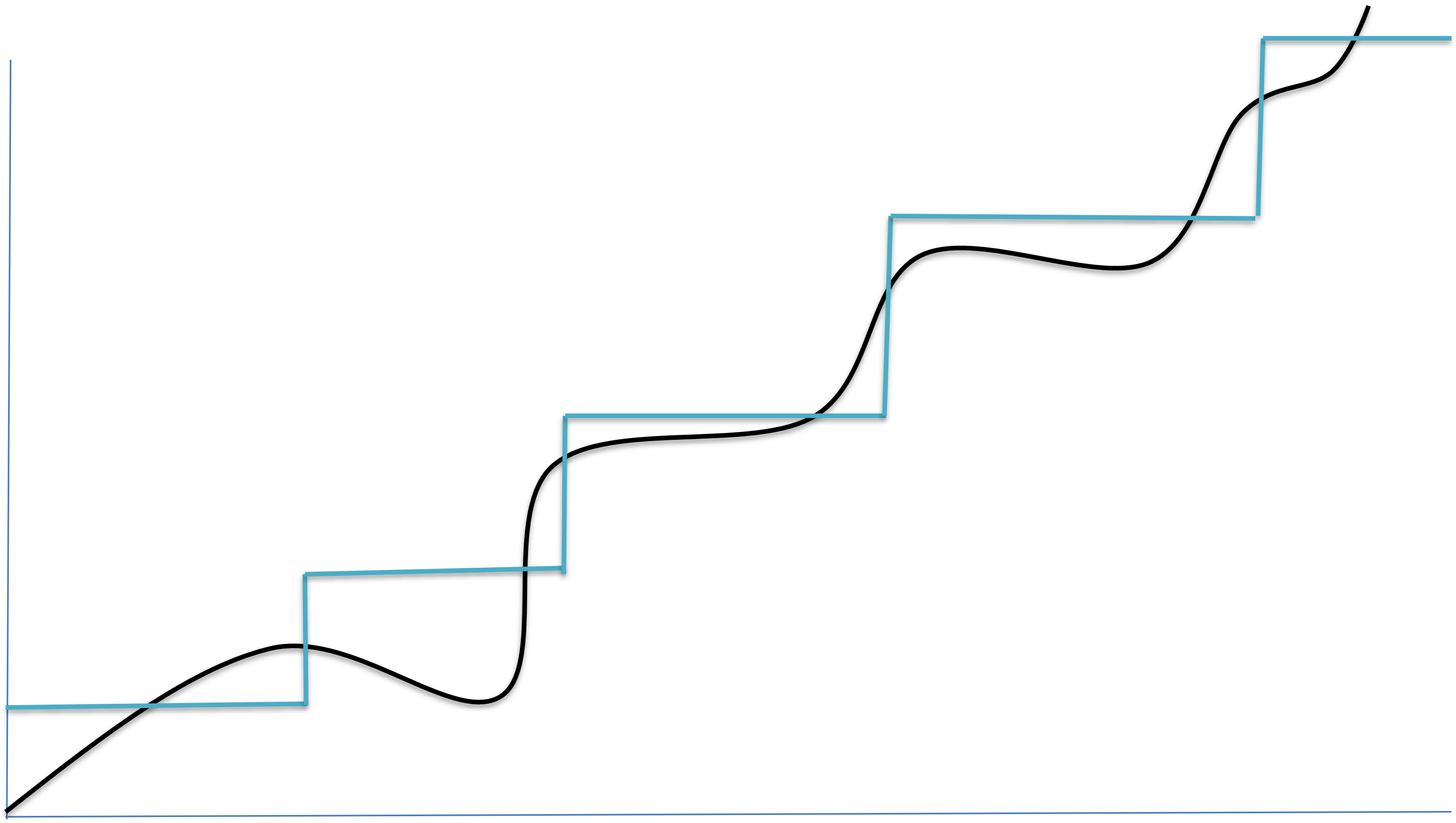
2-fold

Train

Train

- Useful to cherry pick data from a dataset to “cross validate” for machine learning
- Can assign data to “folds” so that you can operate on particular random sampled subsets
- Can take a “stratified” approach and pull data from a different sections of the dataset
- Supports Folds, Sampling and Top ‘n’ Rows

What is Quantization (Binning)?

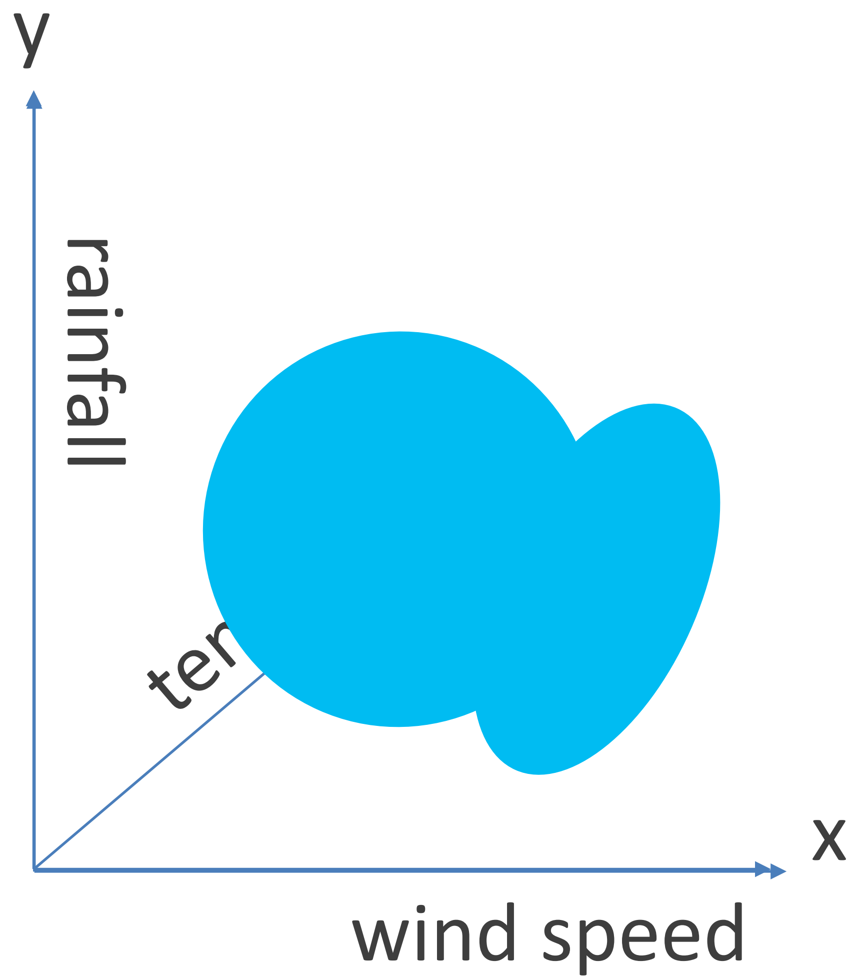


- Common for
  - DSP
  - MPEG/JPEG
- What is it
  - Replaces discrete values with binned values
  - Uses a coefficient matrix to determine best fit binned values

```
import org.apache.spark.ml.feature.QuantileDiscretizer

val metrics = Array((1, 10.2), (2, 17.1), (3, 9.6), (4, 5.0), (5, 3.4))
val df = metrics.toDF(“day", “rainfall")
val discretizer = new QuantileDiscretizer()
        .setInputCol(“rainfall")
        .setOutputCol(“discreterainfall")
        .setNumBuckets(3)
val result = discretizer.fit(df).transform(df)
result.show()
```

How do I reduce dimensions?

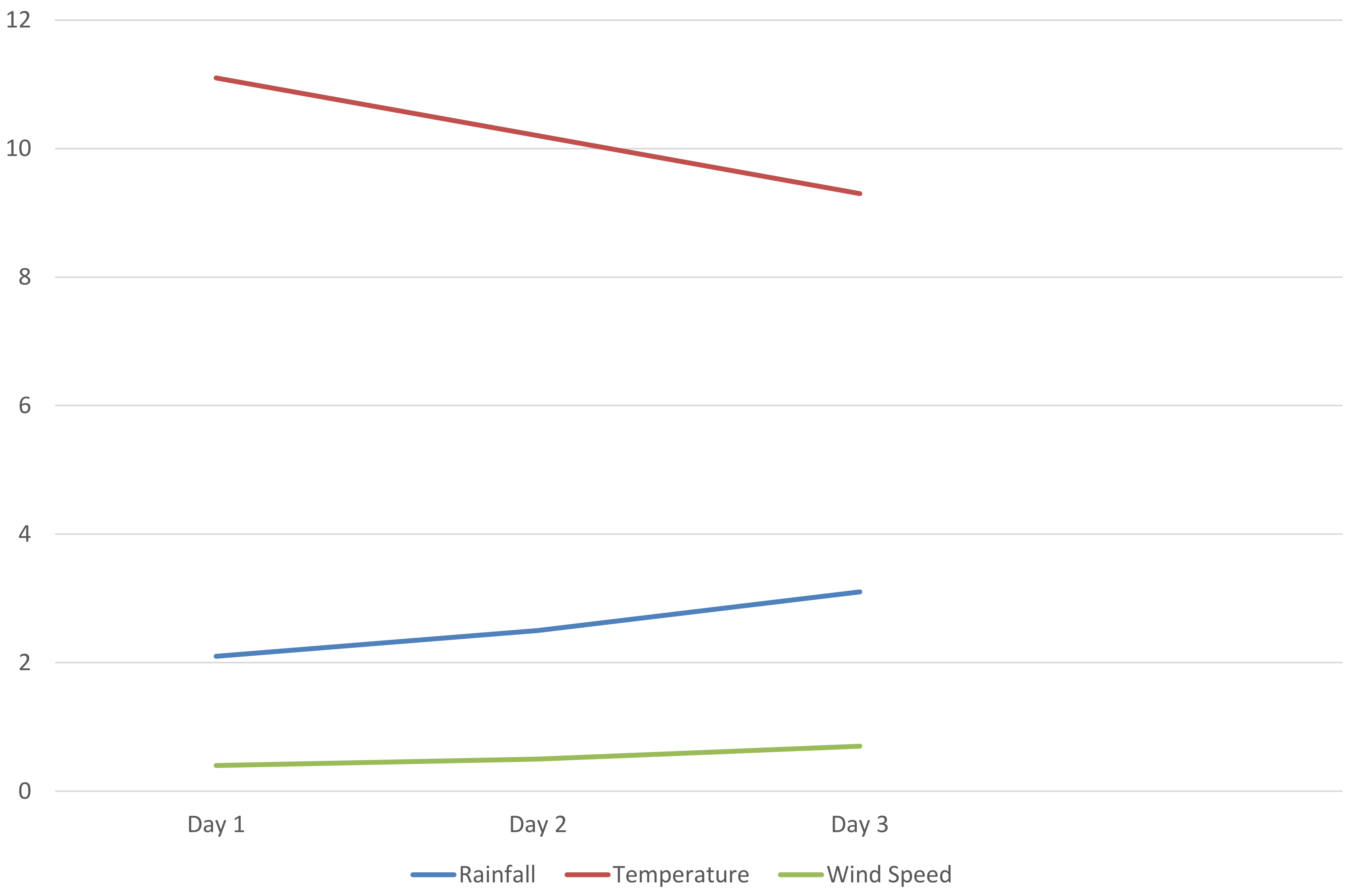


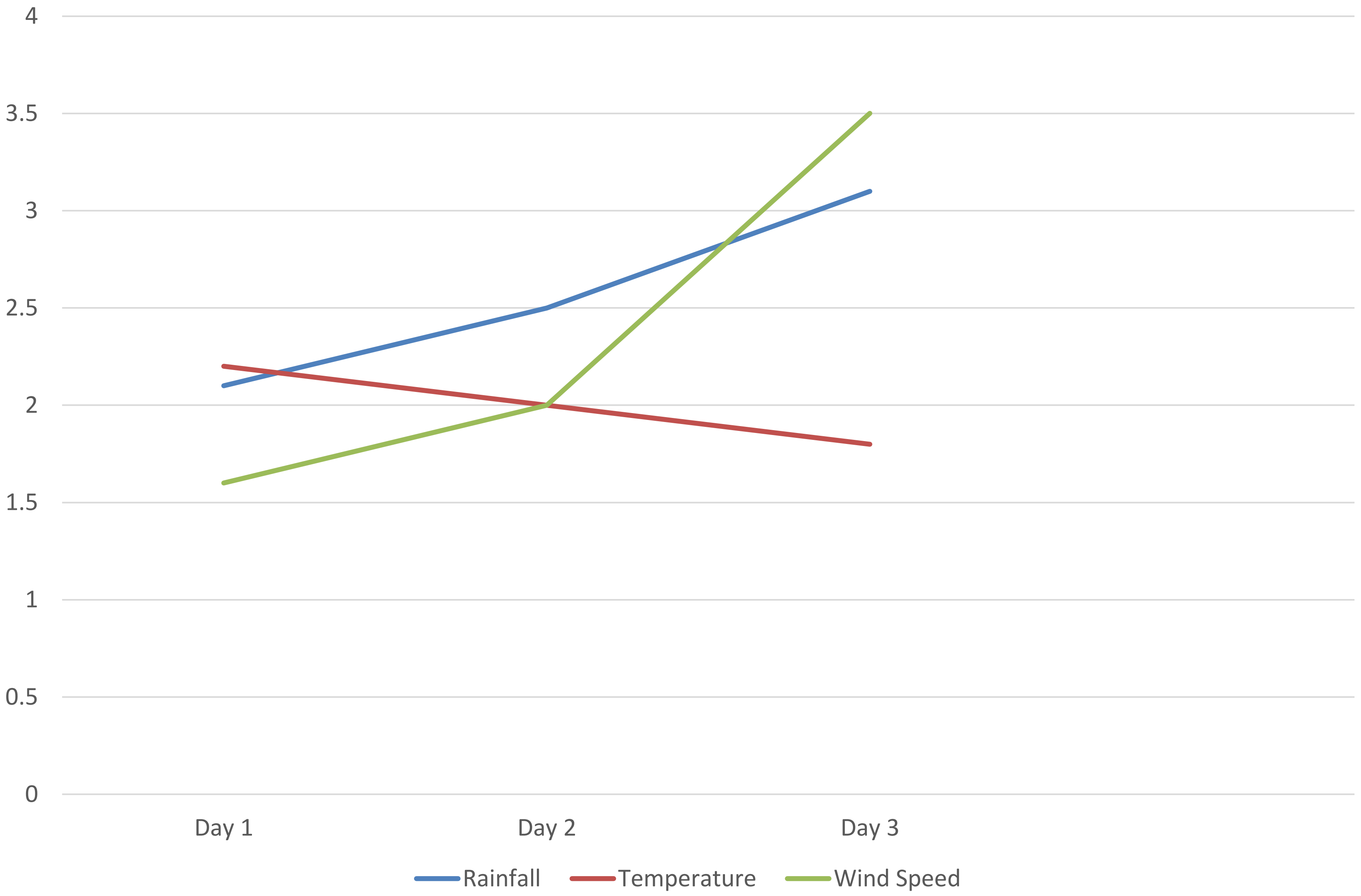
- Common for
  - Used for dimensionality reduction
  - Uses Eigenvectors and eigenvalues to determine most relevant features and rescale
  - Allows plotting in 2D
  - Speeds up calculation
  - Lose some information

```
from pyspark.mllib.feature import PCA
from pyspark.mllib.linalg import Vectors
points = parsedData.map(lambda point :
Vectors.dense(point[0:4]))
pcamod = PCA(2).fit(points)
transformed = pcamod.transform(points)
```

What is normalization?

- Normalization
  - Transform columns in a dataset to a common scale
  - Log, tanh, logistic, min-max, ZScore options
- Clip Values
  - Clip peaks/subpeaks of distribution
  - Replace or remove values
  - Work on absolute values or percentile


12
10
8
6
4
2
0
Day 1
Day 2
Day 3
Rainfall
Temperature
Wind Speed


4
3.5
3
2.5
2
1.5
1
0.5
0
Day 1
Day 2
Day 3
Rainfall
Temperature
Wind Speed

```
val input = sc.textFile(“normal.txt”)

val normalizer = new Normalizer()

val transformed = input.map(x => (x.label,
normalizer1.transform(x.features)))
```