BIG DATA PROCESSING

Machine Learning: Introduction

Machine Learning Topics

- Machine Learning Concept
  - Introduction
  - Algorithms
  - Data preprocessing
  - Model evaluation
- ML with Spark
  - Spark in ML
  - Applications
  - Deployment

Data preparation → Data Preprocessing

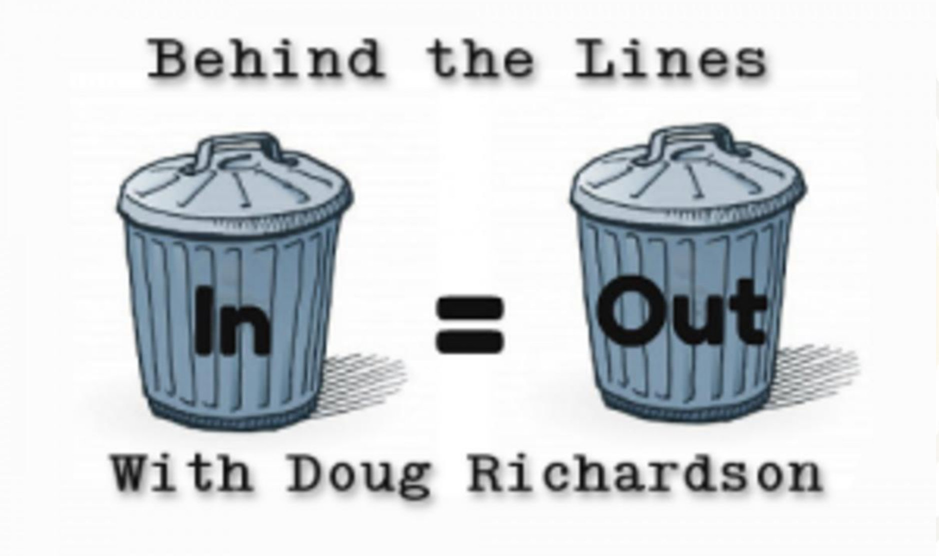
- Data quality issue
  - Inconsistent value
  - Duplicate records
  - Missing values
  - Invalid data
  - Outlier

Data preparation2 → Data Preprocessing

- Domain Expert
  - Remove Duplicate data
  - Remove Missing values
  - Generate range for Invalid data
  - Remove Outlier

Data processing techniques

- Data manipulation/preprocessig/ data wrangling
  - Scaling the data
  - Data transformation
  - Feature Selection
  - Dimensionality reduction



Data Analysis

- Build Model
  - Classification
  - Regrssion
  - Clustering
  - Association

Modelling

- Model training
  - Supervised
  - Unsupervised training

Evaluation

- Accuracy?
- Senstivity?
- Recall?
- Precision?

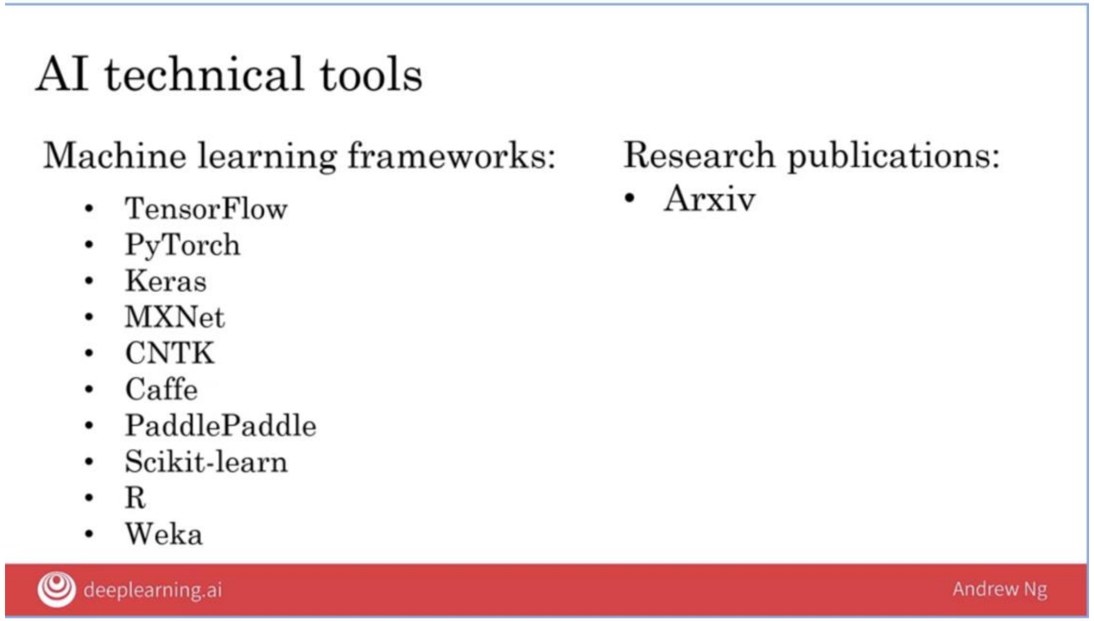
Model deployment

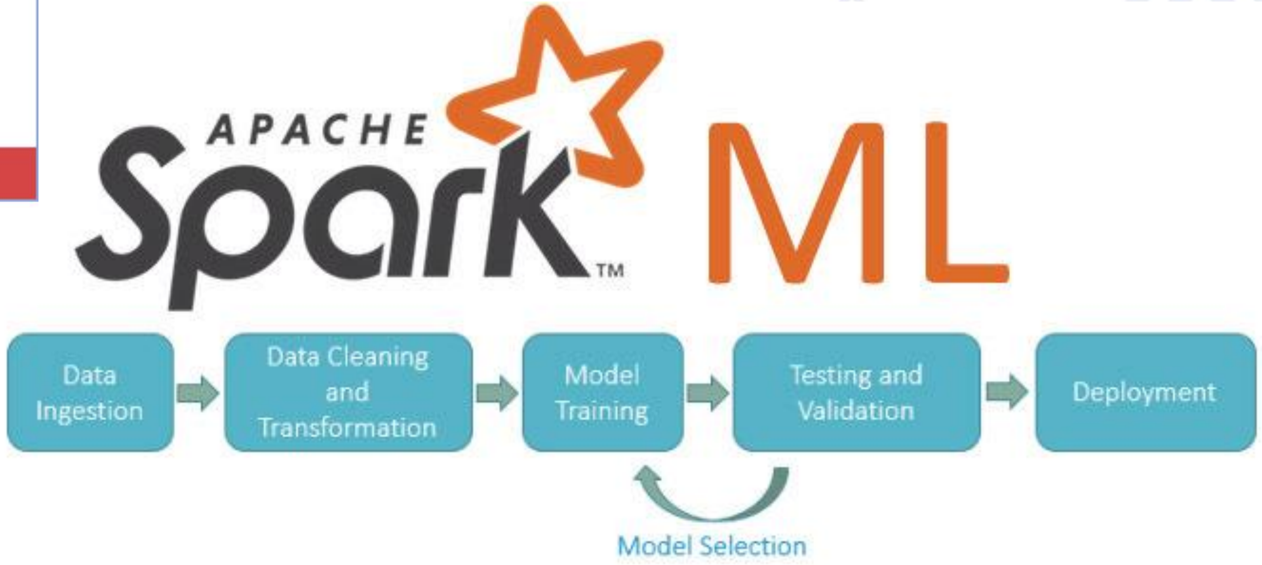
- Put it to work
- Get more feedback

Tools

- Volume
  - unstructured data
- Velocity
  - fast
  - real-time
- NoSQL

Value?


AI technical tools
Machine learning frameworks:
• TensorFlow
• PyTorch
• Keras
• MXNet
• CNTK
• Caffe
• PaddlePaddle
• Scikit-learn
• R
• Weka
Research publications:
• Arxiv
deeplearning.ai
Andrew Ng


APACHE Spark ML
Data Ingestion
Data Cleaning and Transformation
Model Training
Testing and Validation
Deployment
Model Selection

Machine Learning Overview

It helps

- Learn from data
- Discover patterns and trends
- Allow for data –driven decision
- Used in many different applications

Machine Learning Techniques

- Supervised Learning
  - Regression
  - Classification

- Unsupervised Learning
  - Association Analysis
  - Cluster Analysis

- Reinforcement Learning
  - Penalty through behaviour

Supervised Learning

- Target/Label is provided
- Data are ‘Labelled’
- Evaluate mostly based on closeness/correctness of the answer

Target

Today’s High	Today’s Low	Month	Tomorrow’s High
79	64	July	81
60	45	October	58
68	49	May	65
57	47	January	54

Regression

- Output in numeric values

- Examples
  - Forecast temperature(tomorrow)
  - Estimate house price
  - Determine demand
  - Stock price
  - Power usage

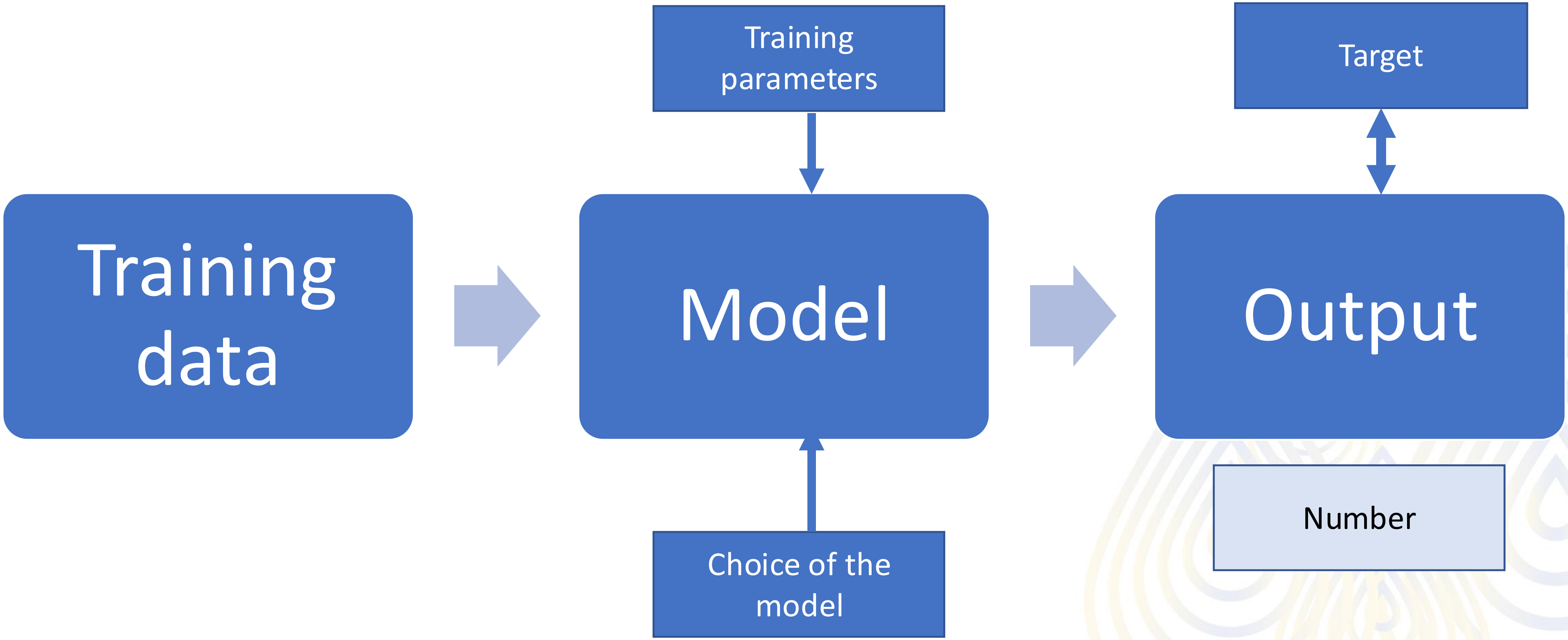
Target

Today's High	Today's Low	Month	Tomorrow's High
79	64	July	81
60	45	October	58
68	49	May	65
57	47	January	54

input

Regression Model

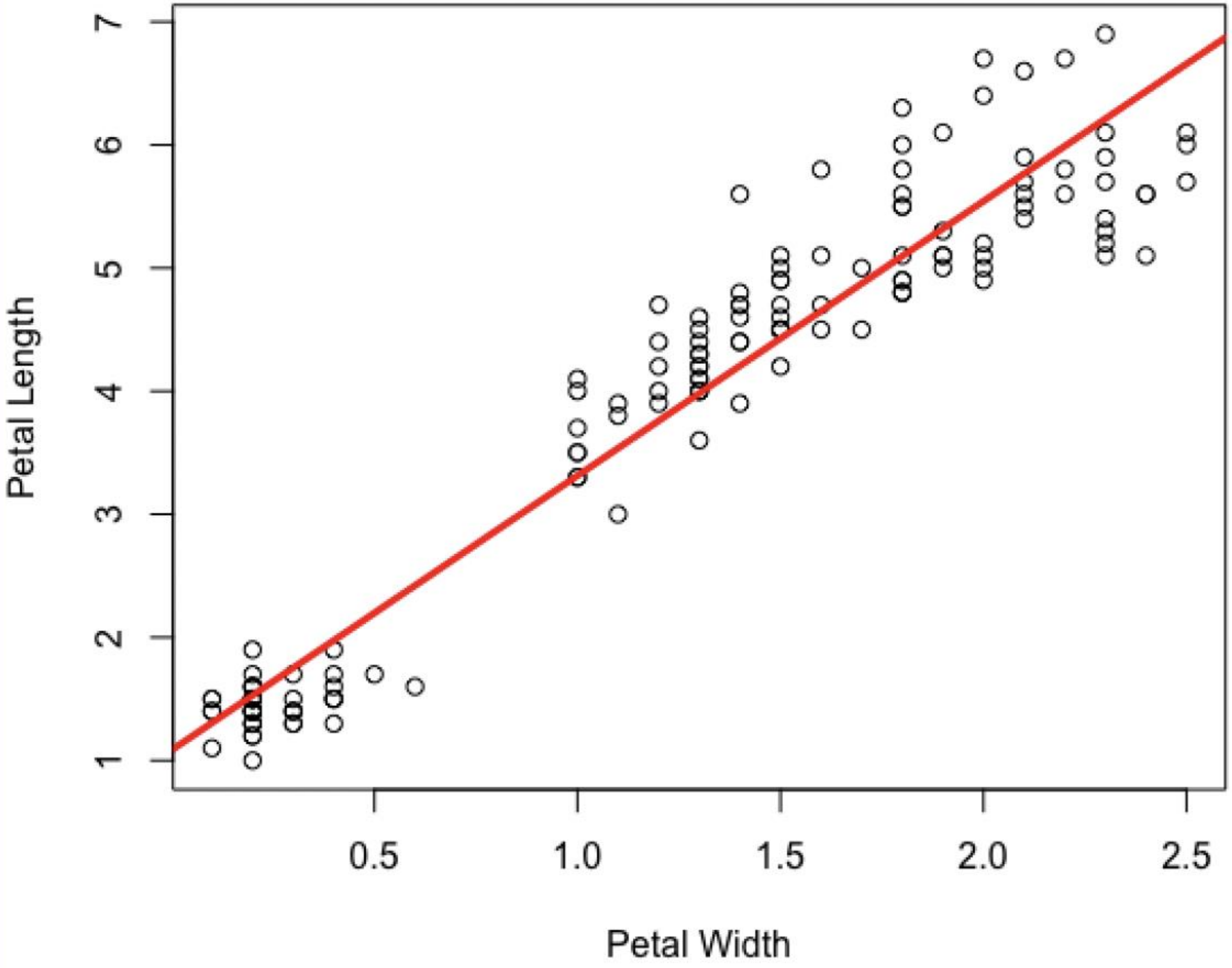
Model Validataion



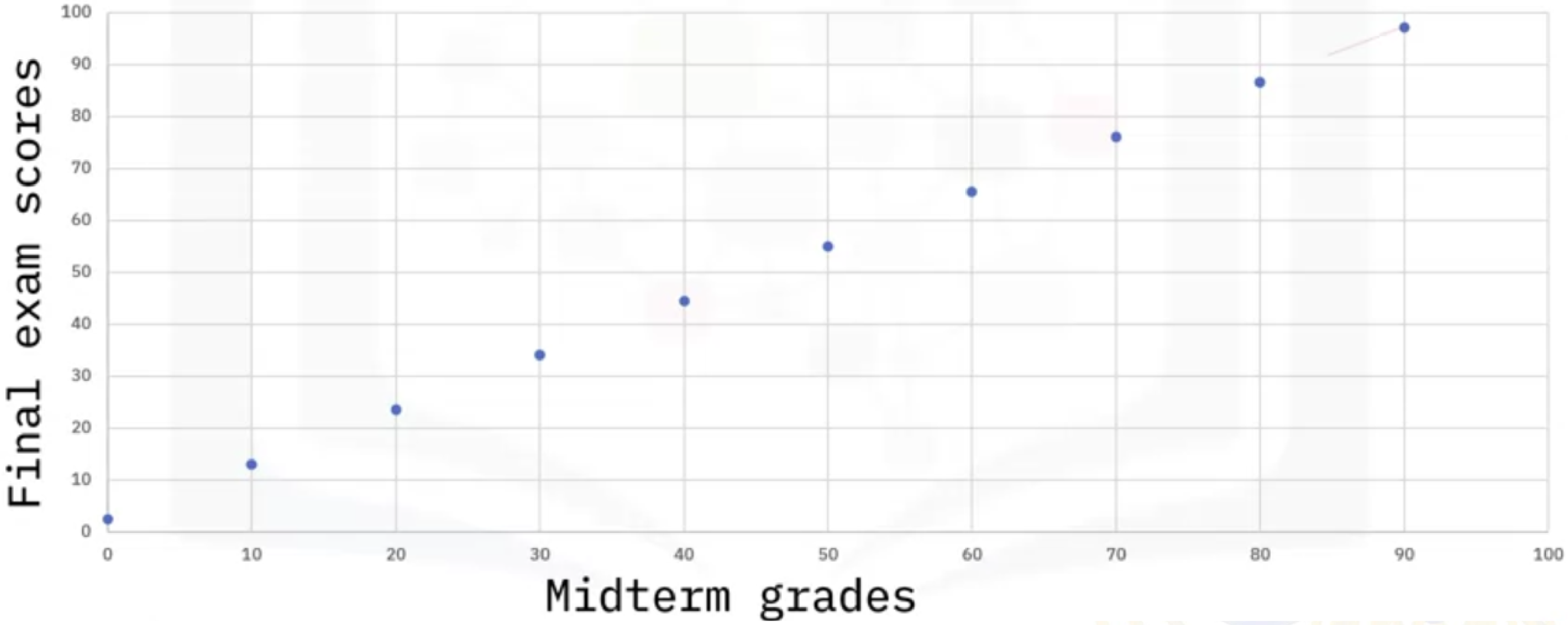
Linear Regression

- Predict linearly
- Simplest one is Least Square Method
- Measure error by the distance line to data

$y=ax+b$

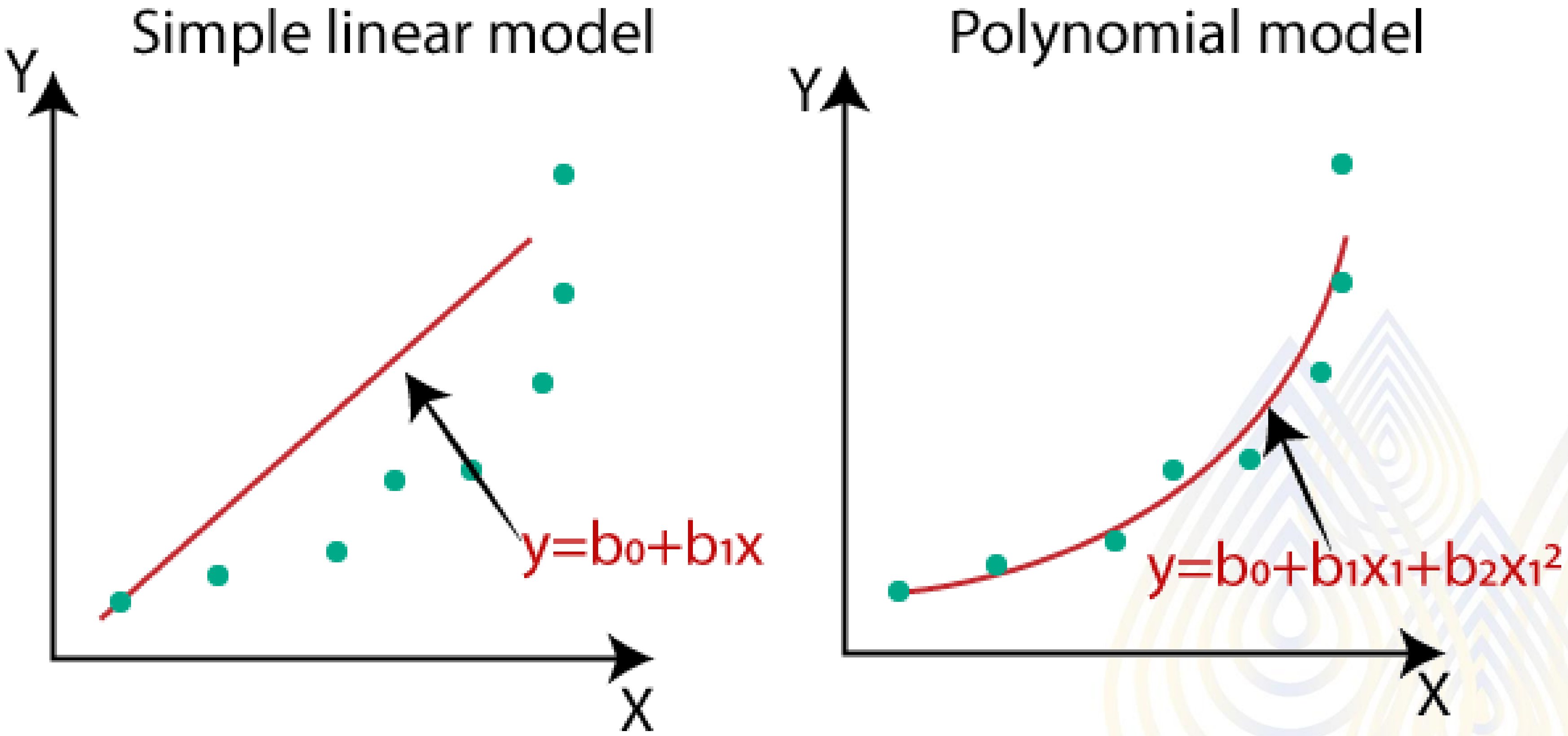


Simple Linear Regression



Simple Linear Regression

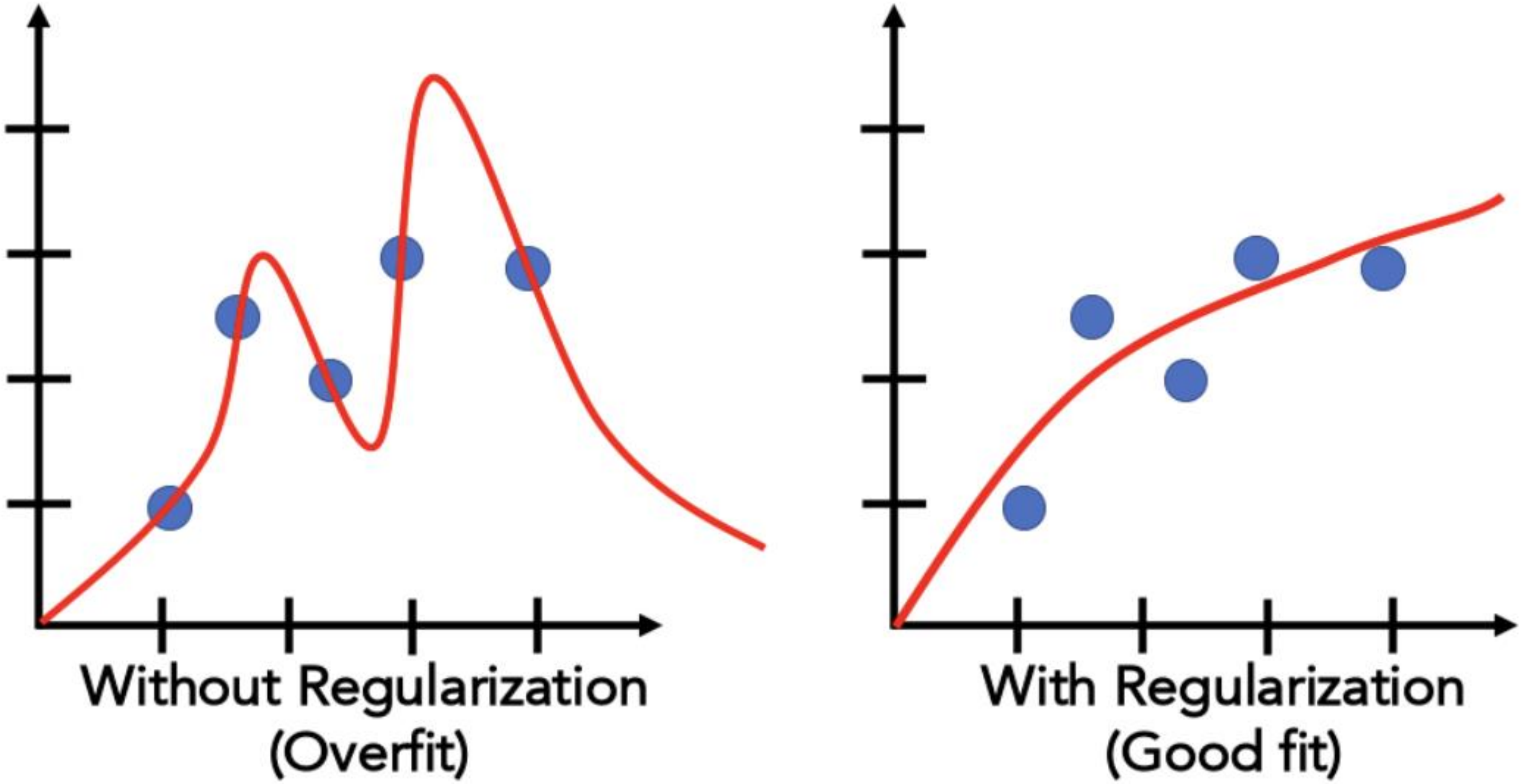
Non-Linear?


Simple linear model
Y
X
y=b0+b1x
Polynomial model
Y
X
y=b0+b1x1+b2x1²

Problems?

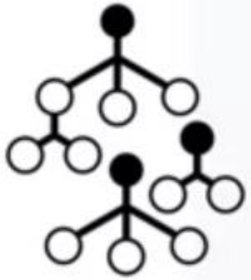
Some regularized Regression

- **Ridged Regression**
- **Lasso Regression**



Famous Regression algorithm

Random forest

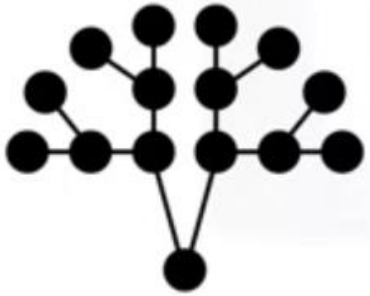


Random forest is a group of decision trees combined into a single model

Support vector regression

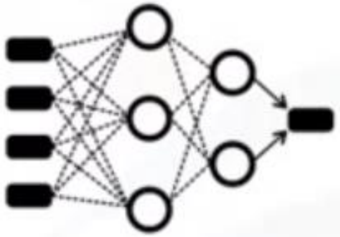
SVR creates a line or a hyperplane that separates the data into classes

Gradient boosting



Gradient boosting makes predictions by using a group of weak models like decision trees

Neural networks



Neural networks function loosely like the neurons in the human brain to make predictions

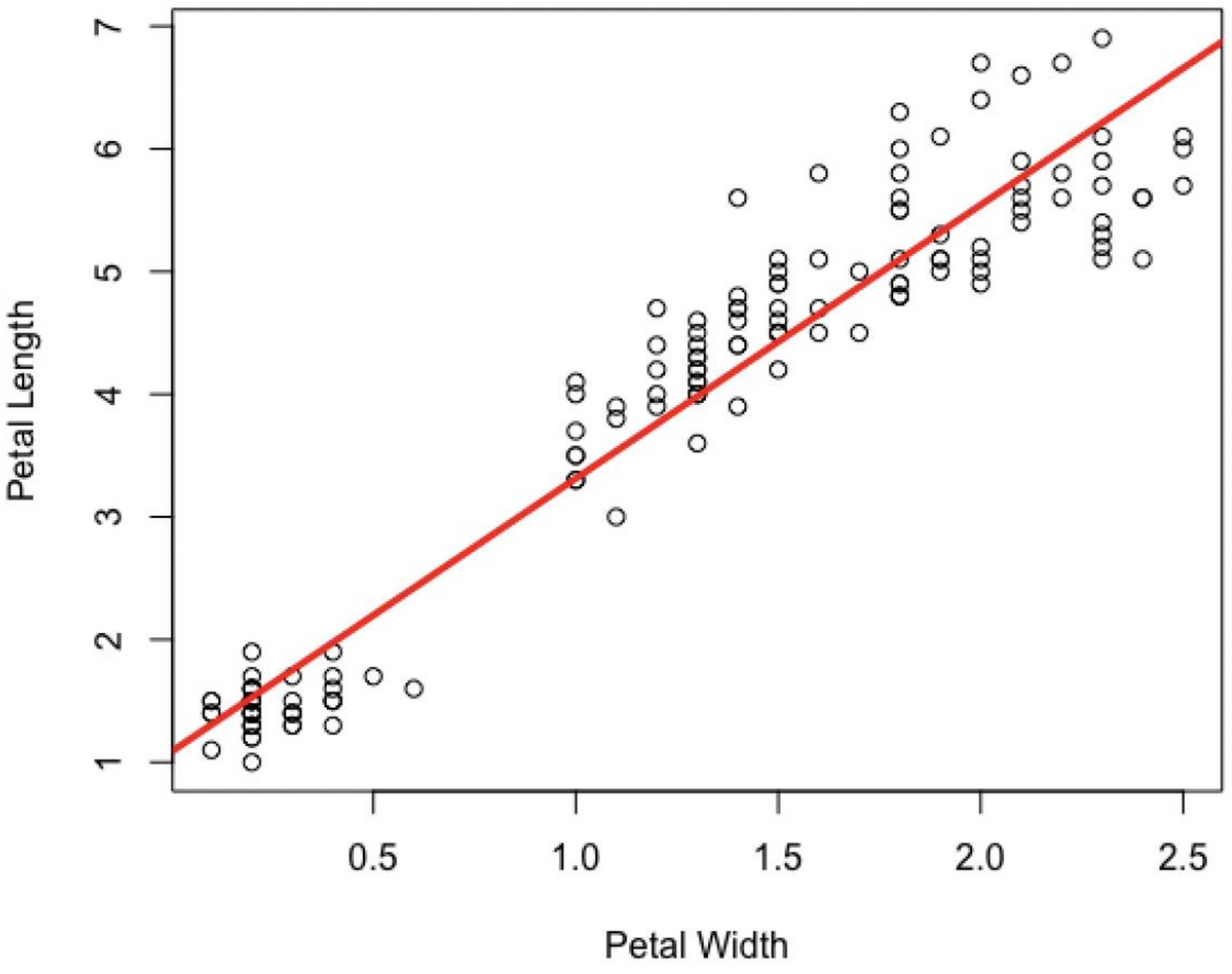
BIG DATA PROCESSING

Regression

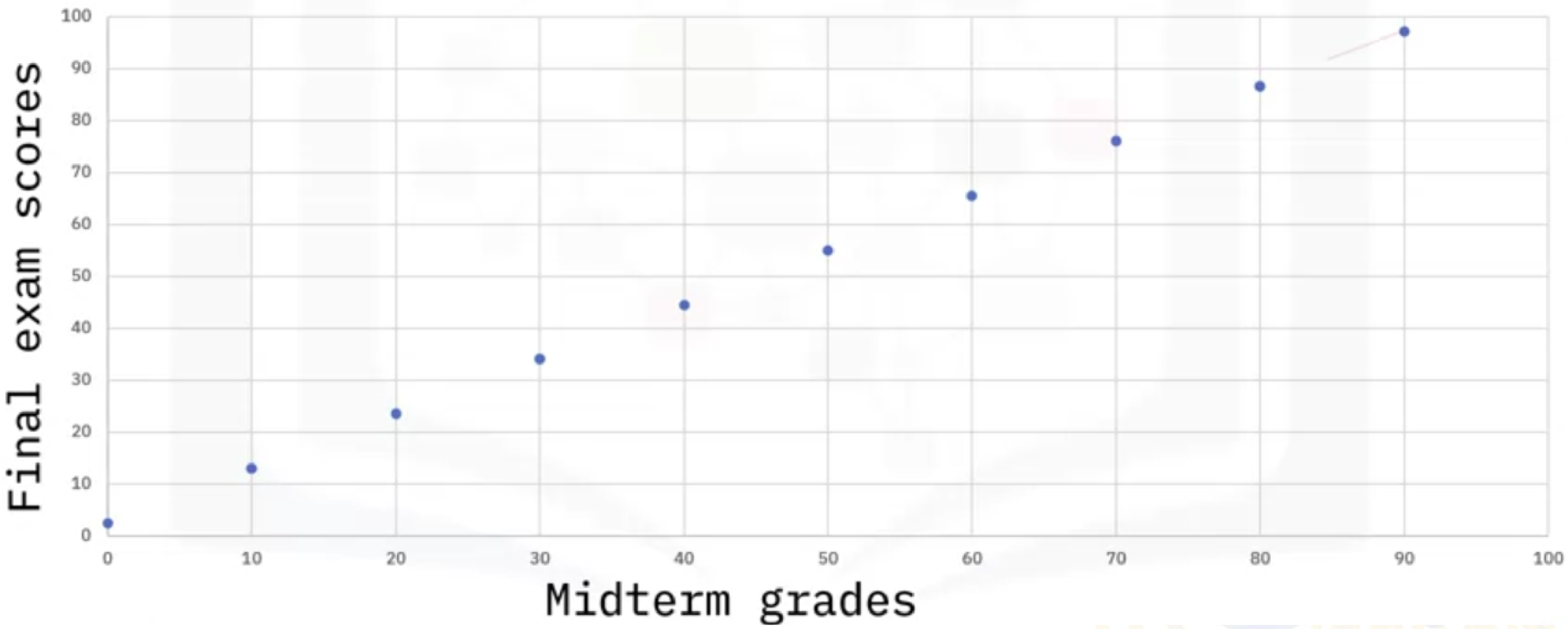
Linear Regression

- Predict linearly
- Simplest one is Least Square Method
- Measure error by the distance line to data

y=ax+b

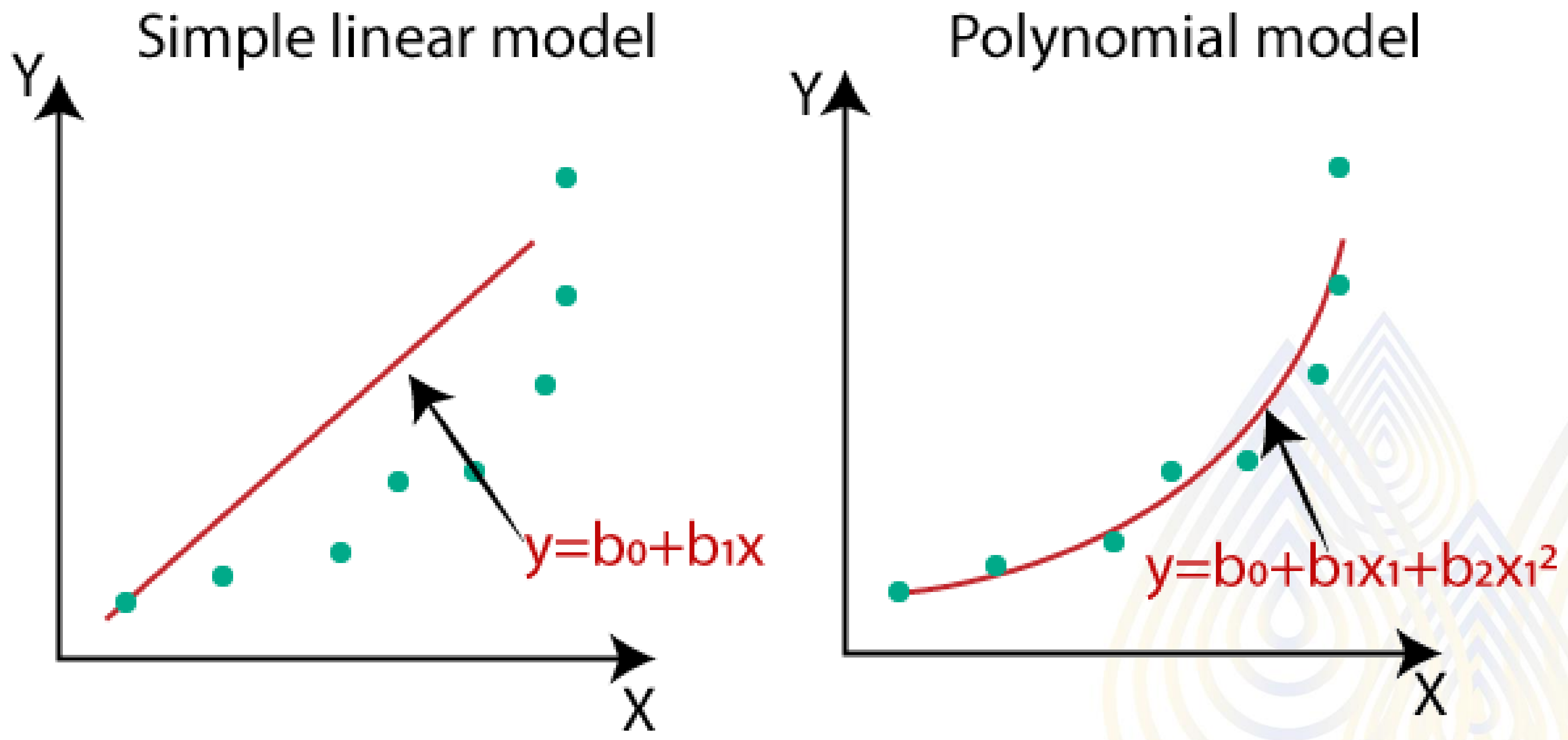


Simple Linear Regression



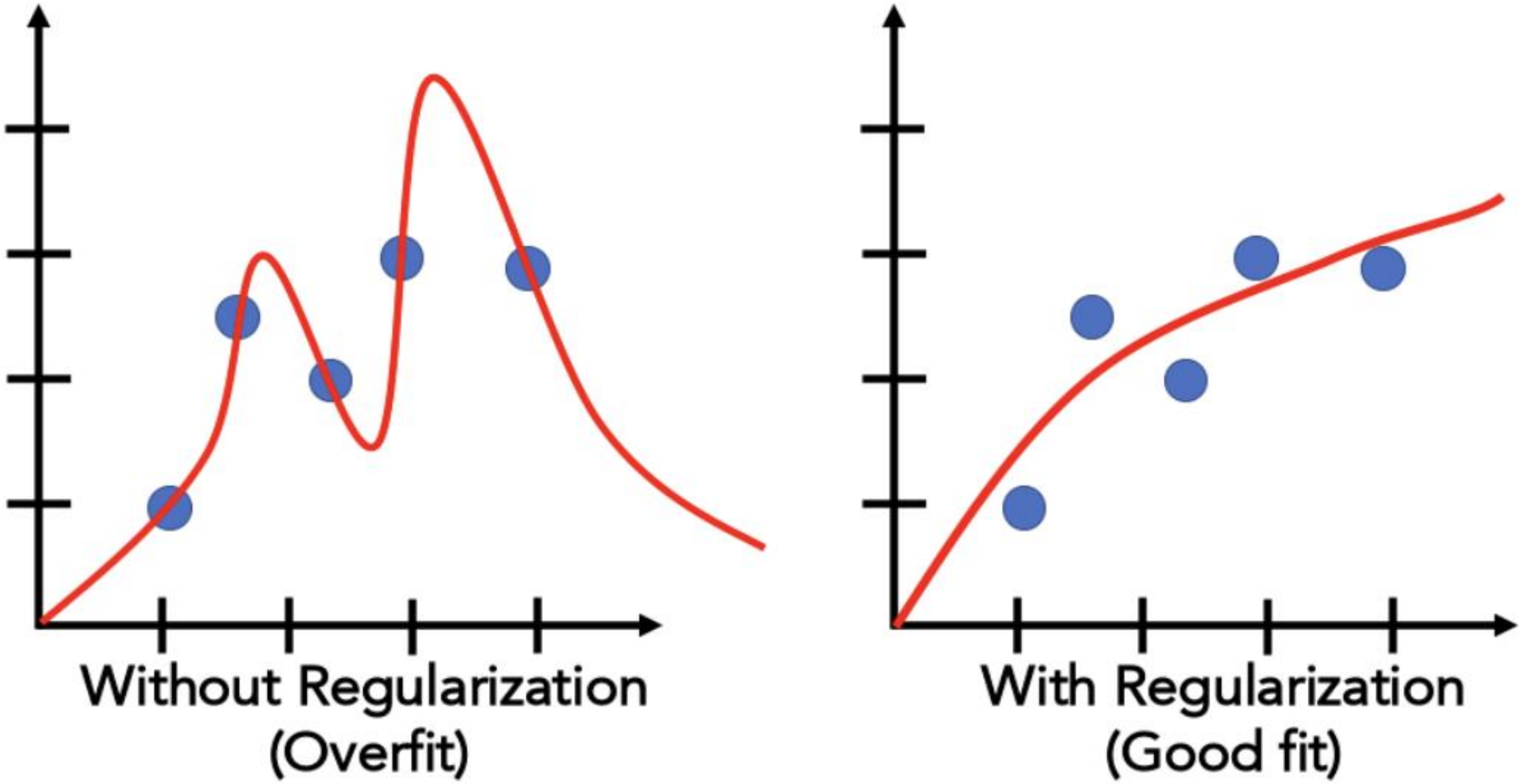
Simple Linear Regression

Non-Linear?


Simple linear model
Y
X
y=b0+b1x
Polynomial model
Y
X
y=b0+b1x1+b2x1²

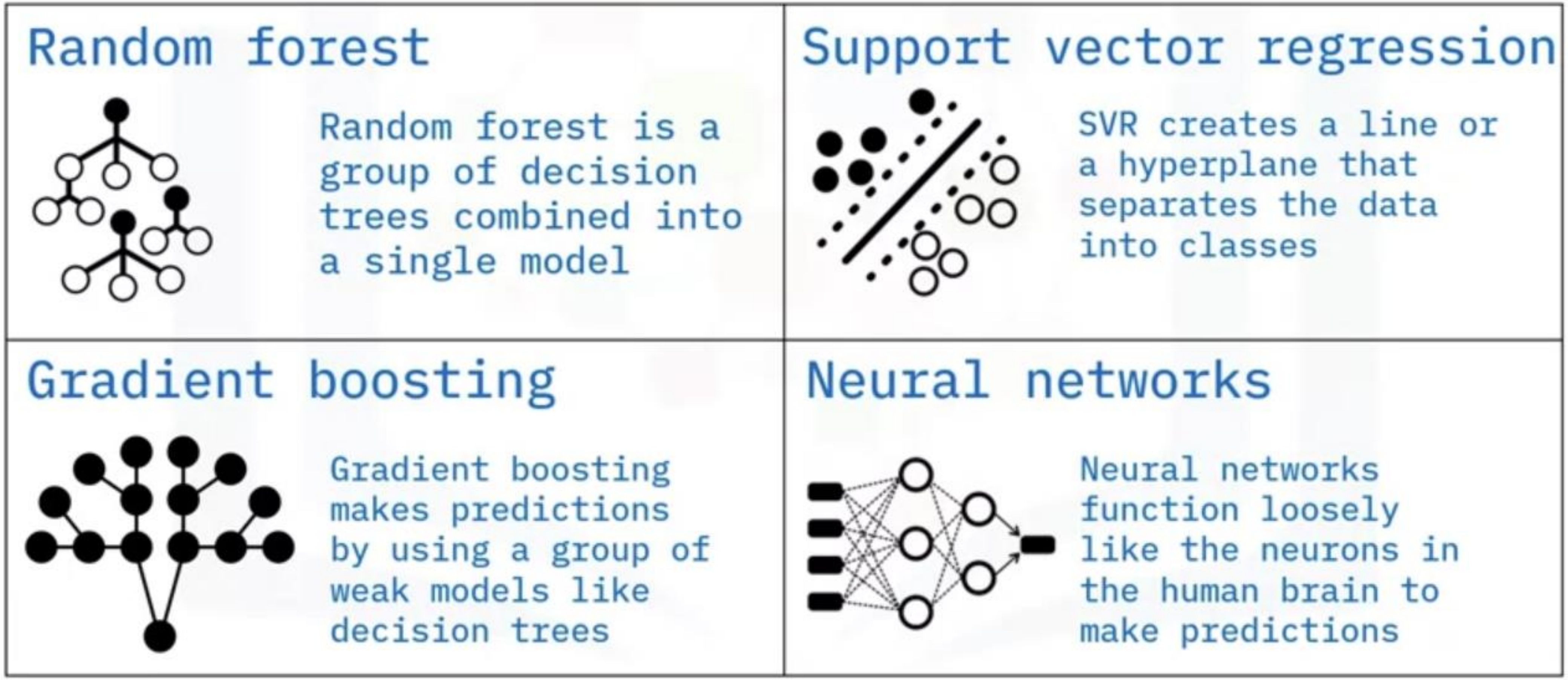
Problems?

Some regularized Regression
- **Ridged Regression**
- **Lasso Regression**

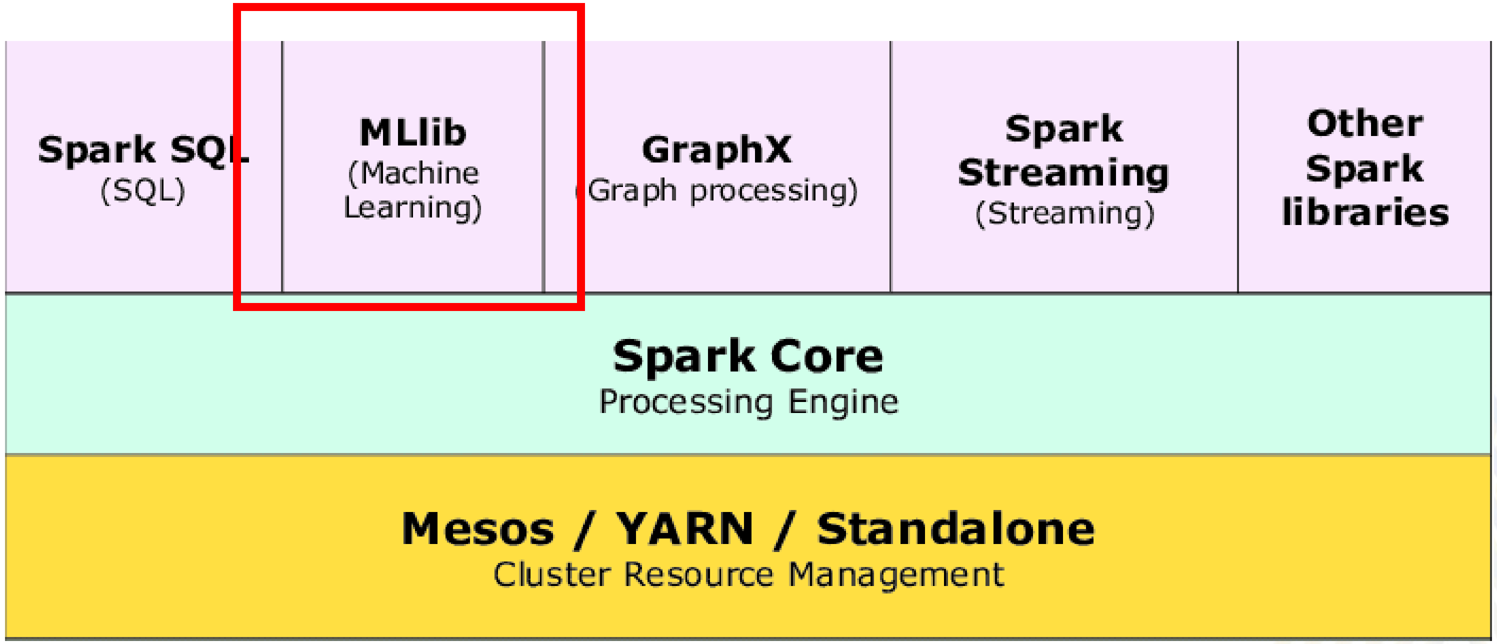


Mahidol University
Wisdom of the Land

Famous Regression algorithm



SPARK ML



Spark SQL
(SQL)
MLlib
(Machine Learning)
GraphX
(Graph processing)
Spark Streaming
(Streaming)
Other Spark libraries
Spark Core
Processing Engine
Mesos / YARN / Standalone
Cluster Resource Management

SPARK ML STEPS

- Create Spark Session
- Import the data
- Data cleansing/preparation
- Combine features using Vector assembler
- Split data into training and testing data
- Choose/Create model for regression
- Fit the model to the training data (train)
- Make prediction on the test data (test)
- Evaluate the model.
- STOP THE SESSION

Spark ML commands

- Drop missing data
  - `df=df.na.drop()`
  - `df=df.na.drop('col')`
- Drop the column
  - `df=df.drop('col')`
- `Binarized the data`
  - `binarizer=Binarizer(threshold = xx, inputcol='input',outputCol='output')`

Spark ML Example

- Aggregate to feature into a single column (Vector Assembler)
  - `assembler=VectorAssembler(inputCols=[list of features], ouputCol=“features”)`

- Split train and test
  - `(trainData, testData) = assembled.randomSplit([0.8,0.2],seed=123)`

Spark ML Training

- Create a model
  - `lr=LinearRegression(labelCol= ‘lable’,featuresCol=“features”)`
- Fit the model
  - `model=lr.fit(trainData)`
- Create a pipeline
  - `pipeline=Pipeline(stages=[ STEPFORTHEPIPELINE])`

Spark ML Evaluation

- Make Prediction
  - `predictions= model.predict(testData)`
- Evaluate (RMSE)
  - `Evaluator=RegressionEvaluator(labelCol="label", Predictioncol="features", metricName="rmse")`
  - `rmse=evaluator.evaluate(predictions)`

Demo

- https://github.com/AjMing/BigData/tree/main/SparkML

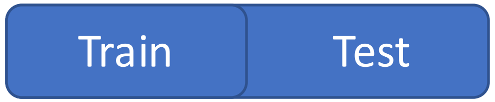
Summary of the command

- Model selectors
  - `model = LinearRegression()`
- Train the model
  - `model = model.fit(train)`
- Predict
  - `model = model.transform(train)`
  - `model = model.transform(test)`

Evaluation Methods

The holdout technique

- **Split dataset into two groups**
  - Training set: used to train the classifier
  - Test set: used to estimate the error rate of the trained classifier



Advantage:

- Simple and easy
- Good for big enough data
- Fast

Disadvantage:

- It can be biased

The holdout method

The holdout method’ s drawbacks

- Small sample size can’t **split** dataset for both training and testing
- Bad splitting can create an unpleasant outcome

Cross validations can overcome this problems

Error Rate for model performance

- **Mean square Error** (MSE)
  - To normalised to the number of data

$$E = \frac{1}{N}\sum (y-t)^2$$

- **Mean Absolute Error** (MAE)

$$E = \frac{1}{N}|(y-t)$$

- Root mean square Error **(RMS)**
  - To measure the precision more common for regression

$$E = \sqrt{\frac{1}{N}\sum (y-t)^2}$$

EXAMPLE

Size (sqft)	Number of Bedrooms	Location	Actual Price ($)	Predicted Price ($)	Error
1500	3	1	300000	310000	10000
1600	3	2	320000	315000	-5000
1700	4	1	350000	345000	-5000
1800	4	3	370000	375000	5000
1900	5	2	400000	390000	10000

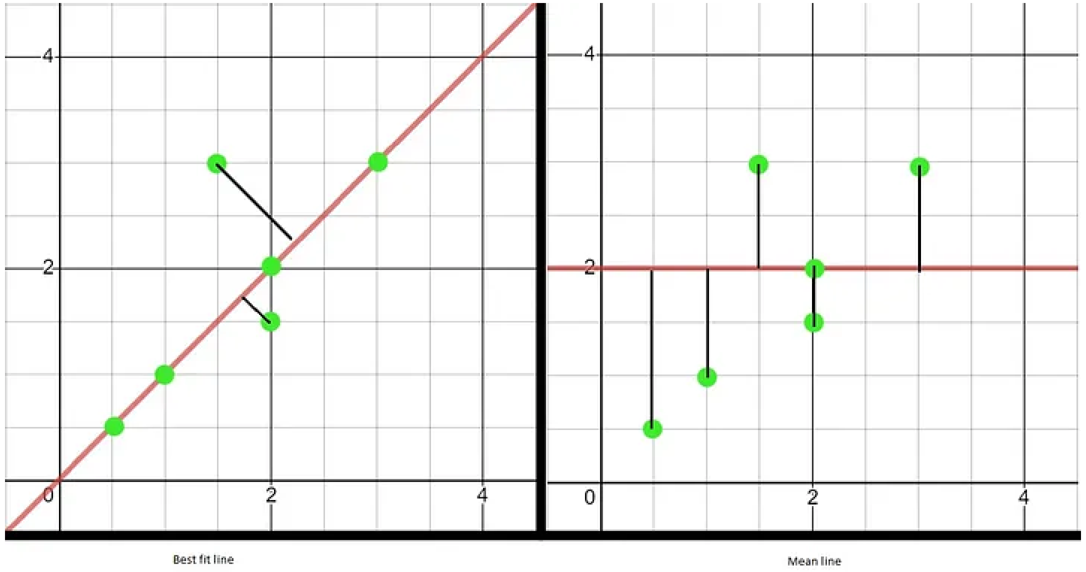
MSE = (1/5) * (10000^2+5000^2+5000^2+5000^2+10000^2) =55,000,000

MAE = (1/5) * (10000+5000+5000+5000+10000) =7,000

RMSE =sqrt(MSE)=~7416

R-Squared

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$



Interpretation

•**R^2 = 1**: The model explains all the variability of the response data around its mean. The predictions perfectly match the actual data.

•**R^2 = 0**: The model doesn’ t account for any variation in the response data around its mean. The predictions are as good as simply using the mean of the actual data.

•**0 < R^2 < 1**: The model explains a portion of the variability, with higher values indicating a better fit.

•**R^2 < 0**: This can occur if the model is worse than a horizontal line (mean of actual values), which typically indicates an incorrect model.