مدخل إلى تعلم الآلة

د. فارس بن صالح القنيعير

www.qunaieer.com
Twitter: @qunaieer

تنظيم

{برمج}

طريقة المحاضرة

- شرح المصطلحات والمفاهيم الأساسية
- المصطلحات وأغلب شرائح العرض بالإنجليزي والشرح بالعربي
- شرح مفصل للأساسيات
- إشارات سريعة للخوارزميات الشهيرة
- أمثلة عملية بسيطة
- خطة تعلم مقترحة

ما هو تعلم الآلة؟

- هو علم يمكن الحاسب التعلم من نفسه بدلاً من برمجته بالتفصيل
- اختزال جوهر البيانات عن طريق بناء نماذج (models)، واتخاذ القرارات والتوقعات المستقبلية بناءً عليها

Mohammed Al-Amoudi

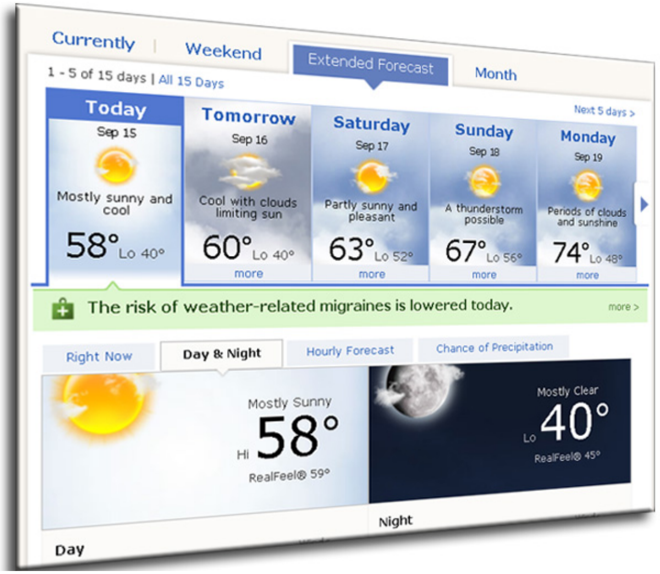
ويصنف تعلم الآلة كفرع رئيسي من فروع الذكاء الصناعي والذي يسمح للكمبيوتر بالتصرف بدون أن يكون مبرمجاً مسبقاً للقيام بذلك والاستجابة للأحداث بشكل ذاتي دون أن يتم تلقينه ذلك من قبل المبرمج.


Machine
Learning

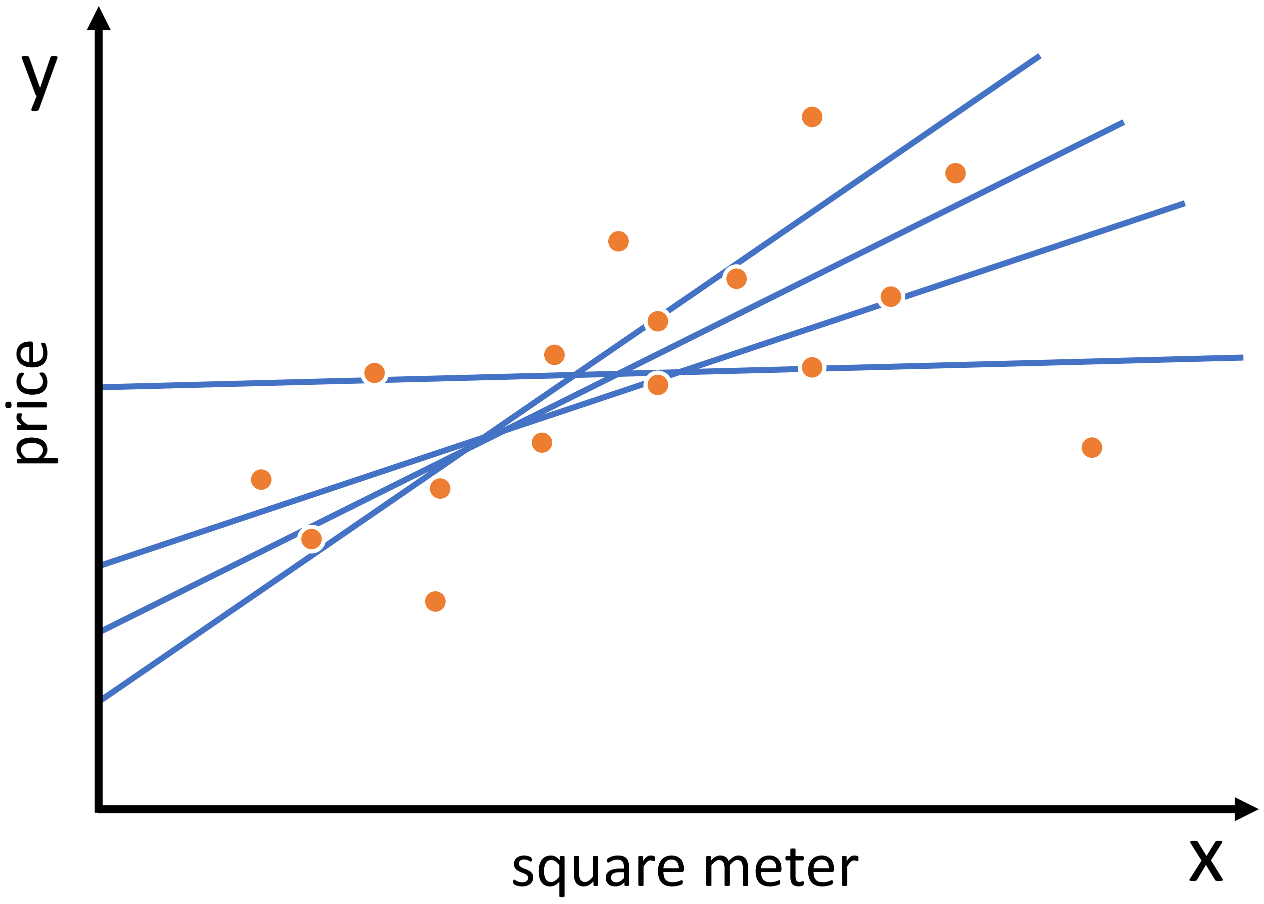
Regression

- **Regression analysis** is a statistical process for estimating the relationships among variables
- Used to predict continuous outcomes

Regression Examples



Linear Regression



Line Equation

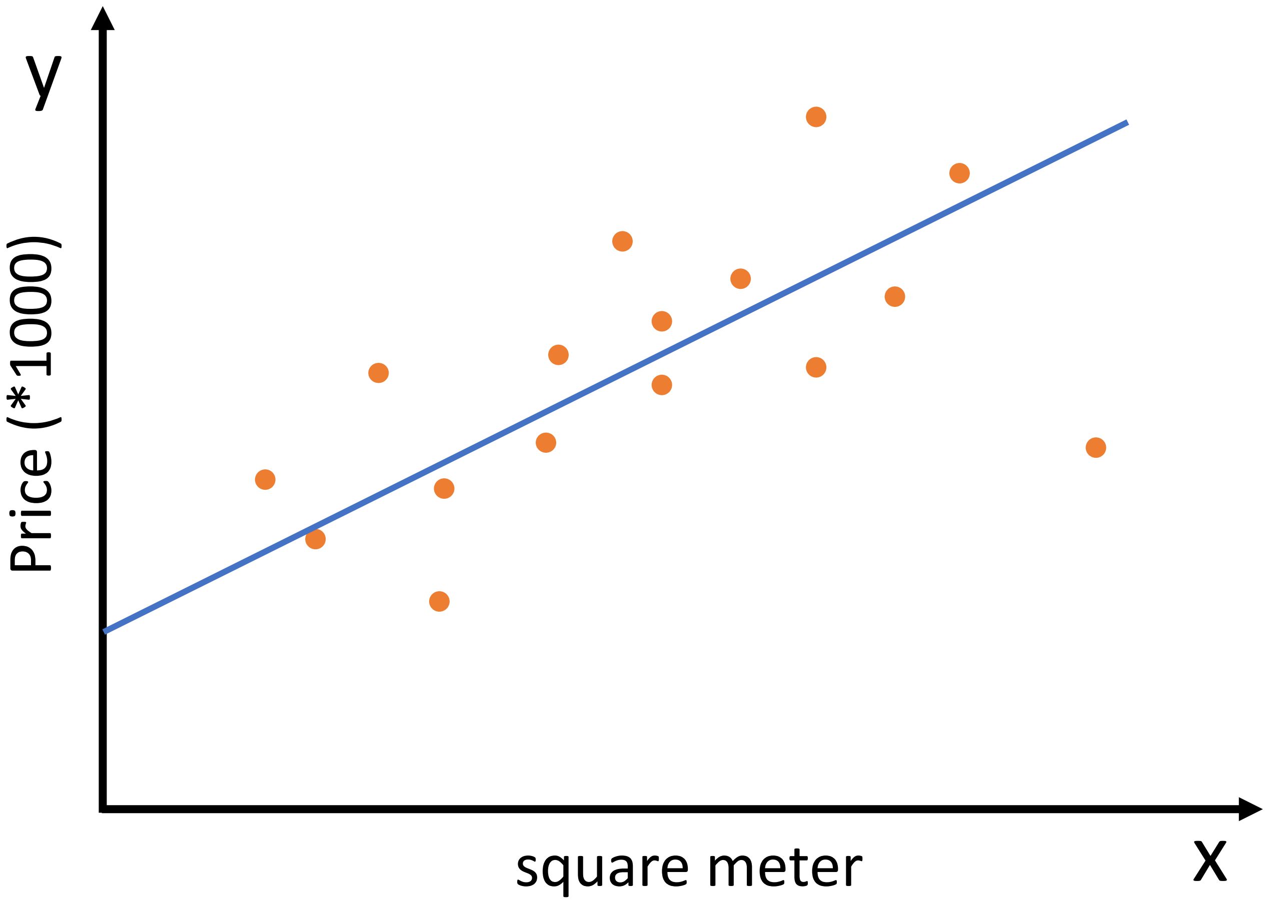
$$y = b + ax$$

intercept slope

$$\hat{y} = w_0 + w_1 x$$

Model/hypothesis

Linear Regression



$$\hat{y} = w_0 + w_1 x$$

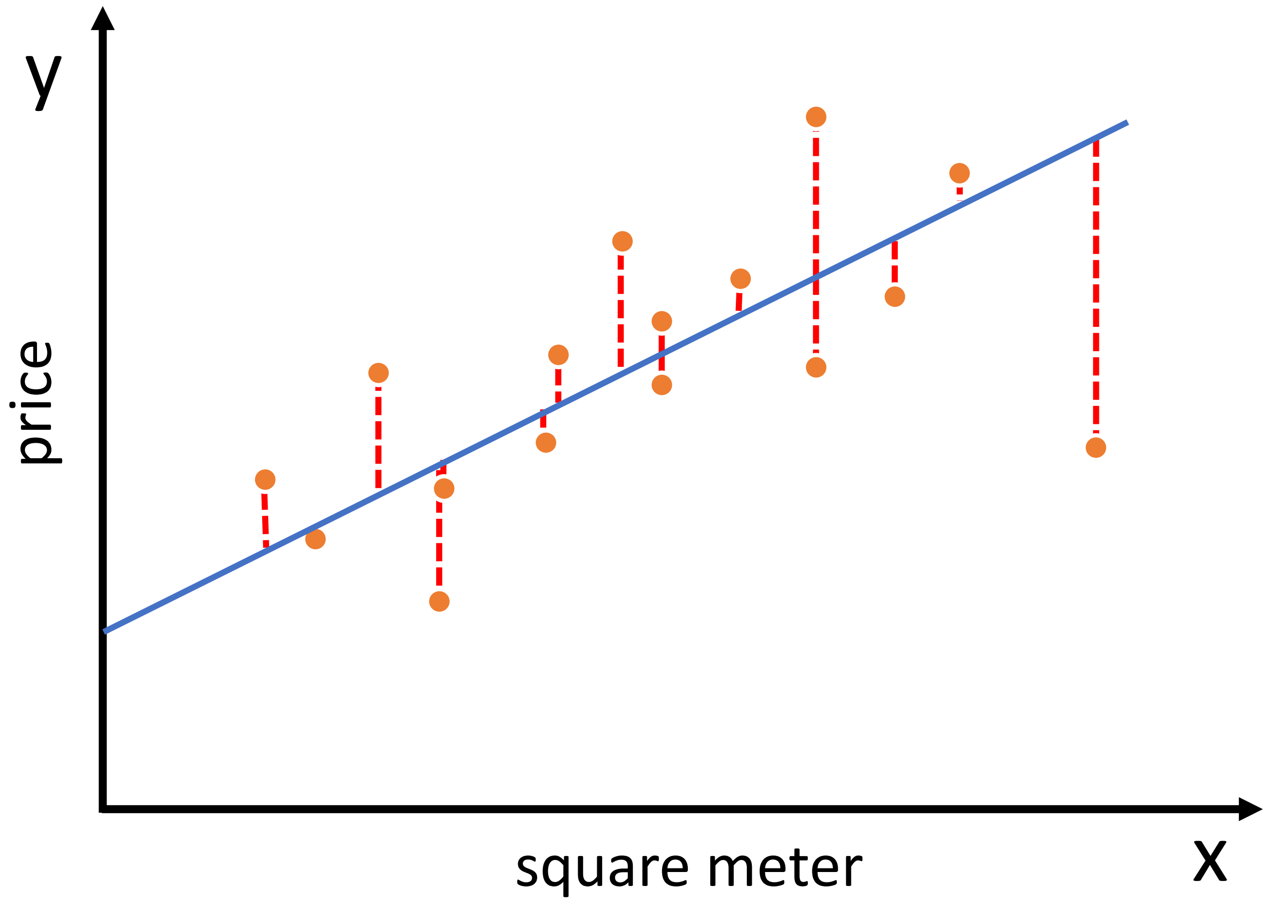
example

w_0=50, w_1=1.8,

x=500

$$\hat{y} = 950$$

Linear Regression



How to quantify error?

Linear Regression

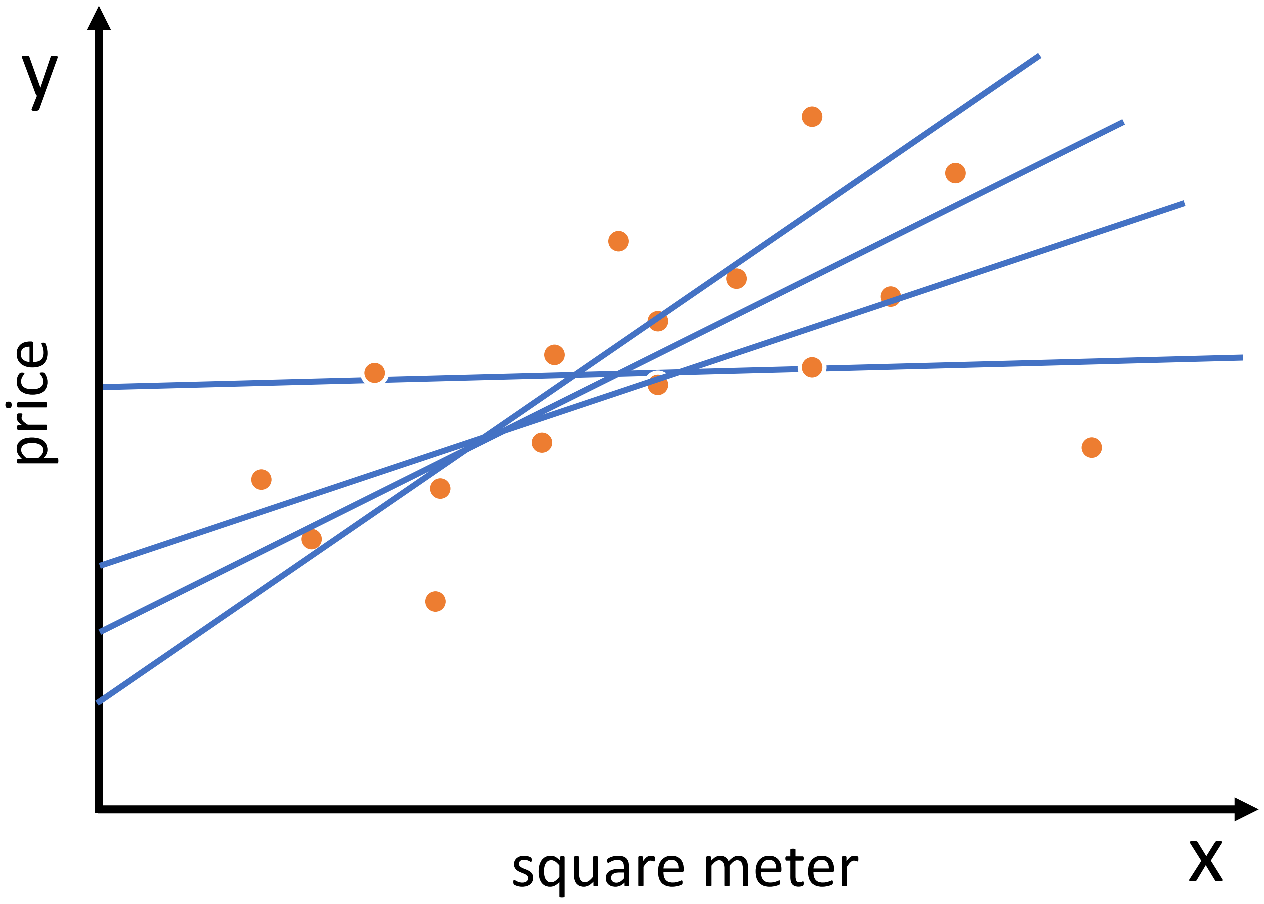
Residual Sum of Squares (RSS)

$$RSS(w_0, w_1) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$

Where $\hat{y}_i = w_0 + w_1 x_i$

$\boxed{\textbf{Cost function}}$

Linear Regression



How to choose best model?

Choose w_0 and w_1 that give lowest RSS value = Find The Best Line

Optimization

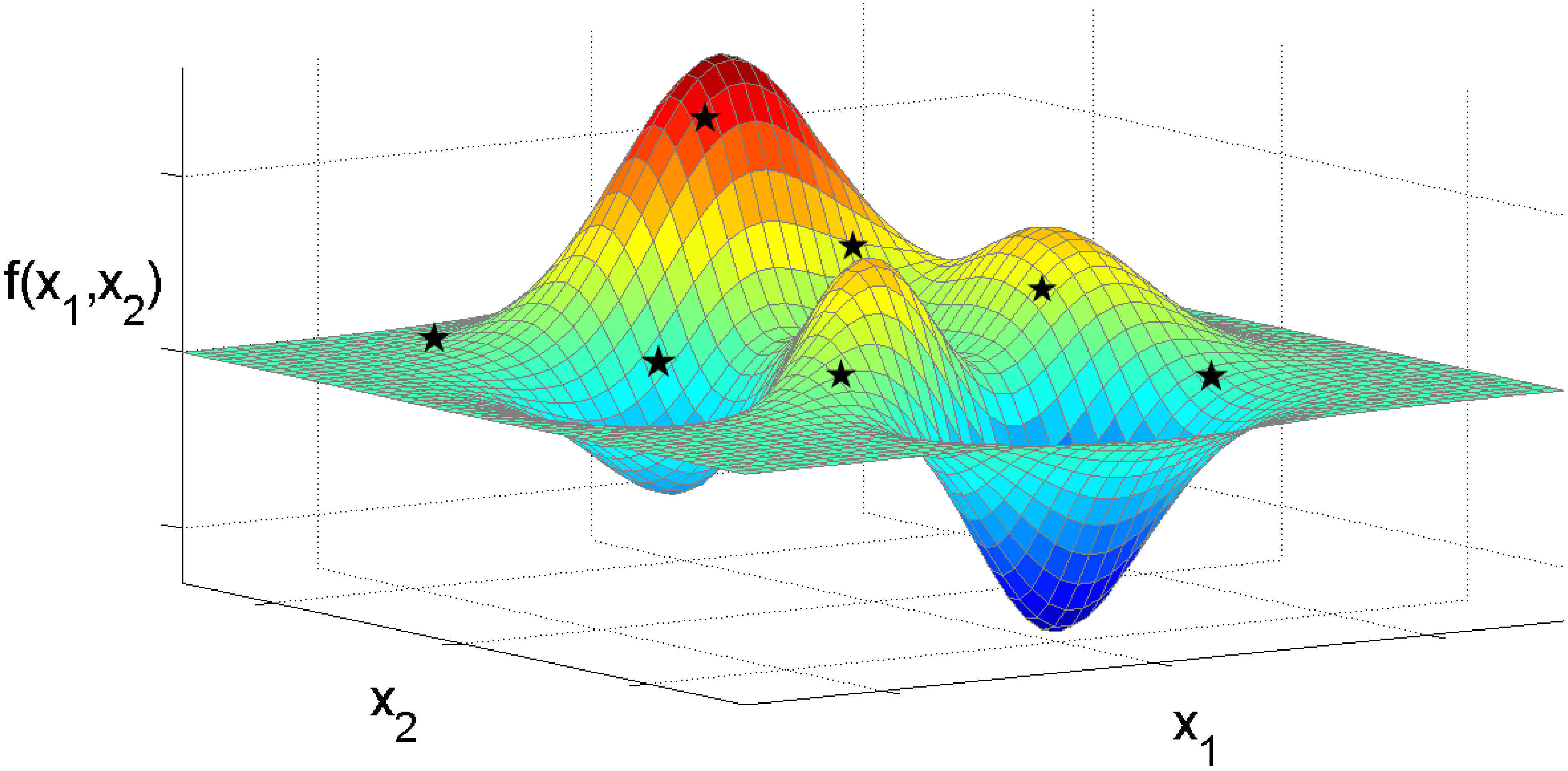
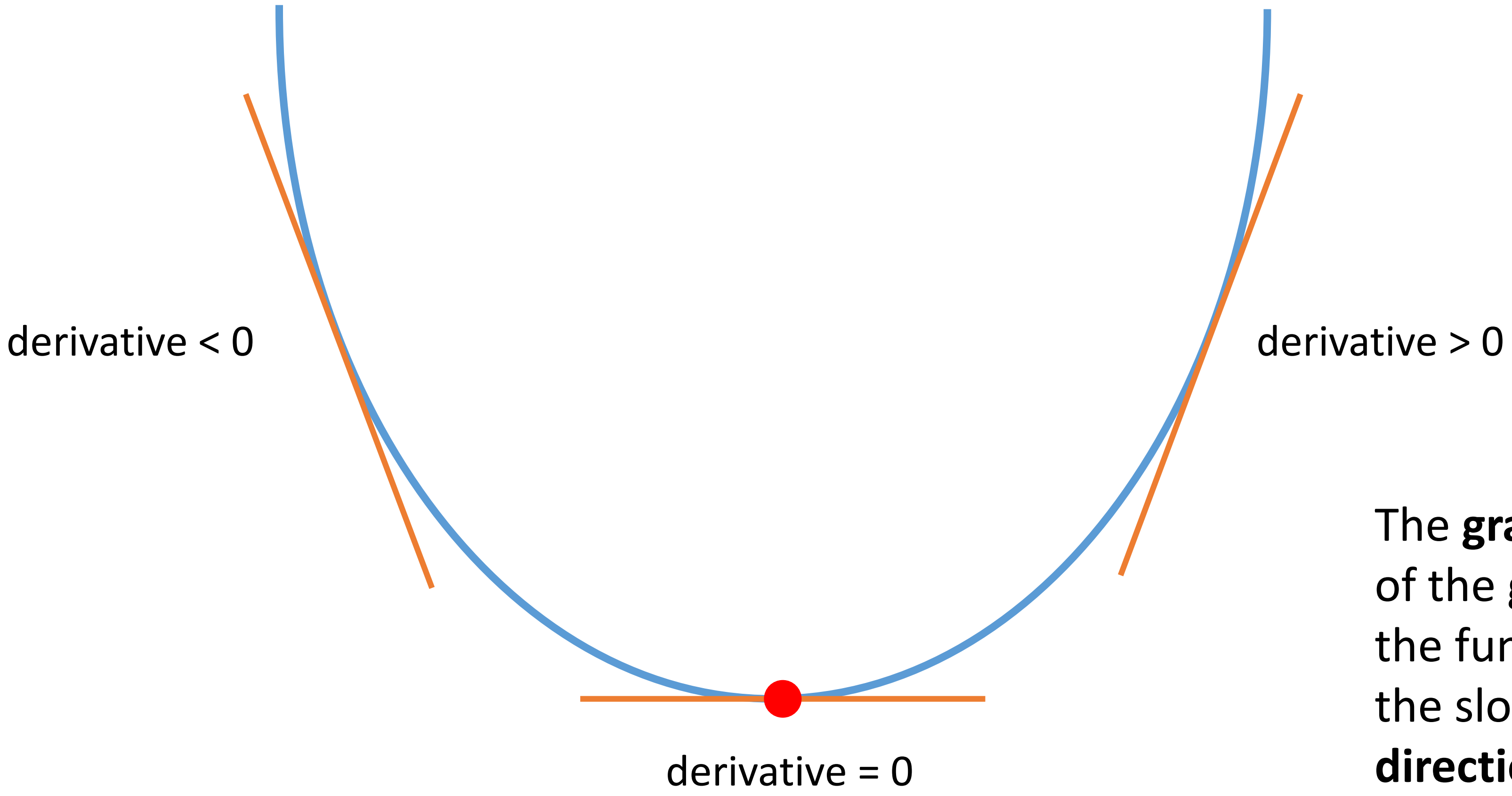


Image from https://ccse.lbl.gov/Research/Optimization/index.html

Optimization (convex)



The **gradient** points in the **direction** of the greatest rate of increase of the function, and its magnitude is the slope of the graph in that **direction**. - Wikipedia

Optimization (convex)

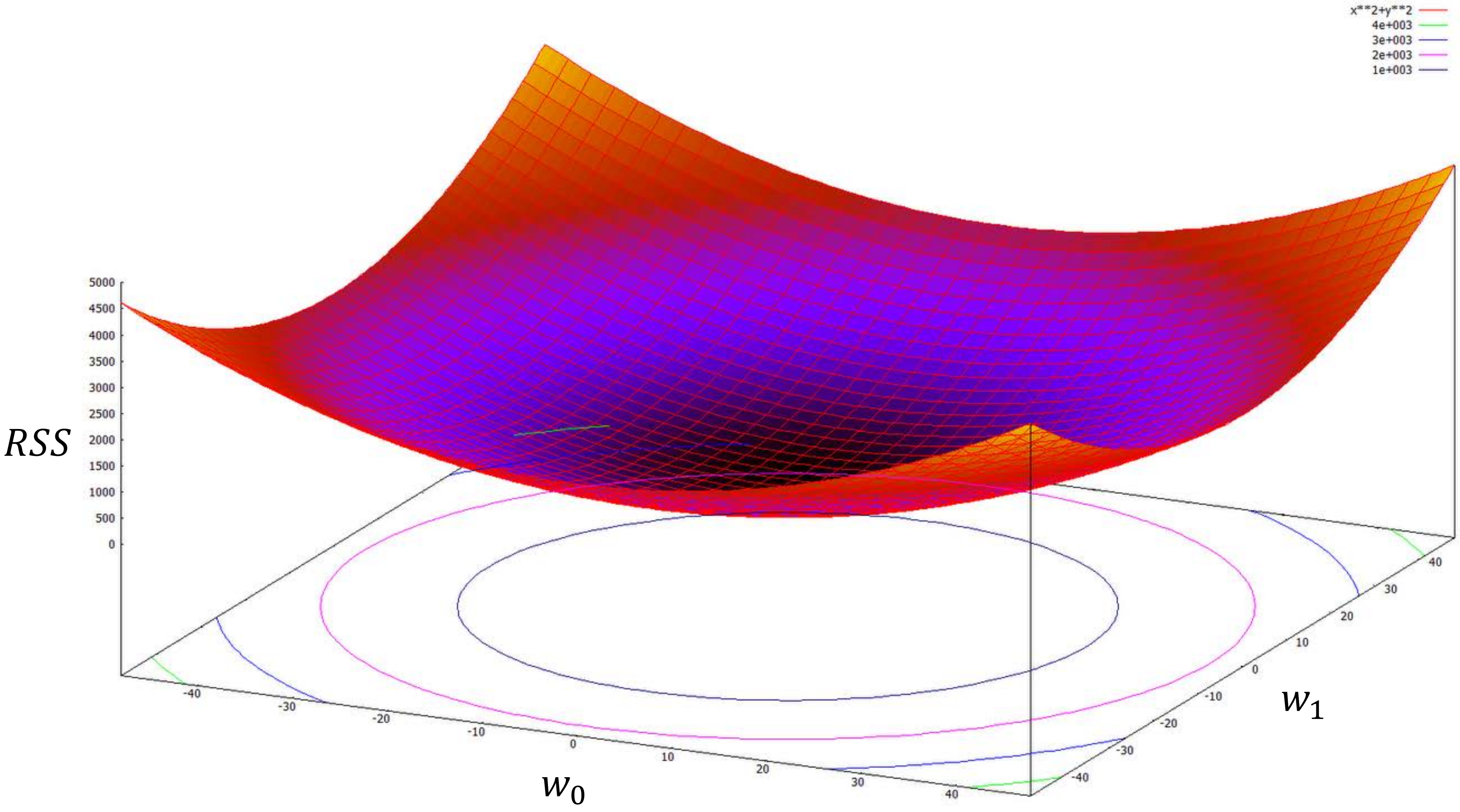


Image from http://codingwiththomas.blogspot.com/2012/09/particle-swarm-optimization.html

Optimization (convex)

- First, lets compute the gradient of our cost function

$$\nabla RSS(w_0, w_1) = \begin{bmatrix} -2\sum_{i=1}^{N}[\hat{y}_i - y_i] \\ -2\sum_{i=1}^{N}[\hat{y}_i - y_i]x_i \end{bmatrix}$$

Where $\hat{y}_i = w_0 + w_1 x_i$

- To find best lines, there are two ways:
  - Analytical (normal equation)
  - Iterative (gradient descent)

Gradient Descent

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^{t} - \eta \nabla RSS$$

$$\begin{bmatrix} w_0^{t+1} \\ w_1^{t+1} \end{bmatrix} = \begin{bmatrix} w_0^{t} \\ w_1^{t} \end{bmatrix} - \eta \begin{bmatrix} -2\sum_{i=1}^{N}[\hat{y}_i - y_i] \\ -2\sum_{i=1}^{N}[\hat{y}_i - y_i]x_i \end{bmatrix}$$

Update the weights to minimize the cost function

η is the step size (important hyper-parameter)

Gradient Descent

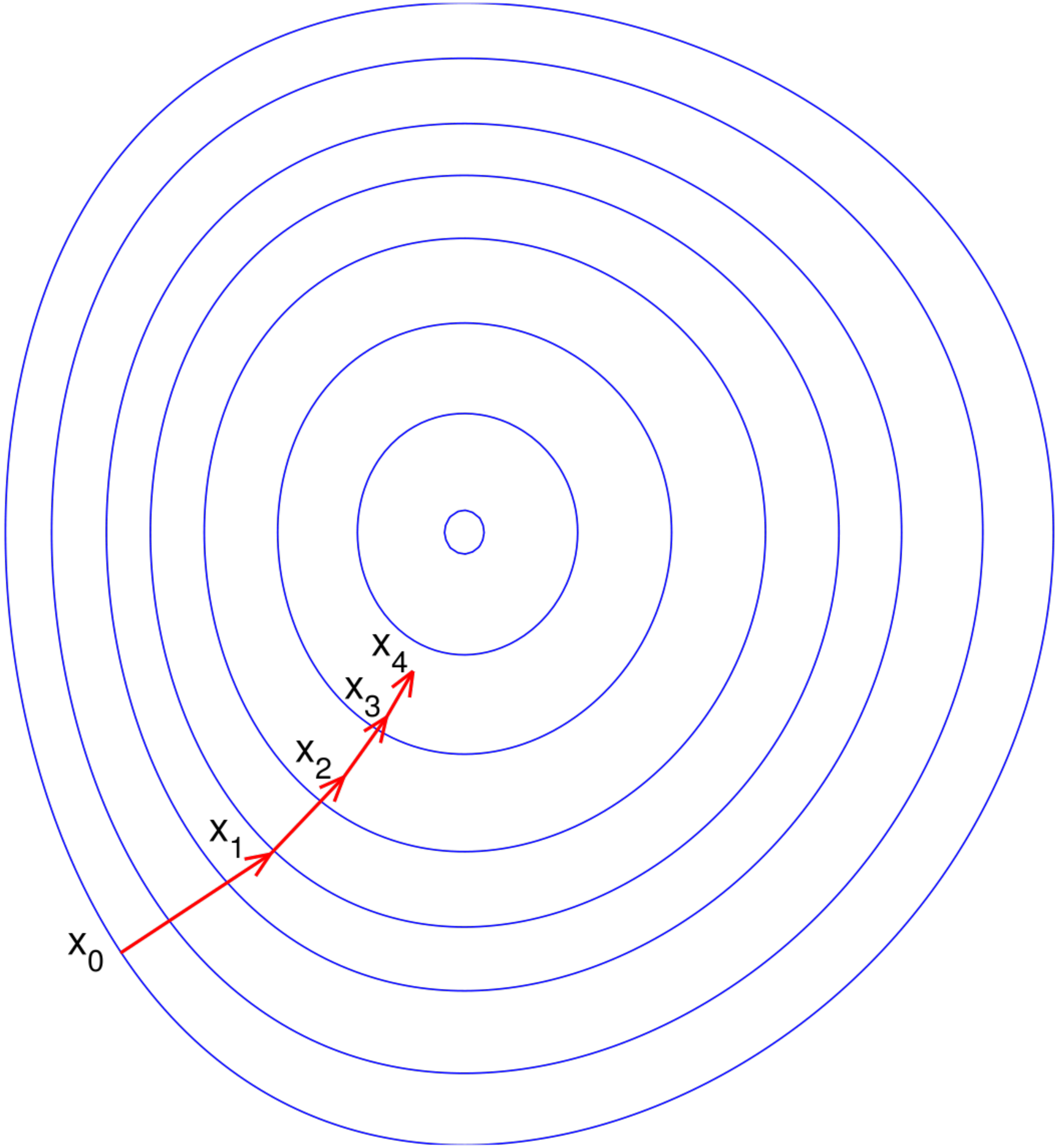


Image from wikimedia.org

Linear Regression: Algorithm

- **Objective:** $\min_{w_0, w_1} J(w_0, w_1)$, here $J(w_0, w_1) = RSS(w_0, w_1)$
- Initialize w_0, w_1, e.g. random numbers or zeros
- $for\ number\ of\ iterations\ or\ stopping\ criteria$:
  - *Compute the gradient:* ∇J
  - $W^{t+1} = W^t - \eta \nabla J$, where $\mathrm{W} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$

The Essence of Machine Learning

Model/hypothesis

$$\hat{y} = w_0 + w_1 x$$

Cost function

$$RSS(w_0, w_1) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

Optimization

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \nabla RSS$$

Linear Regression: Multiple features

- Example: for house pricing, in addition to size in **square meters**, we can use **city**, **location**, **number of rooms**, **number of bathrooms**, etc
- The model/hypothesis becomes

$$\hat{y} = w_0 + w_1 x_{\ 1} + w_2 x_2 + \cdots + w_n x_n$$

$$wehre\ n = number\ of\ features$$

Representation

- Vector representation of n features

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

$$\boldsymbol{x} = \begin{bmatrix} x_0 = 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \qquad \hat{y} = \begin{bmatrix} w_0 & w_1 & w_2 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x}$$

Representation

- matrix representation of m data samples and n features

$$\hat{y}^{(i)} = w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_n x_n^{(i)}$$

i is the i^{th} data sample

$$X = \begin{bmatrix} x_0^{(0)} & x_1^{(0)} & \cdots & x_n^{(0)} \\ x_0^{(1)} & x_1^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{(m)} & x_1^{(m)} & \dots & x_n^{(m)} \end{bmatrix} \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

Size: m x n

Size: n x 1

$$\boldsymbol{\hat{y}} = X\boldsymbol{w}$$

$$\begin{bmatrix} \hat{y}^{(0)} \\ \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(m)} \end{bmatrix} = \begin{bmatrix} x_0^{(0)} & x_1^{(0)} & \cdots & x_n^{(0)} \\ x_0^{(1)} & x_1^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{(m)} & x_1^{(m)} & \dots & x_n^{(m)} \end{bmatrix} \times \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

Analytical solution (normal equation)

$$\boldsymbol{w} = (X^T X)^{-1} X^T \boldsymbol{y}$$

Analytical vs. Gradient Descent

- **Gradient descent:** must select parameter η
- **Analytical solution:** no parameter selection

- **Gradient descent:** a lot of iterations
- **Analytical solution:** no need for iterations

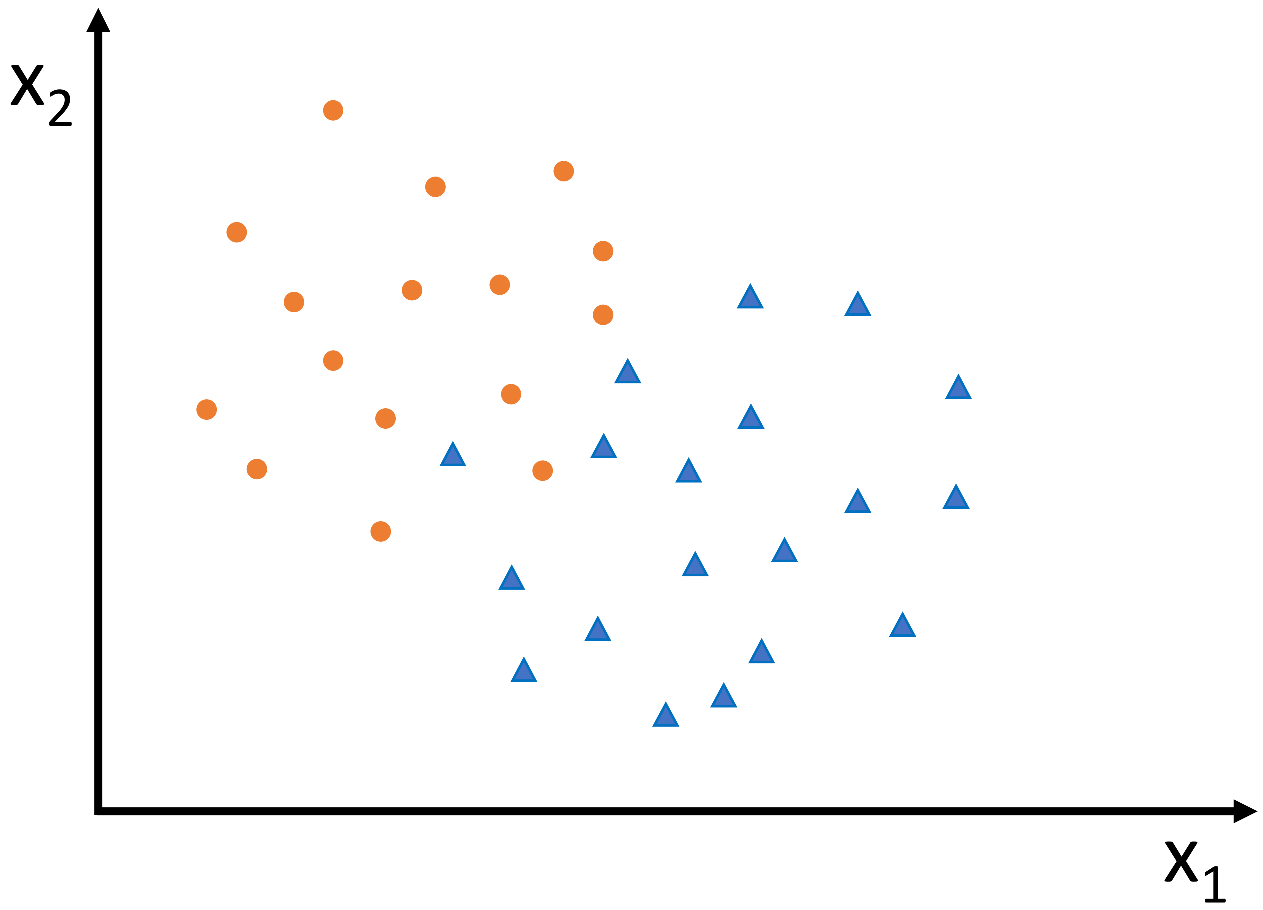
- **Gradient descent:** works with large number of features
- **Analytical solution:** slow with large number of features

Demo

- Matrices operations
- Simple linear regression implementation
- Scikit-learn library’s linear regression

Classification

classification



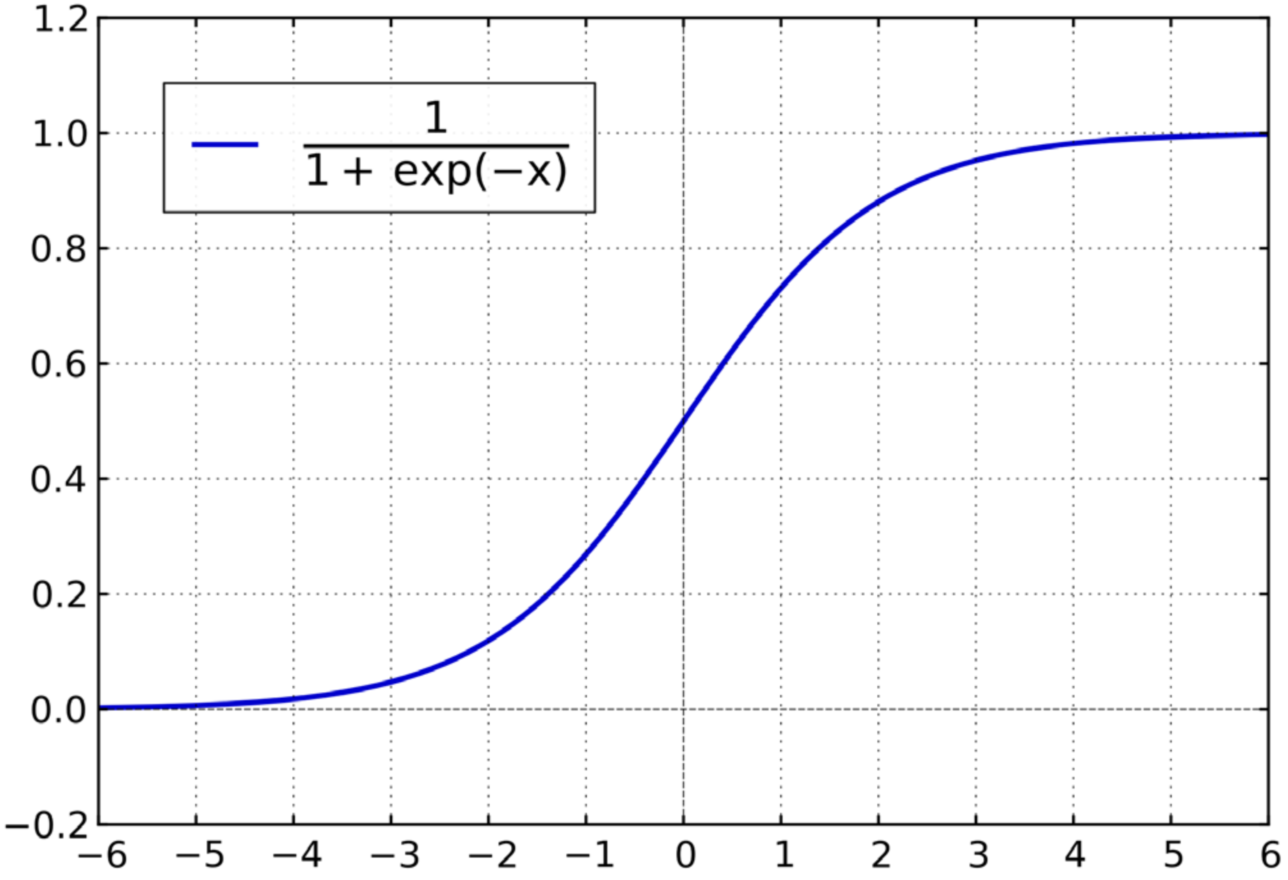
Classification Examples

Logistic Regression

- How to turn regression problem into classification one?
- y = 0 or 1
- Map values to [0 1] range

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

Sigmoid/Logistic Function



Logistic Regression

- Model (sigmoid\logistic function)

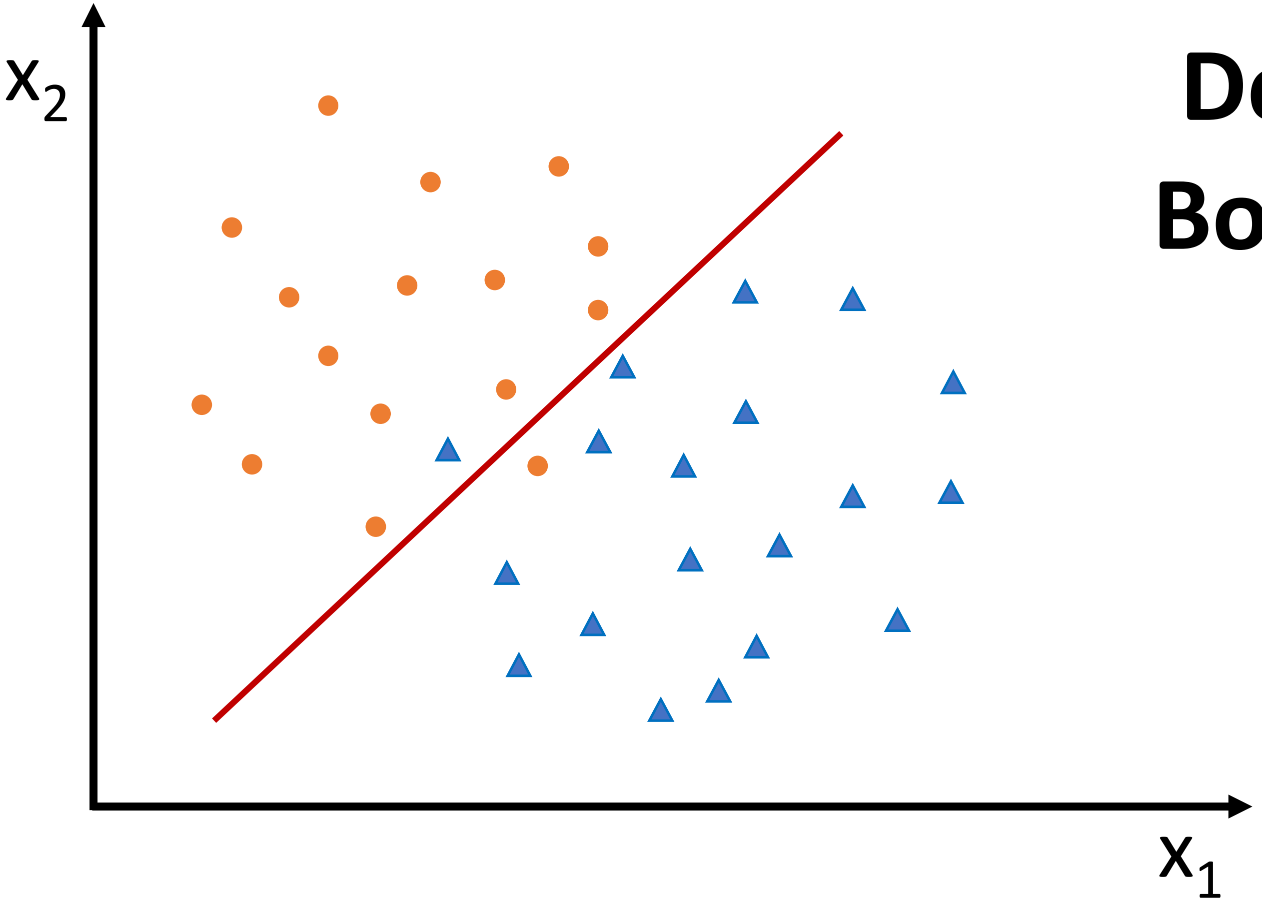
$$h_{\boldsymbol{w}}(\boldsymbol{x}) = g(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^T\boldsymbol{x}}}$$

- Interpretation (probability)

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = p(y = 1|\boldsymbol{x}; \boldsymbol{w})$$

$$if\ h_{\boldsymbol{w}}(\boldsymbol{x}) \geq 0.5 \Rightarrow y = 1$$
$$if\, h_{\boldsymbol{w}}(\boldsymbol{x}) < 0.5 \Rightarrow y = 0$$

Logistic Regression



Decision Boundary

Logistic Regression

- Cost function

$$J(\boldsymbol{w}) = \; y\log(h_{\boldsymbol{w}}(\boldsymbol{x})) + (1-y)\log(1-h_{\boldsymbol{w}}(\boldsymbol{x}))$$

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = g(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}}}$$

Logistic Regression

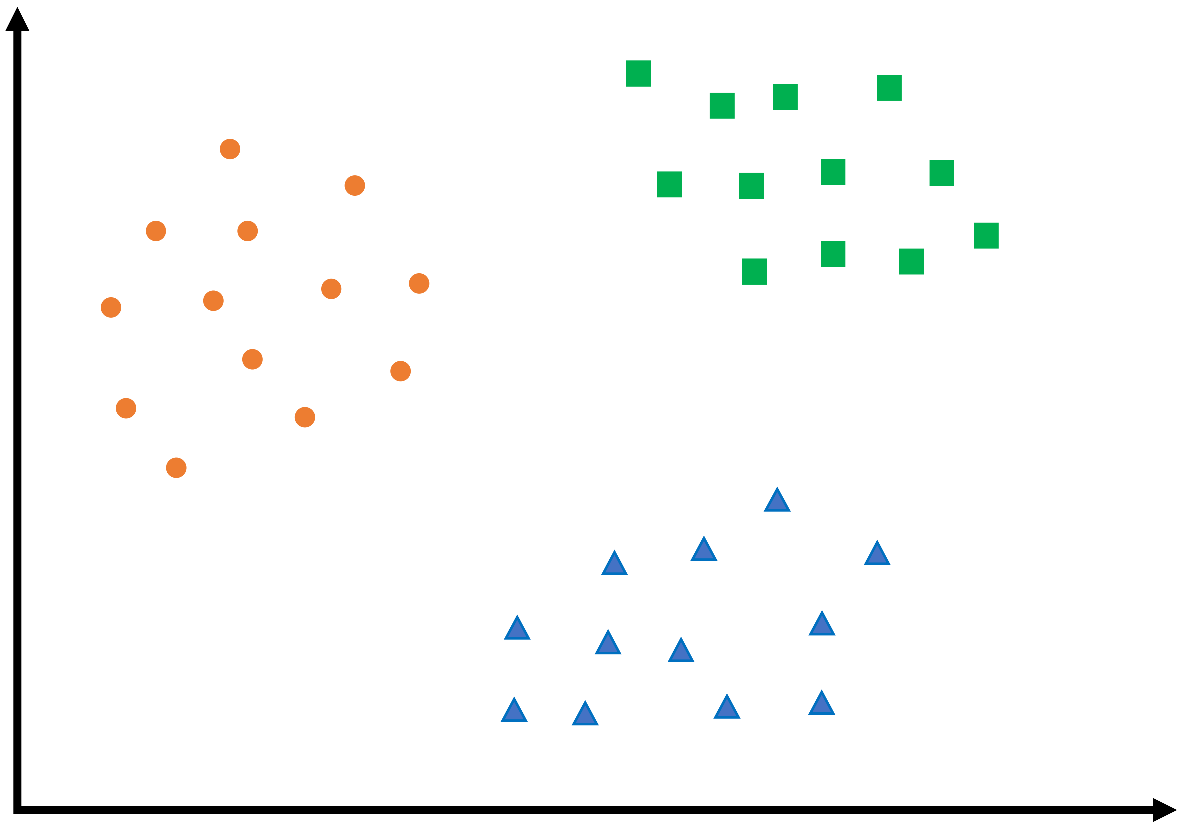
- **Optimization:** Gradient Descent
- Exactly like linear regression
- Find best **w** parameters that minimize the cost function

Logistic Regression: Algorithm

- **Objective:** $\min\limits_{w_0, w_1} J(w_0, w_1)$, here $J(w_0, w_1)$ is the logistic regression cost function
- Initialize w_0, w_1, e.g. random numbers or zeros
- $for\ number\ of\ iterations\ or\ stopping\ criteria$:
  - *Compute the gradient:* ∇J *(not discussed here)*
  - $\boldsymbol{w}^{t+1} = \boldsymbol{w}^{t} - \eta \nabla J$, where $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$

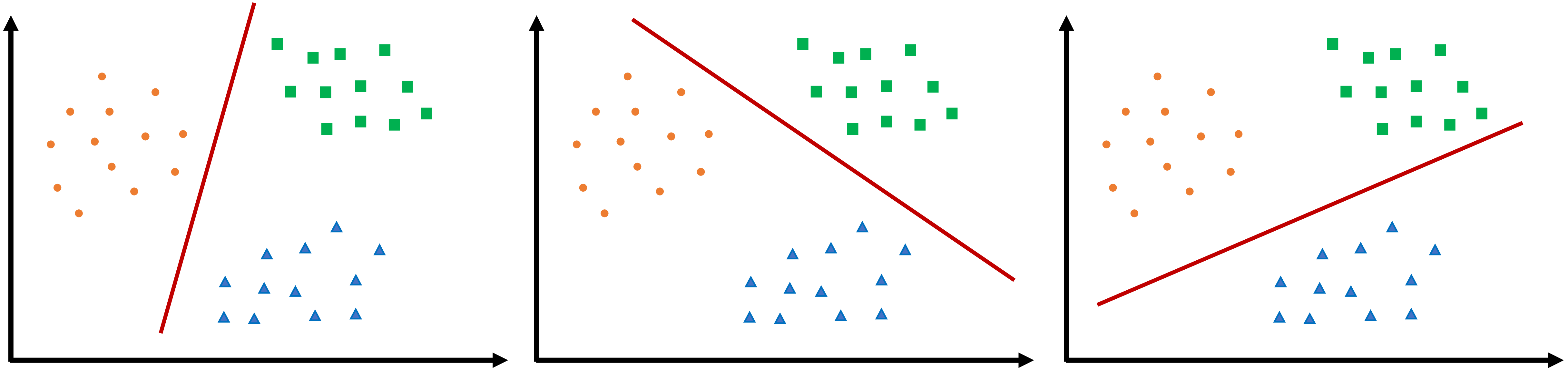
Logistic Regression

- Multi-class classification

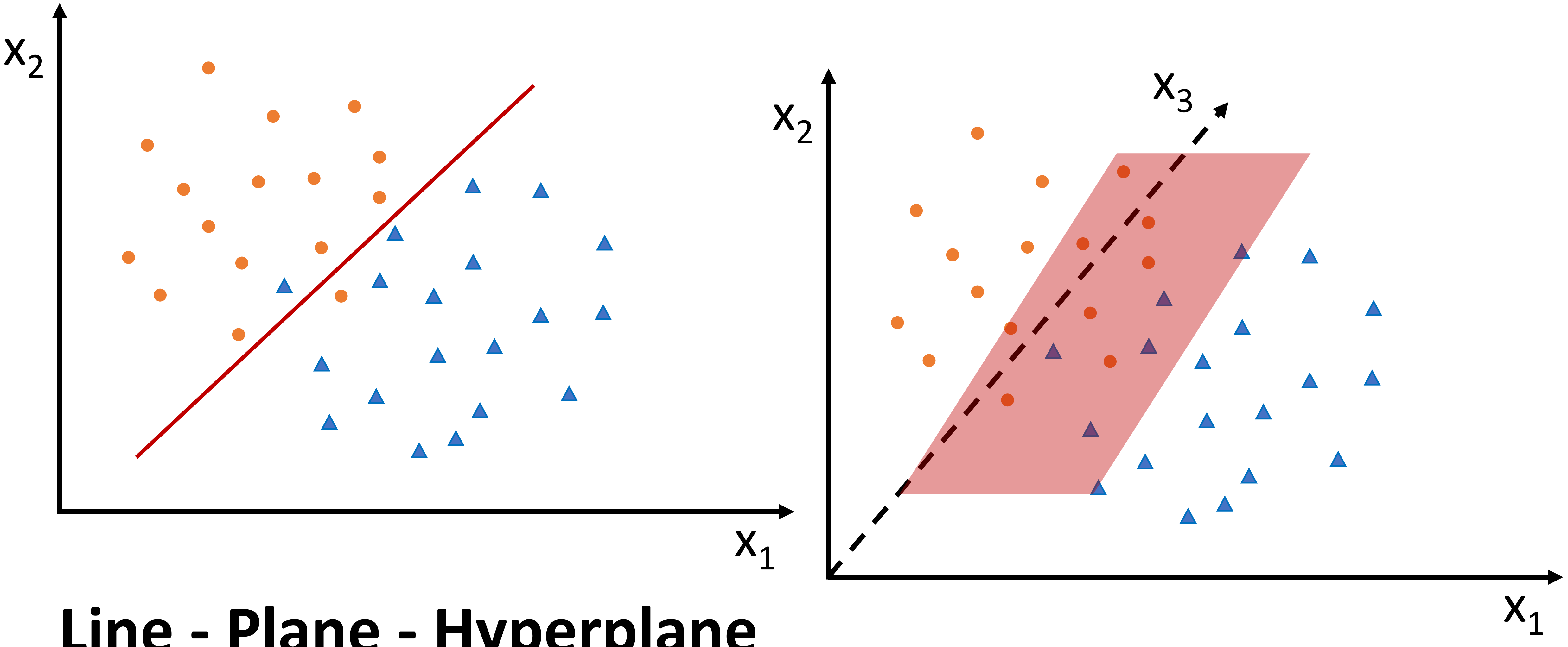


Logistic Regression

One-vs-All



Logistic Regression: Multiple Features



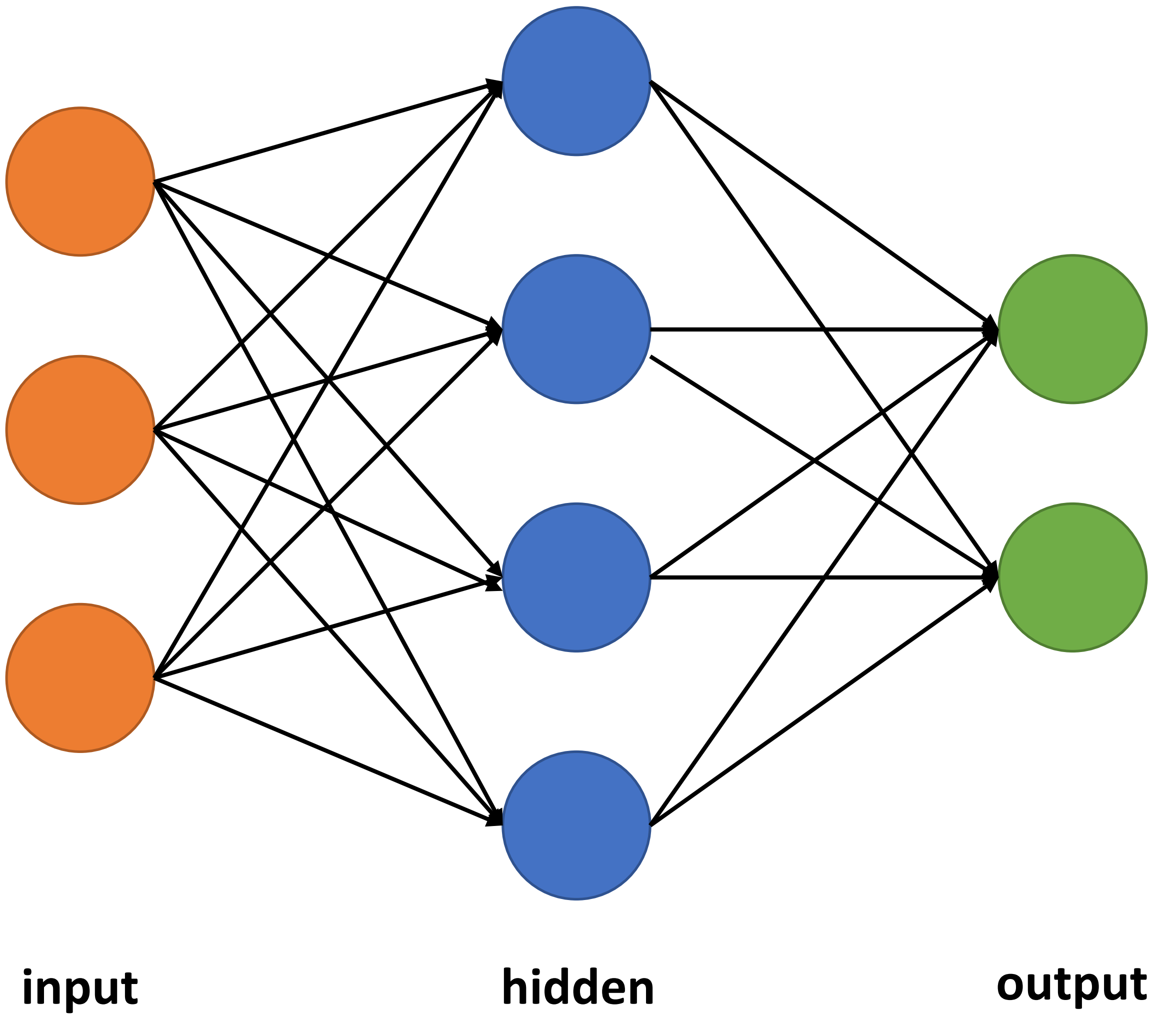
Line - Plane - Hyperplane

Demo

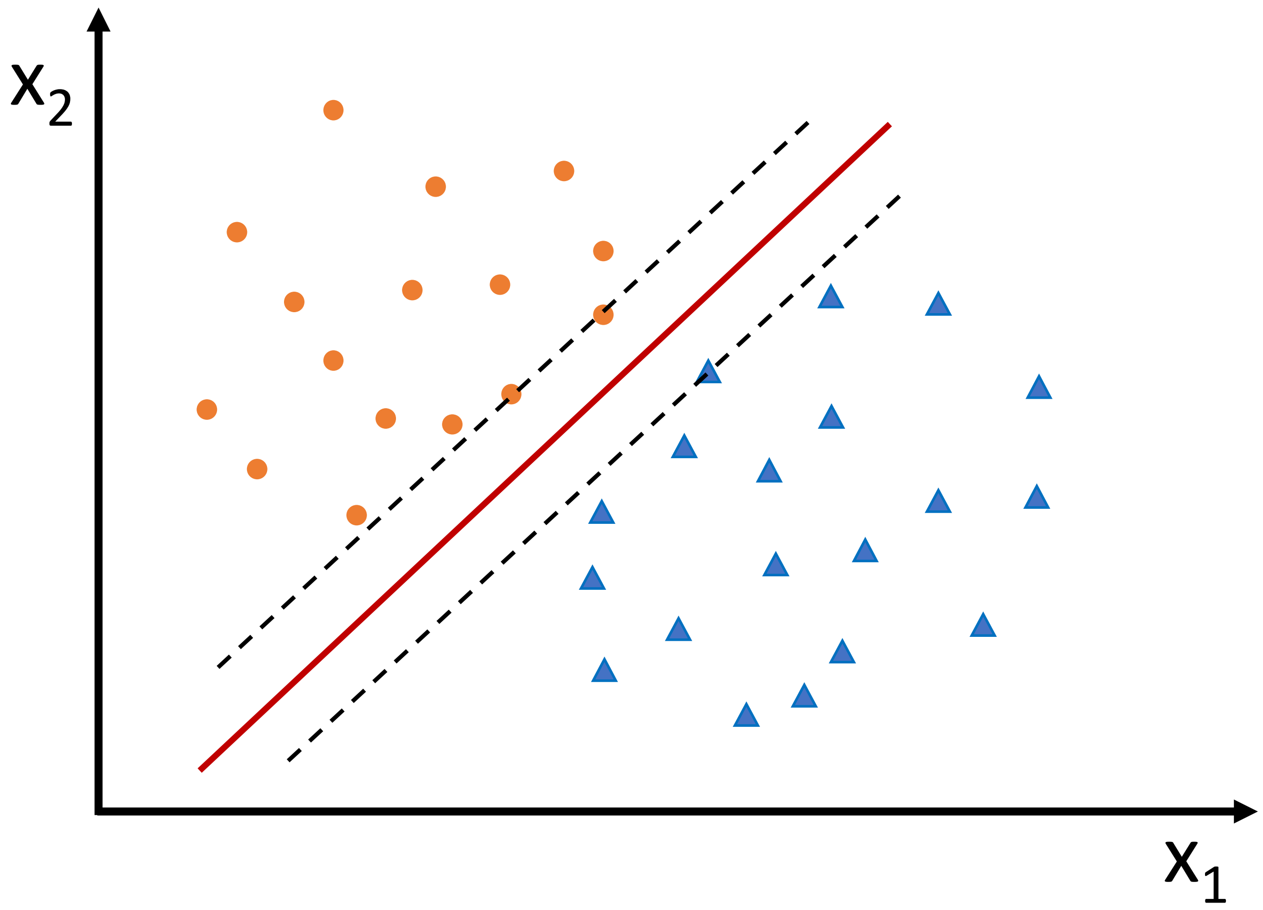
- Scikit-learn library’s logistic regression

Other Classification Algorithms

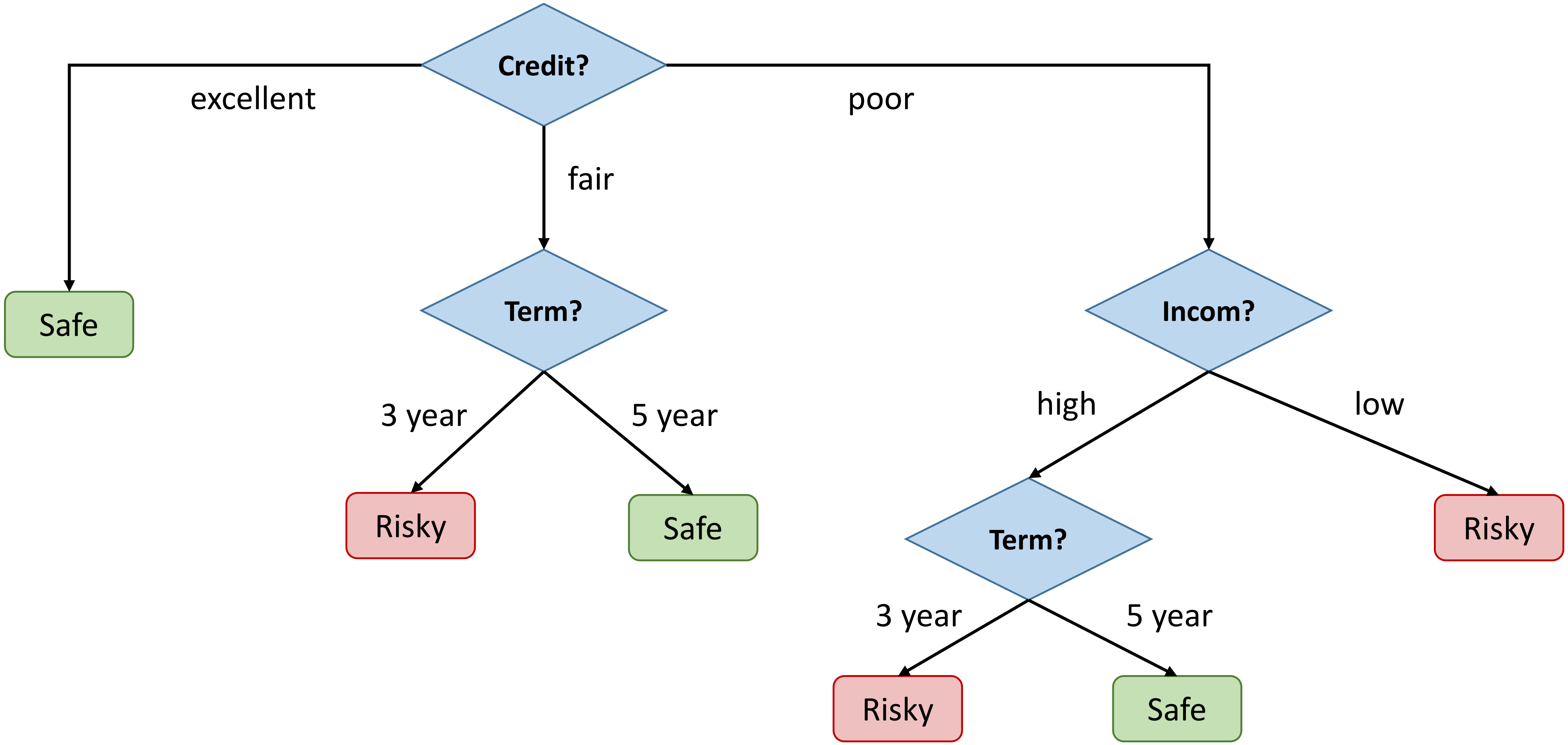
Neural Networks



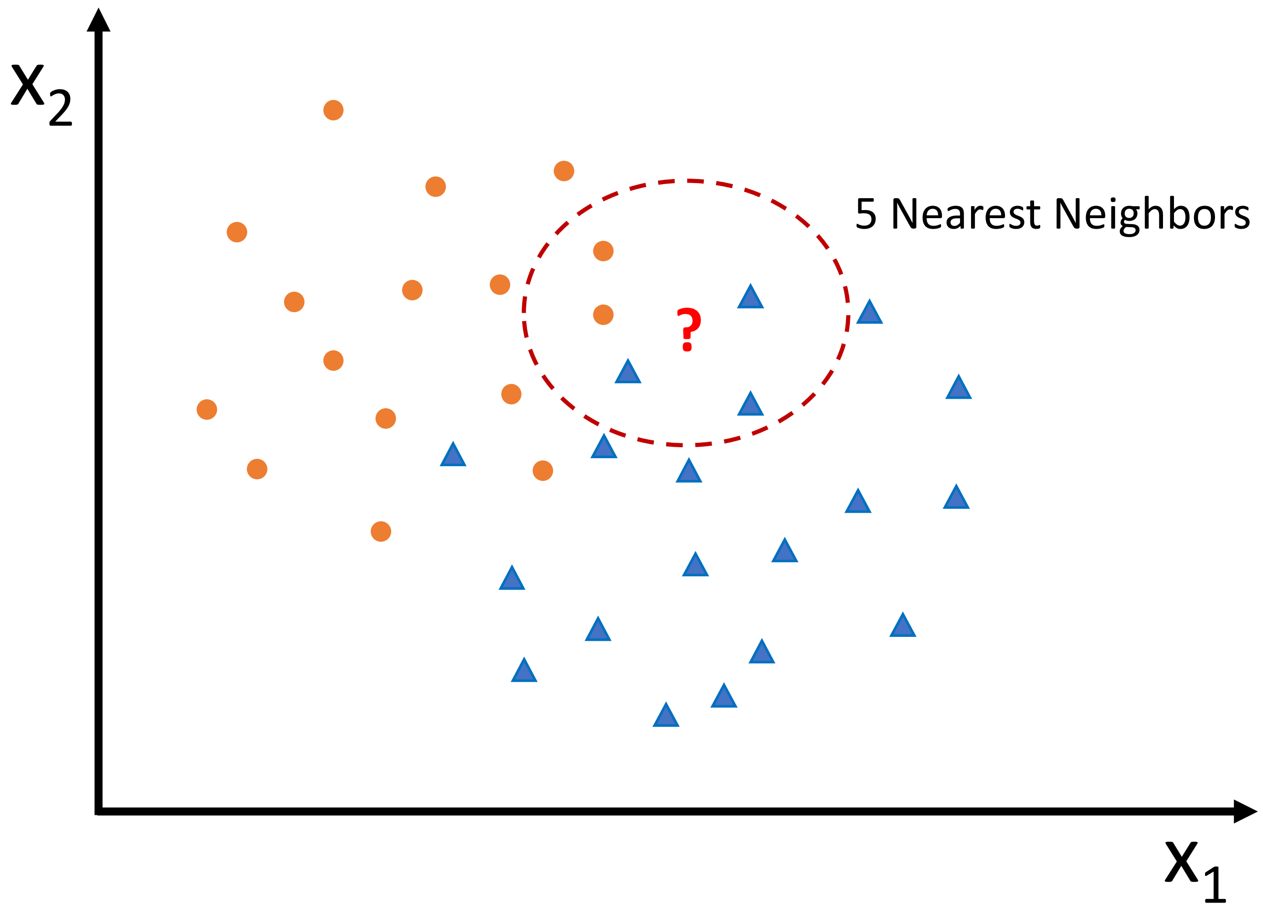
Support Victor Machines (SVM)



Decision Trees



K Nearest Neighbors (KNN)

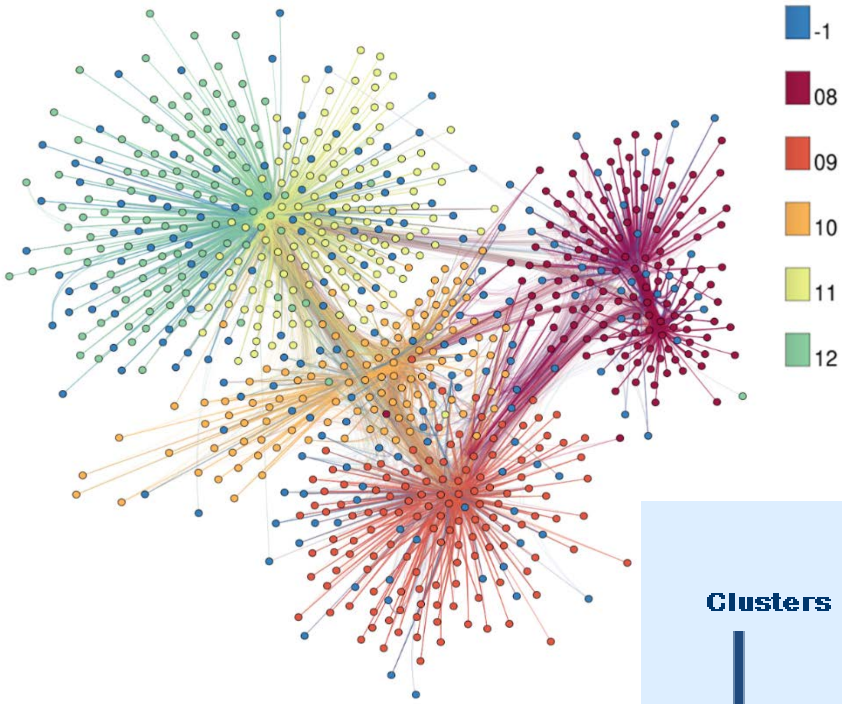


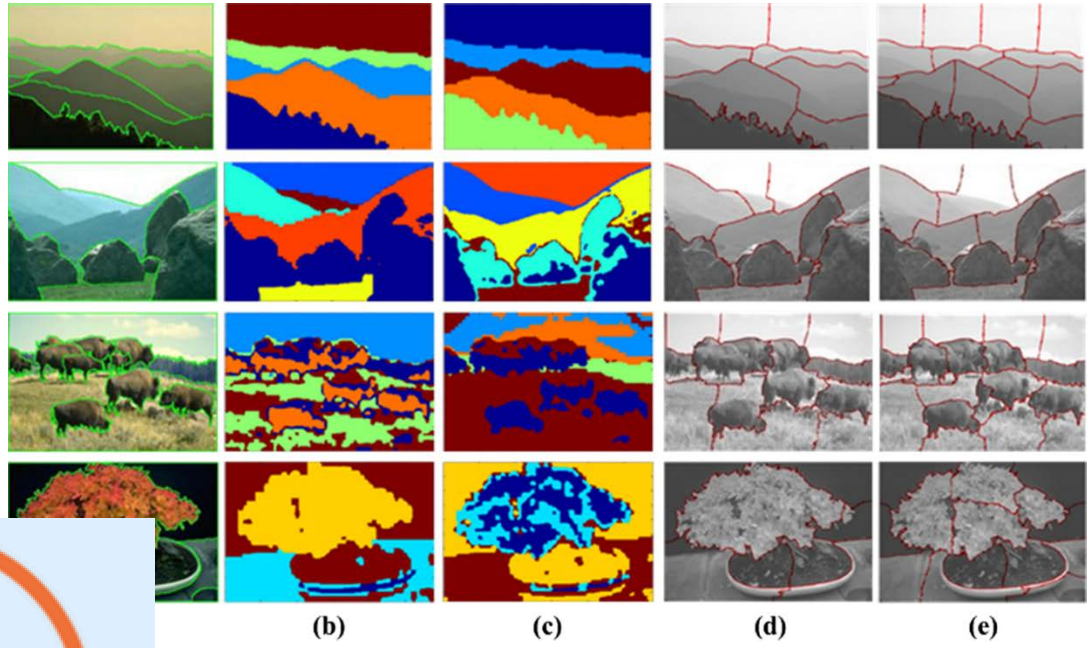
Clustering

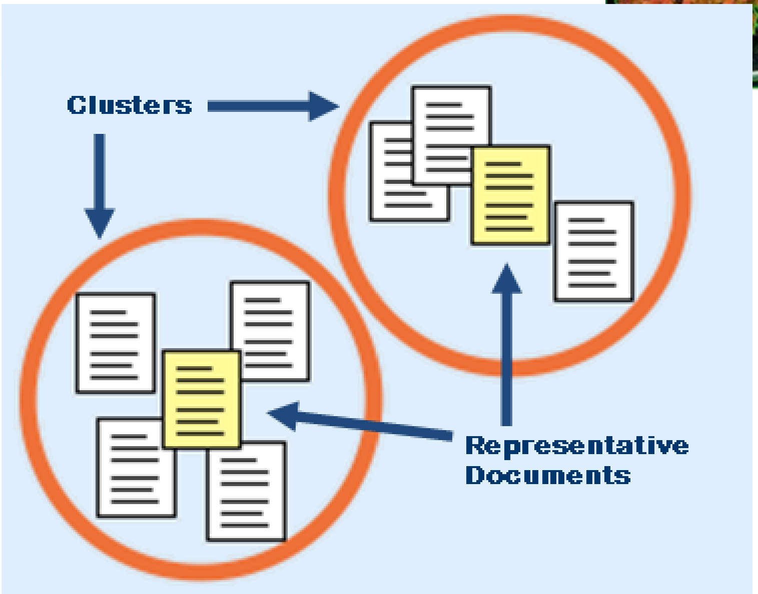
Clustering

- Unsupervised learning
- Group similar items into clusters
- K-Mean algorithm

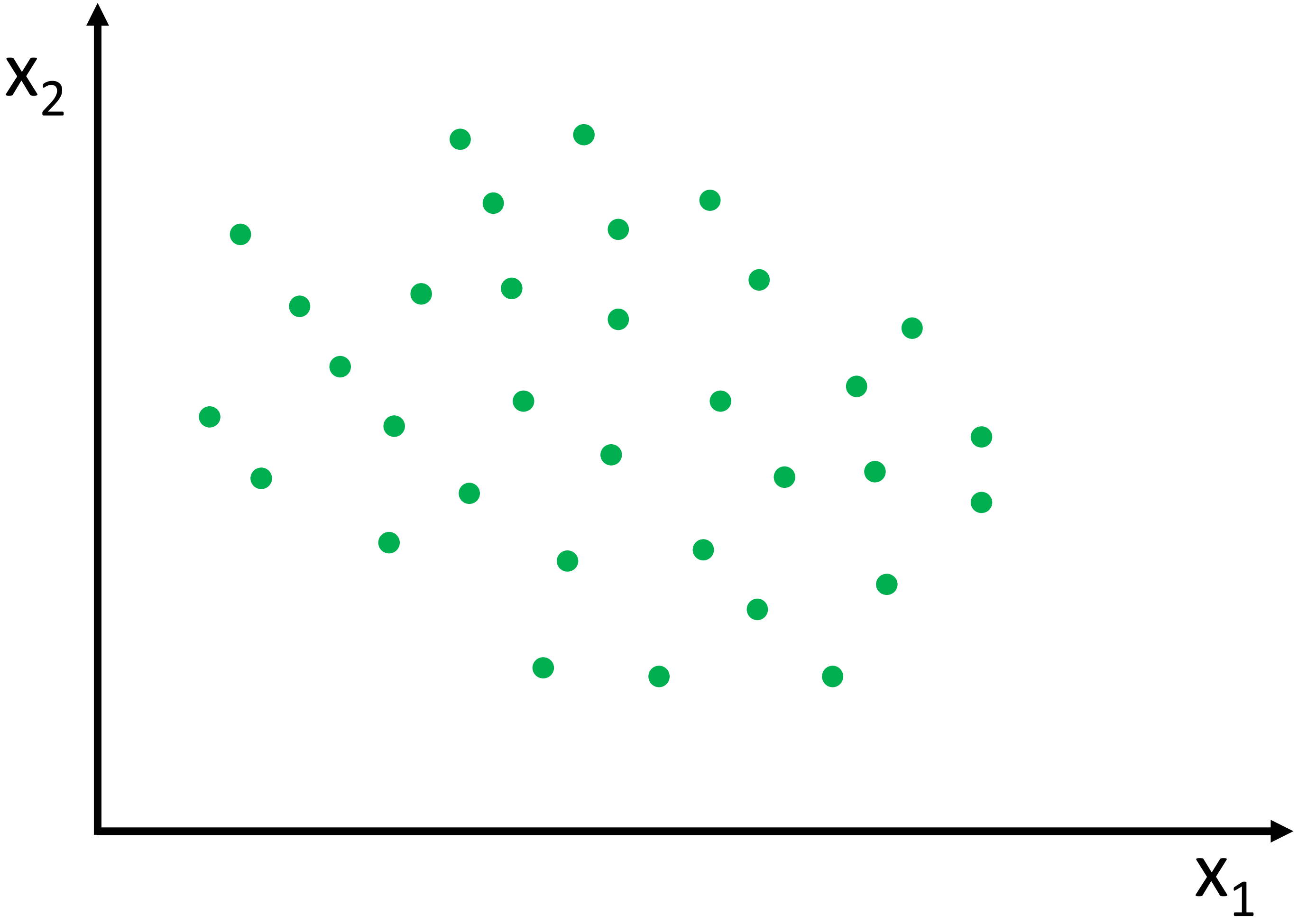
Clustering Examples



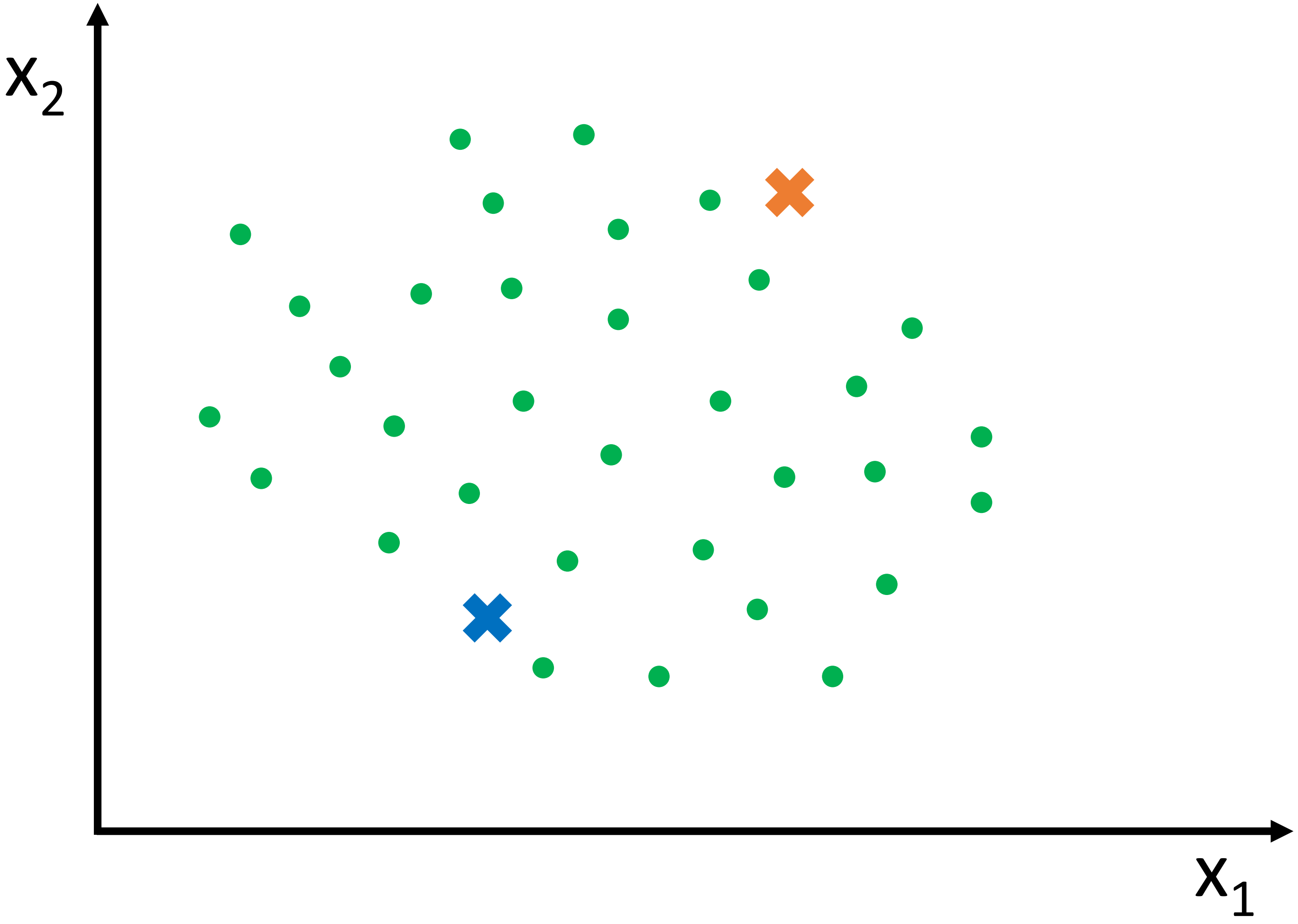




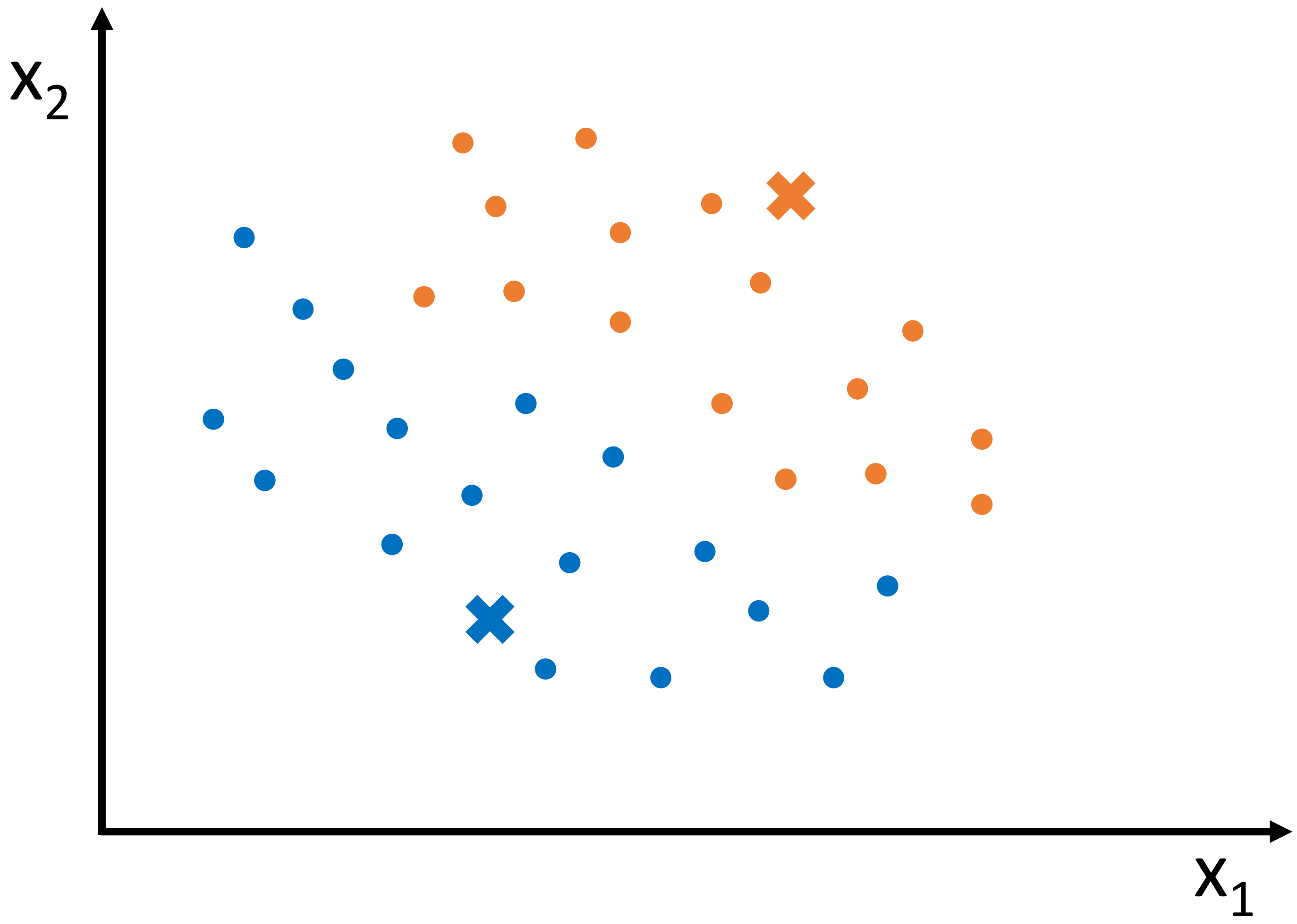
K-Mean



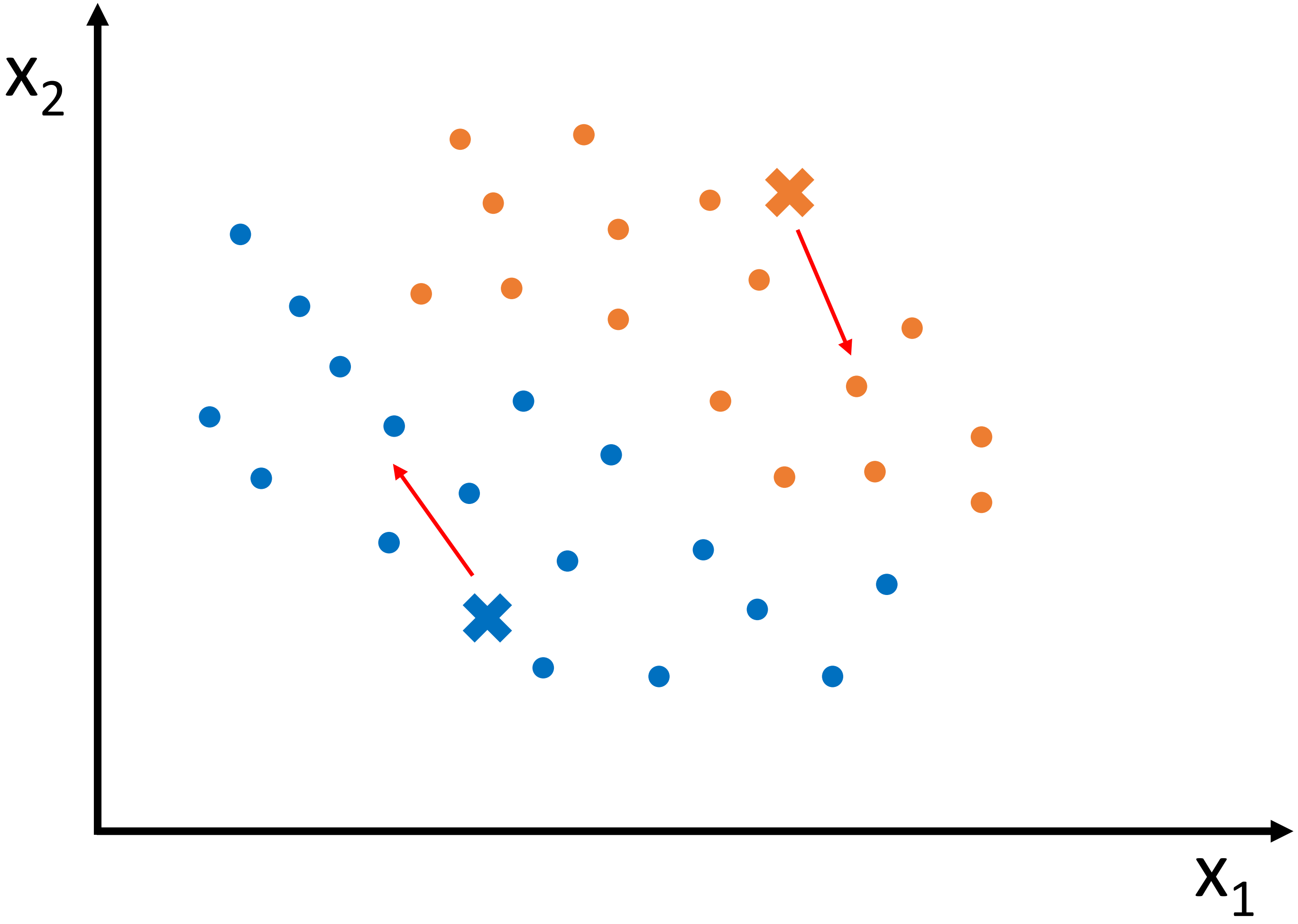
K-Mean



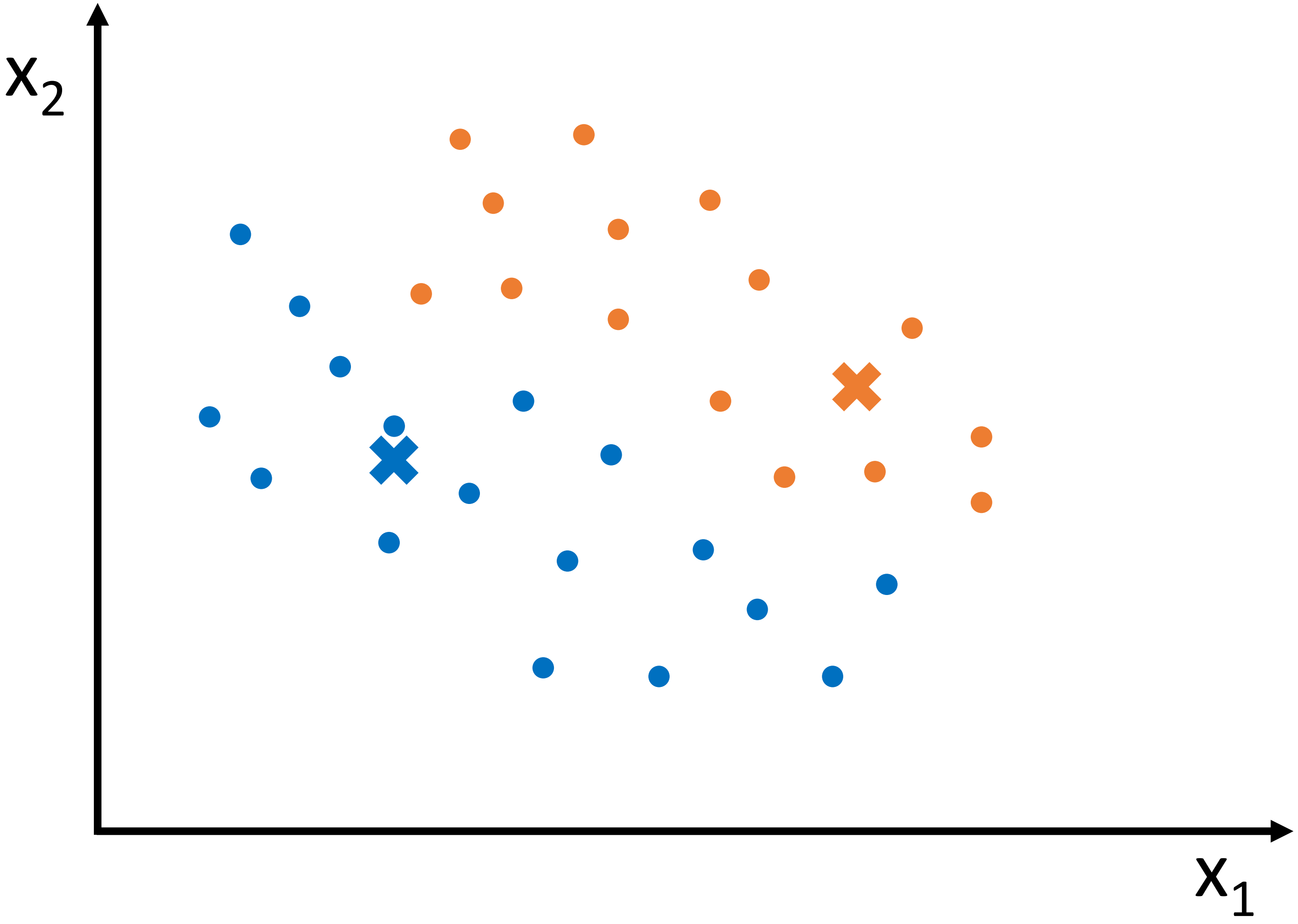
K-Mean



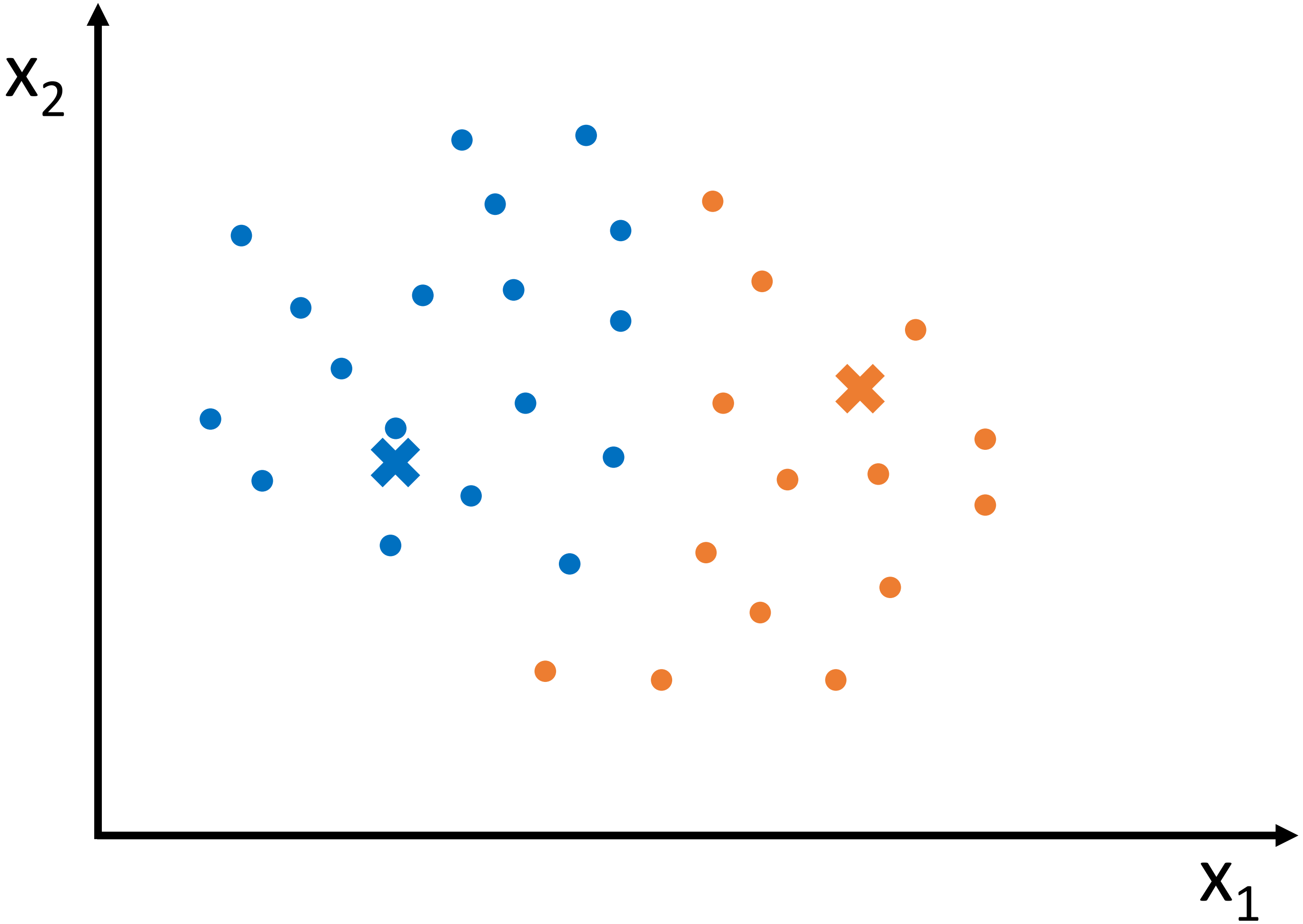
K-Mean



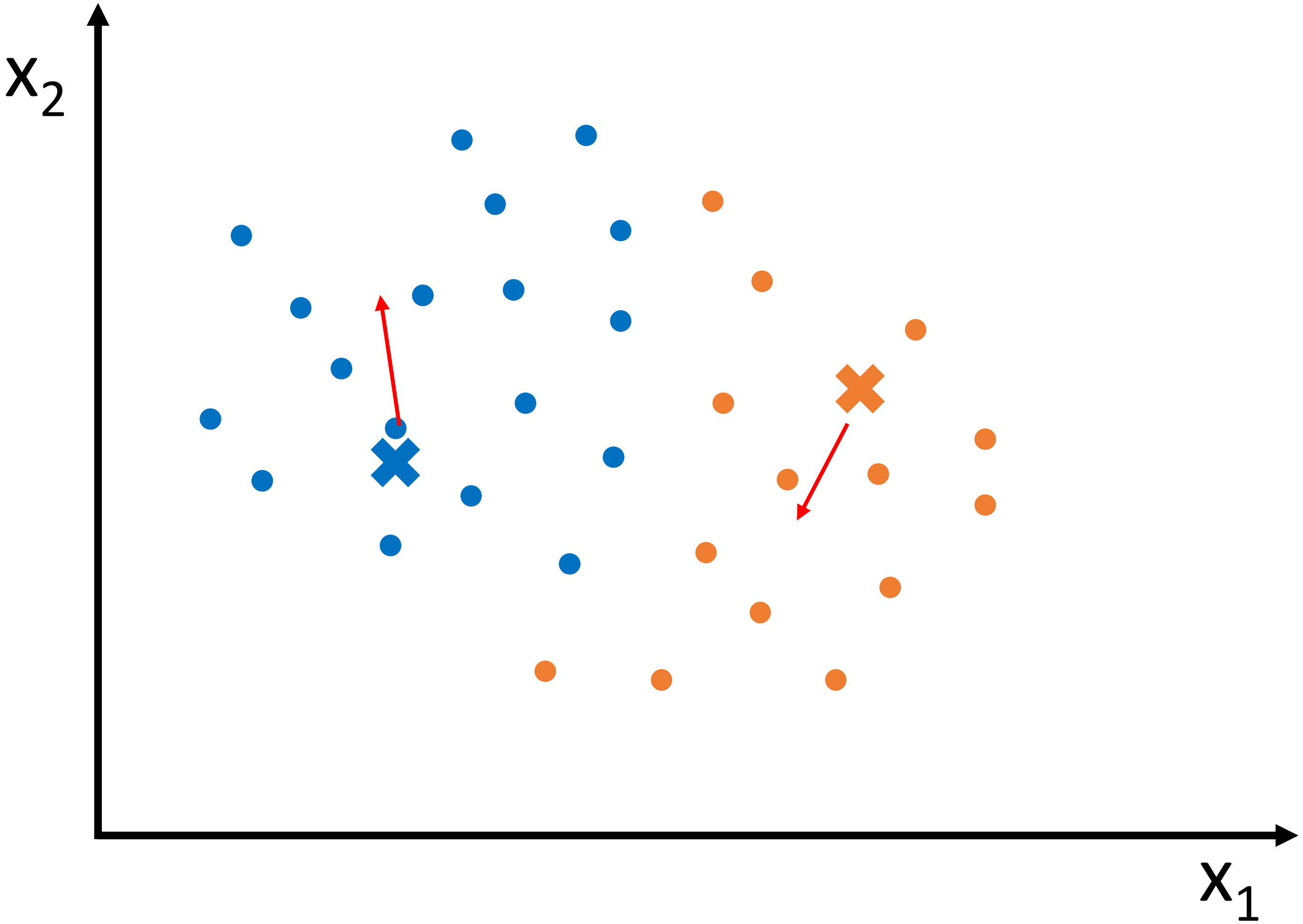
K-Mean



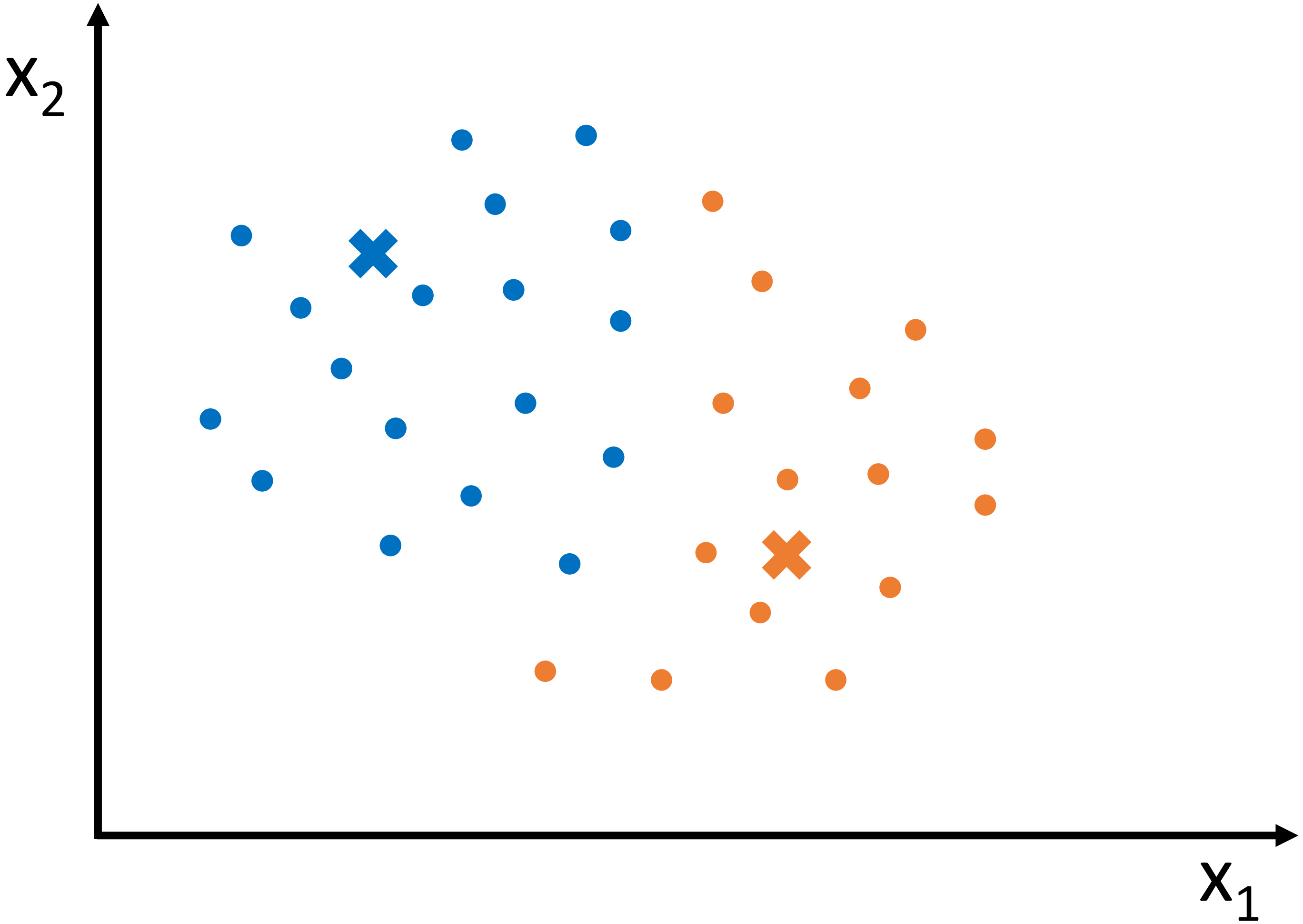
K-Mean



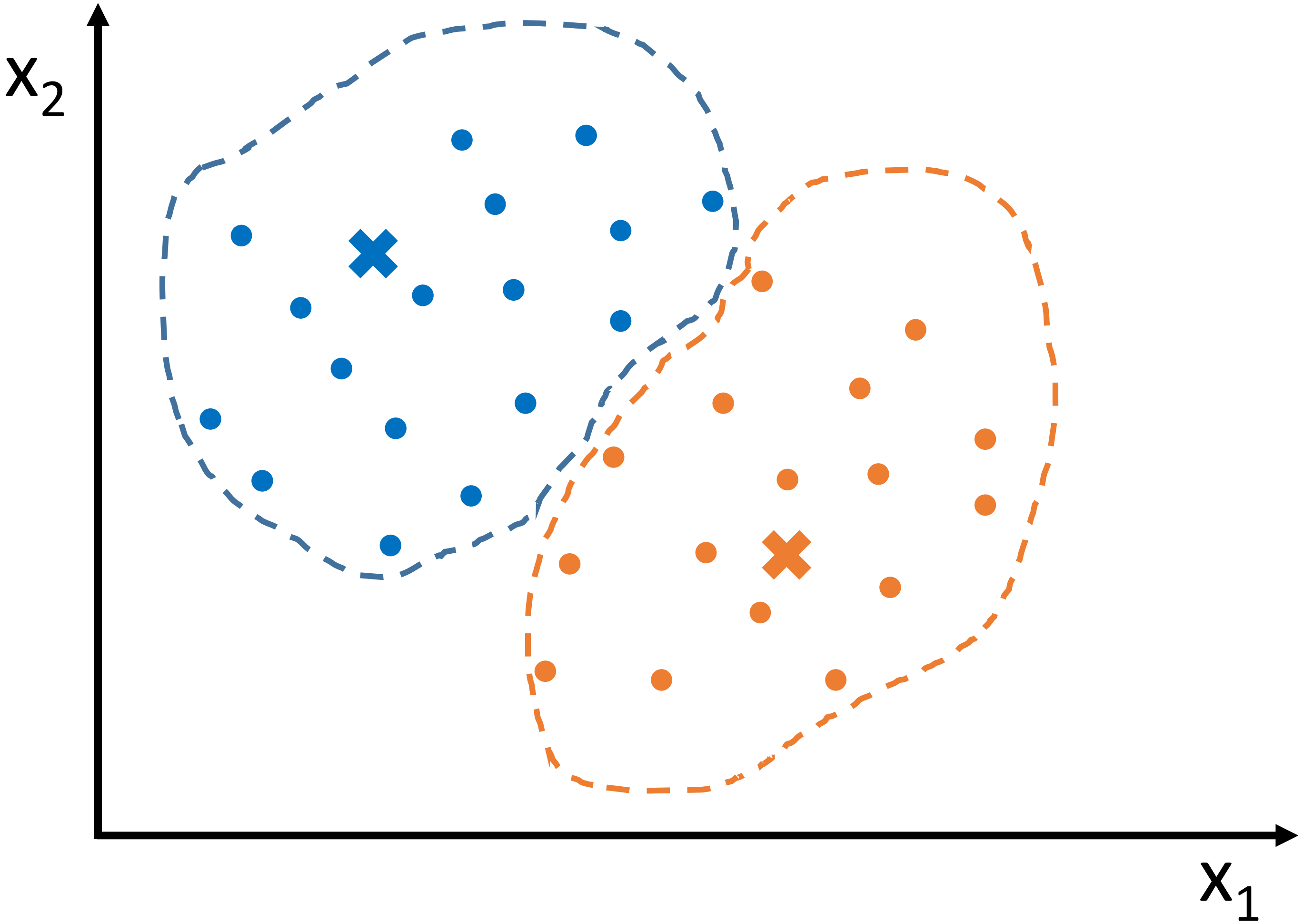
K-Mean



K-Mean



K-Mean

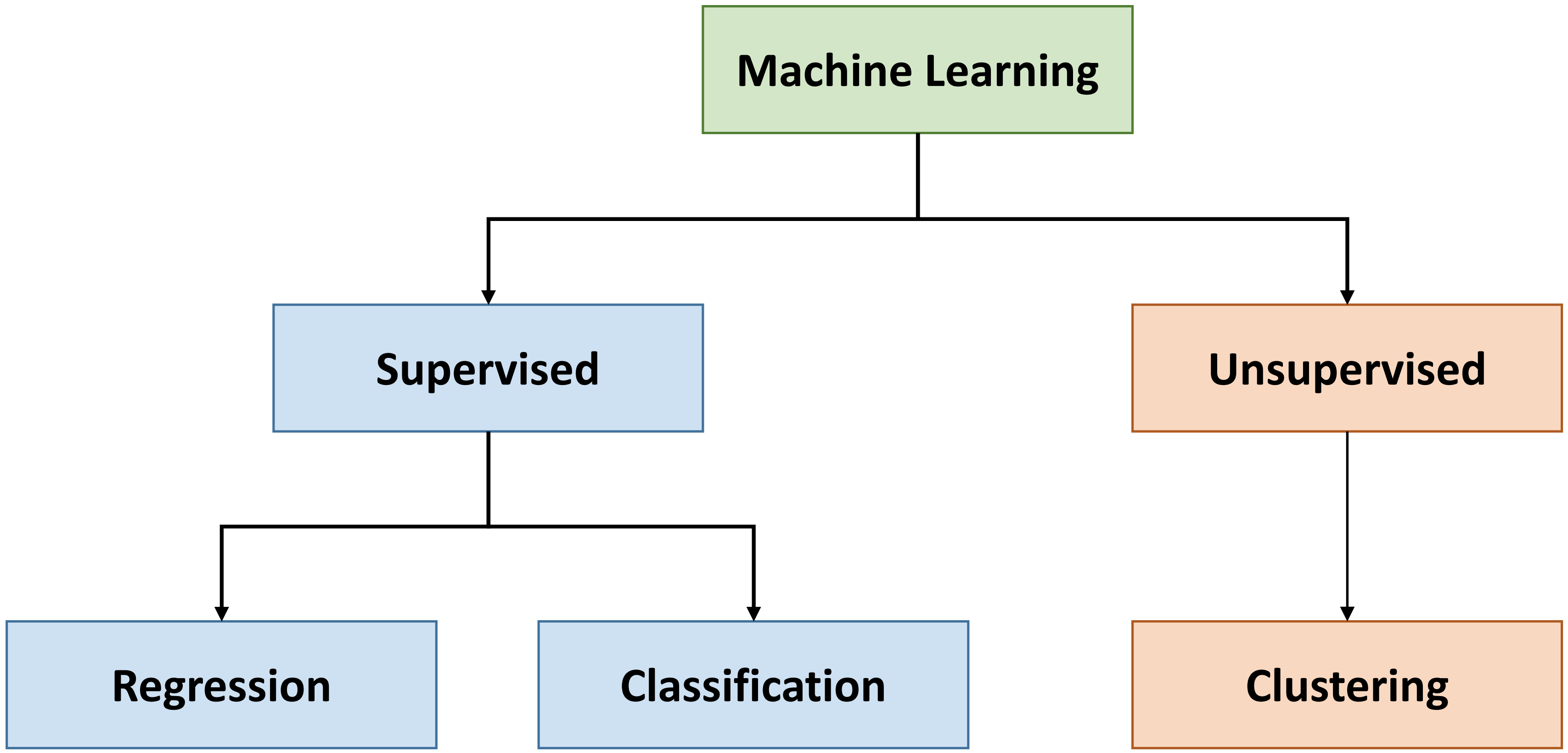


K-Mean Algorithm

- Select number of clusters: K (number of centroids $\mu_1, \dots, \mu_K$)
- Given dataset of size N
- $for\ number\ of\ iterations\ t$:
- $for\ i = 1\ to\ N$:
    - $c_i :=$ assign cluster c_i to sample x_i as the smallest Euclidean distance between x_i and the centroids
- $for\ k = 1\ to\ K$:
    - $\mu_k :=$ mean of the points assigned to cluster c_k

Demo

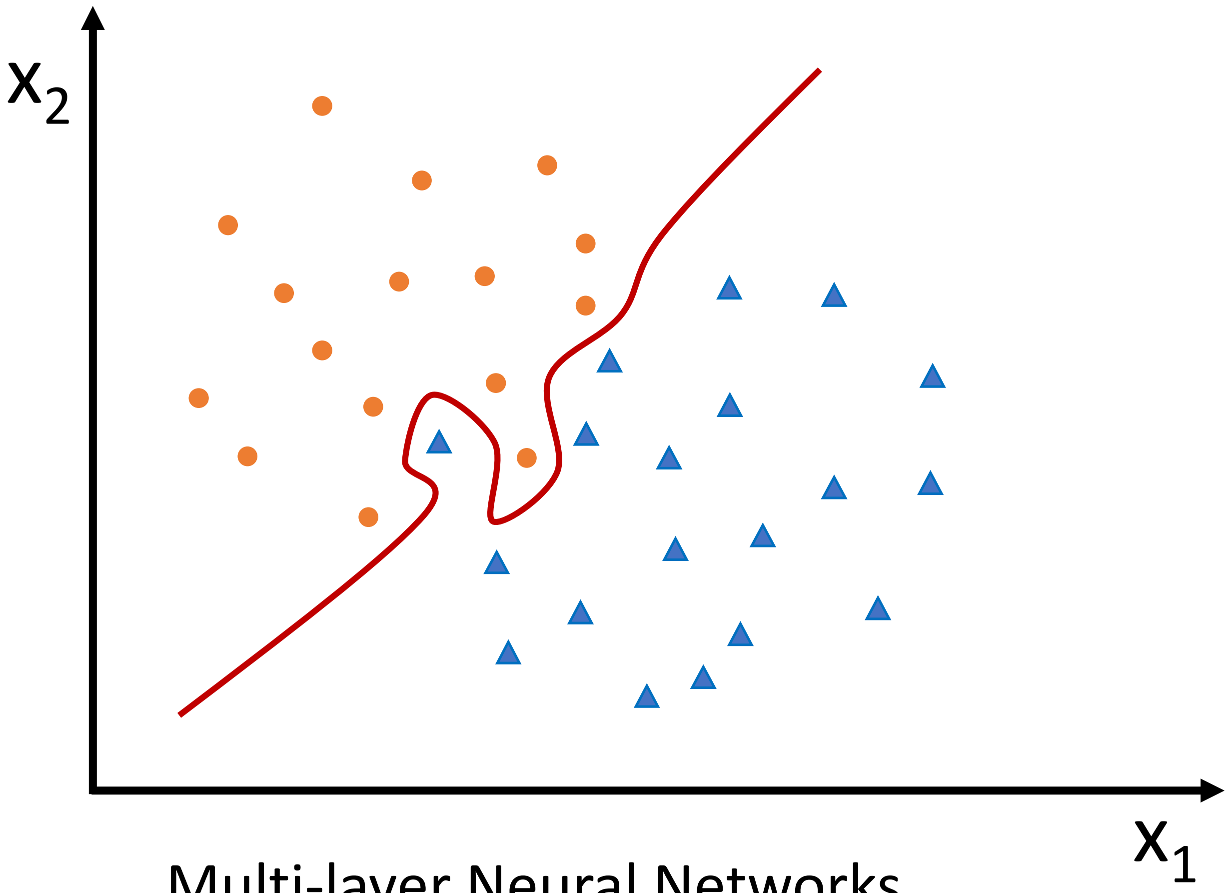
- Scikit-learn library’s k-mean


Machine Learning
Supervised
Unsupervised
Regression
Classification
Clustering

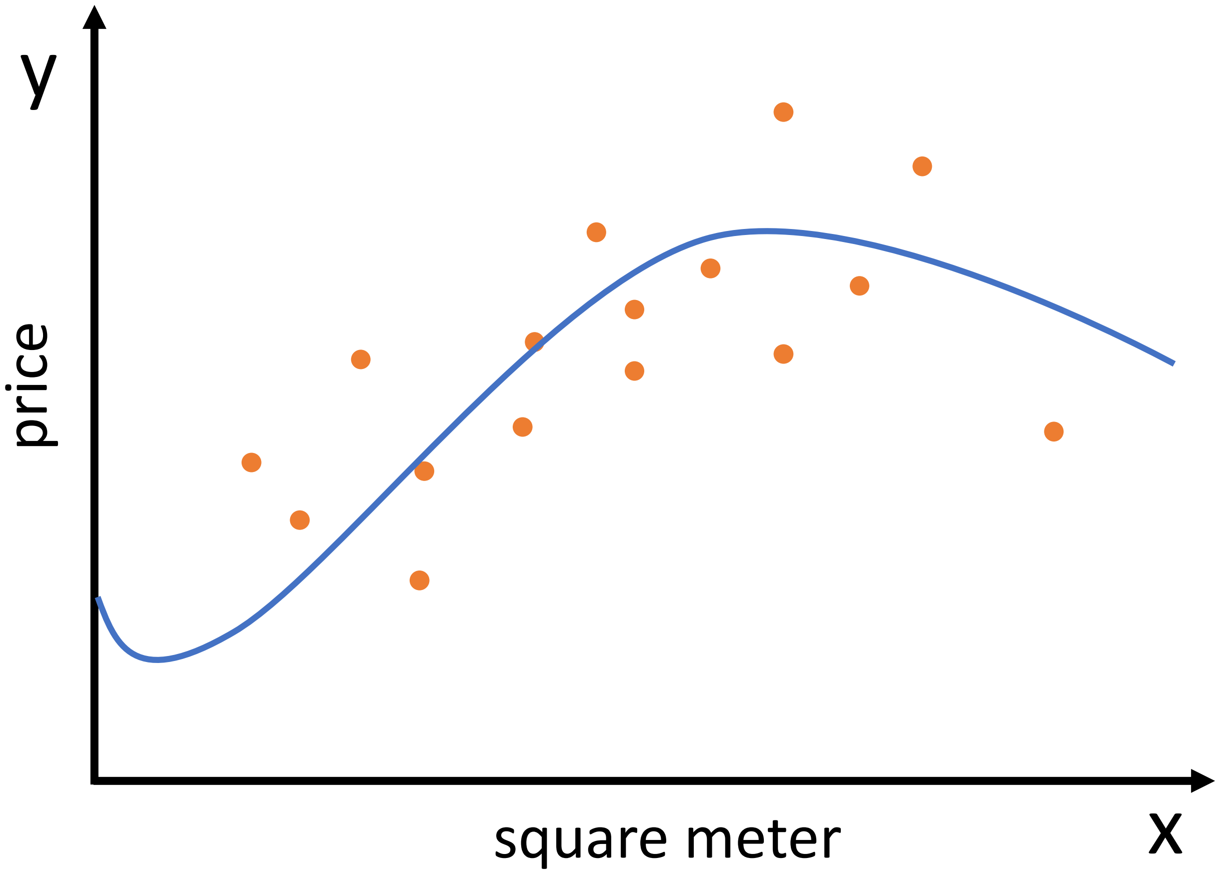
Other Machine Learning Algorithms

- Probabilistic models
- Ensemble methods
- Reinforcement Learning
- Recommendation algorithms (e.g., Matrix Factorization)
- Deep Learning

Linear vs Non-linear

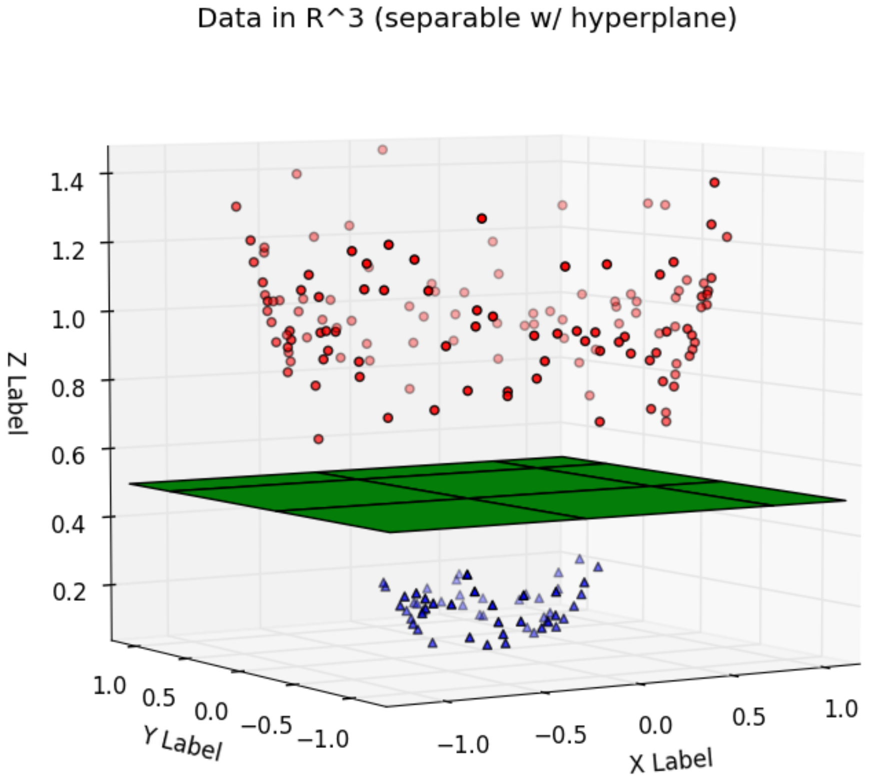


Multi-layer Neural Networks
Support Victor Machines (kernel trick)



Kernel Trick

$$K([x_1, x_2]) = [x_1, x_2, x_1^2 + x_2^2]$$



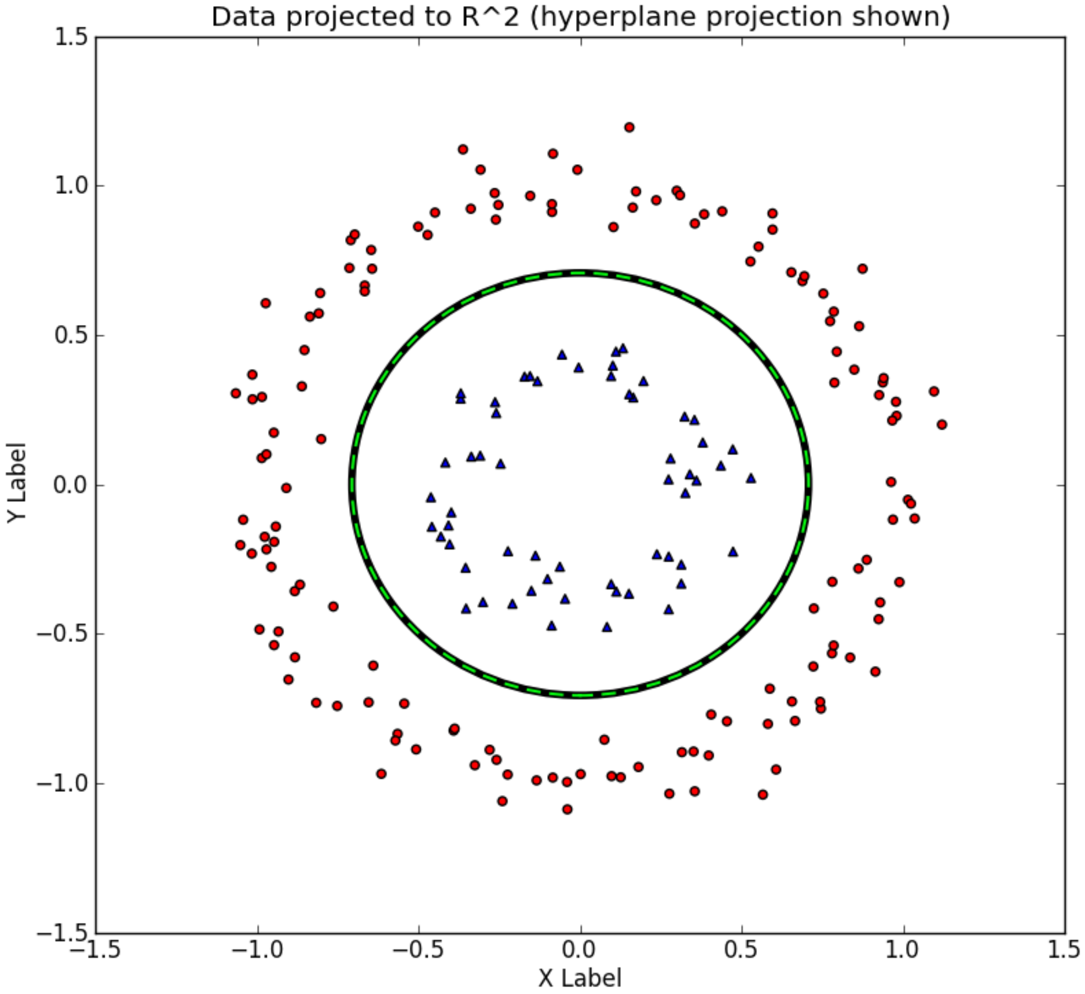


Image from http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

Practical aspects

Data Preprocessing

- Missing data
  - Elimination (samples/features)
  - Imputation
- Categorical data
  - Mapping (for ordinal features)
  - One-hot-encoding (for nominal features)
- Features scaling (normalization, standardization)

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{max}}{x_{max} - x_{min}}$$

$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x},$$

$$where\ \mu_x: mean\ of\ feature\ x, \sigma_x: standard\ deviation$$

- Data/problem specific preprocessing (e.g., images, signals, text)

Model Evaluation

- Splitting data (training, validation, testing) IMPORTANT
  - No hard rule: usually 60%-20%-20% will be fine

Training	Validation	Testing

- k-fold cross-validation
  - If dataset is very small
  - Leave-one-out
- Fine-tuning hyper-parameters
  - Automated hyper-parameter selection
  - Using validation set

Performance Measures

- Depending on the problem
- Some of the well-known measure are:
- Classification measures
  - Accuracy
  - Confusion matrix and related measures
- Regression
  - Mean Squared Error
  - R^2 metric
- Clustering performance measure is not straight forward, and will not be discussed here

Performance Measures: Accuracy

$$Accuracy = \frac{correct\ prediction}{all\ prediction}$$

- If we have 100 persons, one of them having cancer. What is the accuracy if classify all of them as having no cancer?
- Accuracy is not good for heavily biased class distribution

Performance Measures: Confusion matrix

		Actual Class	
		Positive	Negative
Predicated Class	Positive	Ture Positives (TP)	False Positives (FP)
	Negative	False Negatives (FN)	True Negatives (TN)

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

$$Precision = \frac{TP}{TP + FP}$$

a measure of result relevancy

$$Recall = \frac{TP}{TP + FN}$$

a measure of how many truly relevant results are returned

$$F - measure = 2 * \frac{Precision * Recall}{Precision + Recall}$$

the harmonic mean of precision and recall

Performance Measures: Mean Squared Error (MSE)

- Defined as

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2$$

- Gives an idea of how wrong the predictions were
- Only gives an idea of the magnitude of the error, but not the direction (e.g. over or under predicting)
- Root Mean Squared Error (RMSE) is the square root of MSE, which has the same unit of the data

Performance Measures: R^2

- Is a statistical measure of how close the data are to the fitted regression line
- Also known as the **coefficient of determination**
- Has a value between 0 and 1 for no-fit and perfect fit, respectively

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \ , where\ \bar{y}\ is\ the\ mean\ of\ the\ observed\ data$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Dimensionality Reduction

Curse of dimensionality

when the dimensionality increases → the volume of the space increases so fast that the available data become sparse

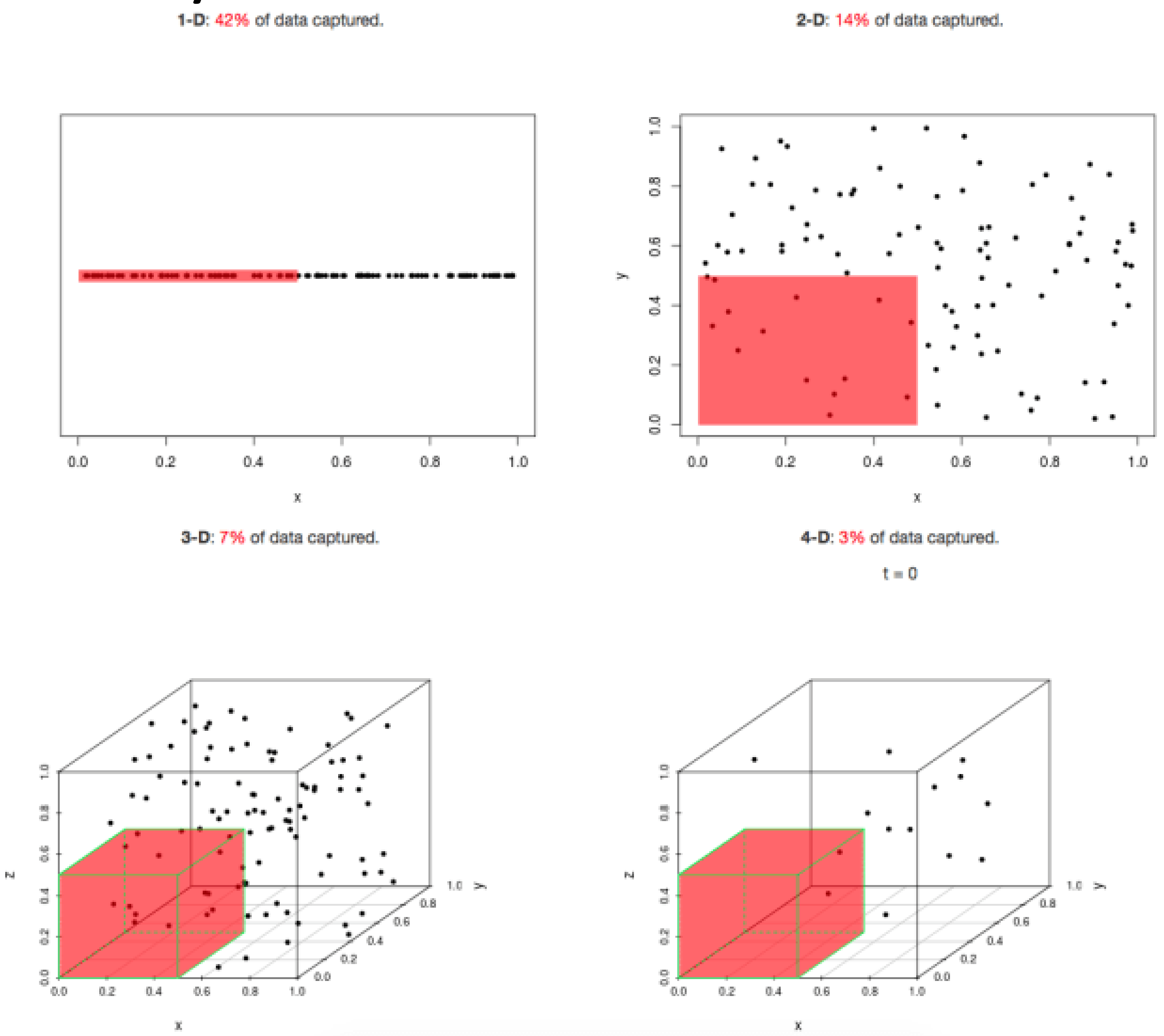


Image from http://www.newsnshit.com/curse-of-dimensionality-interactive-demo/

Feature selection

- Comprehensive (all subsets)
- Forward stepwise
- Backward stepwise
- Forward-Backward
- Many more…

Features compression/projection

- Project to lower dimensional space while preserving as much information as possible
- Principle Component Analysis (PCA)
- Unsupervised method

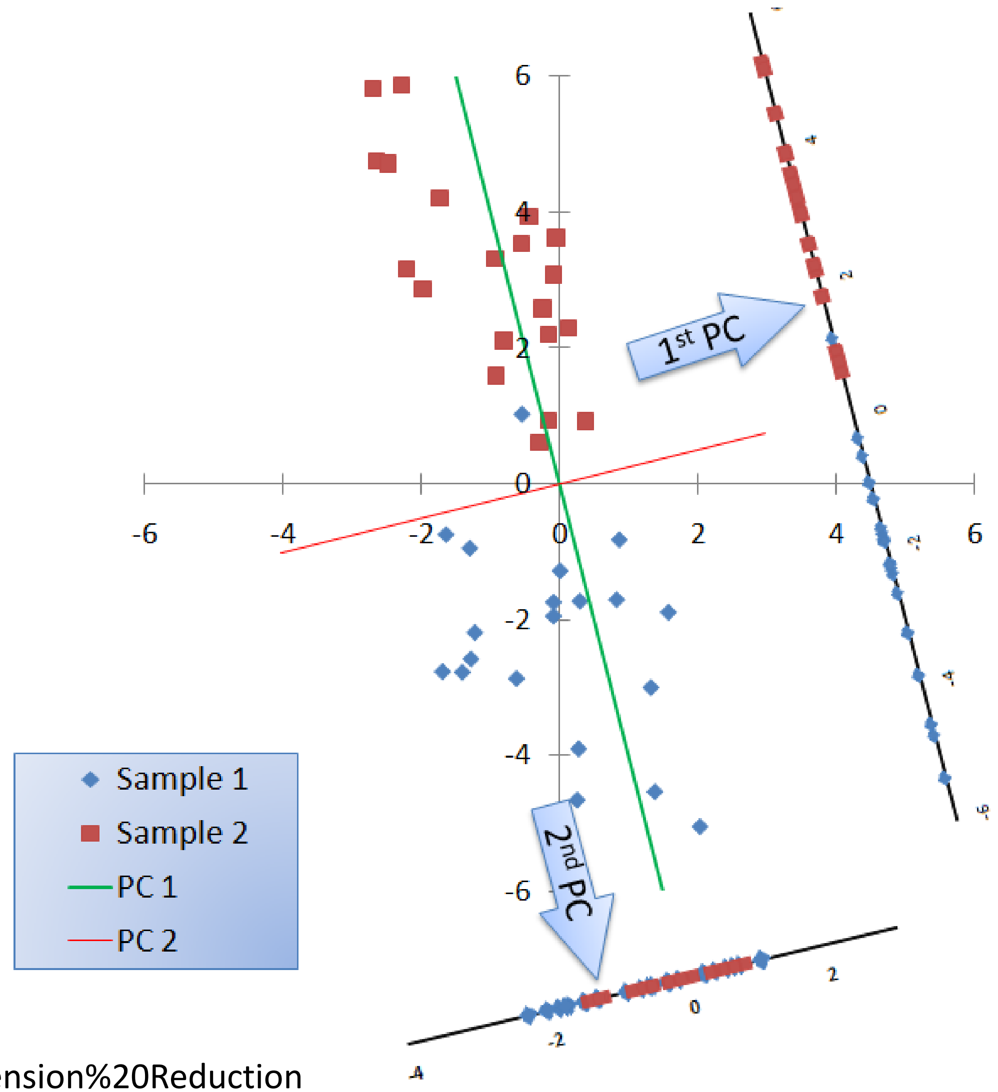
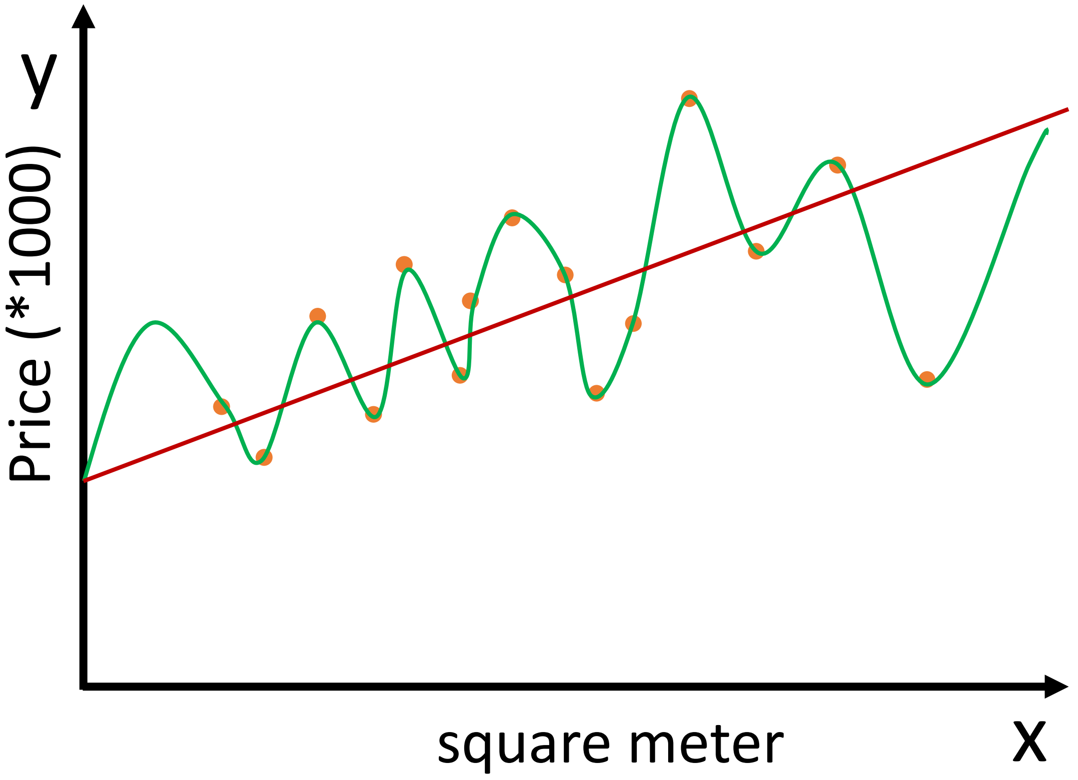


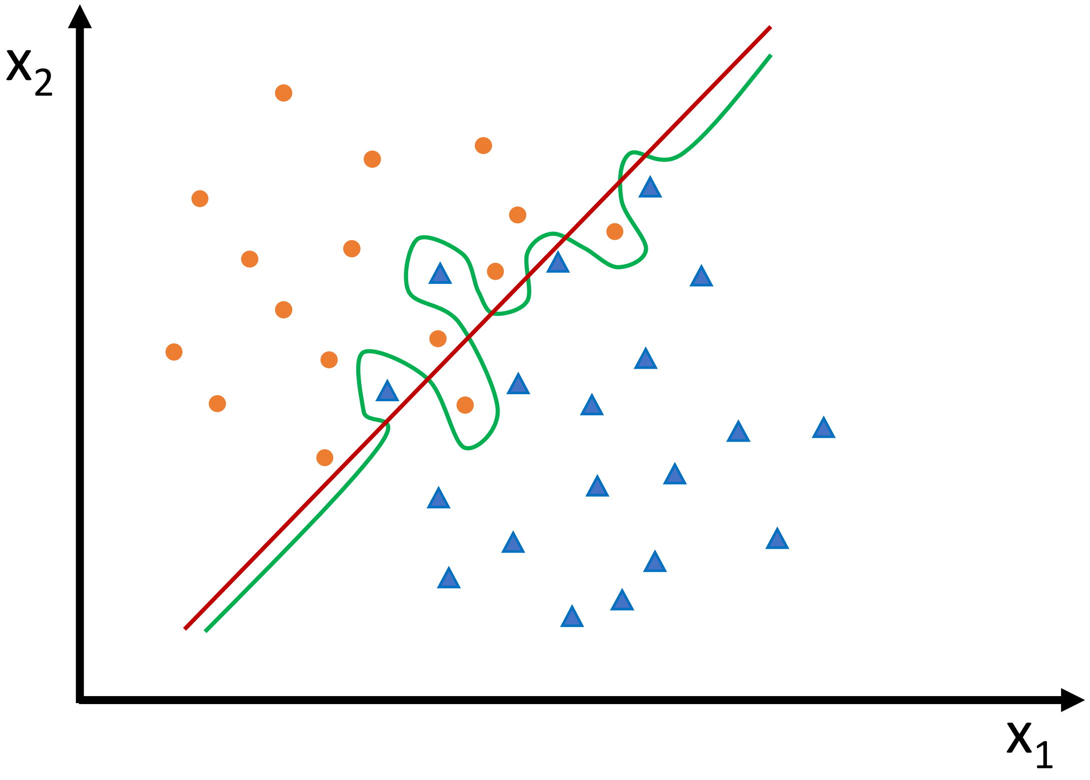
Image from http://compbio.pbworks.com/w/page/16252905/Microarray%20Dimension%20Reduction

Overfitting and Underfitting

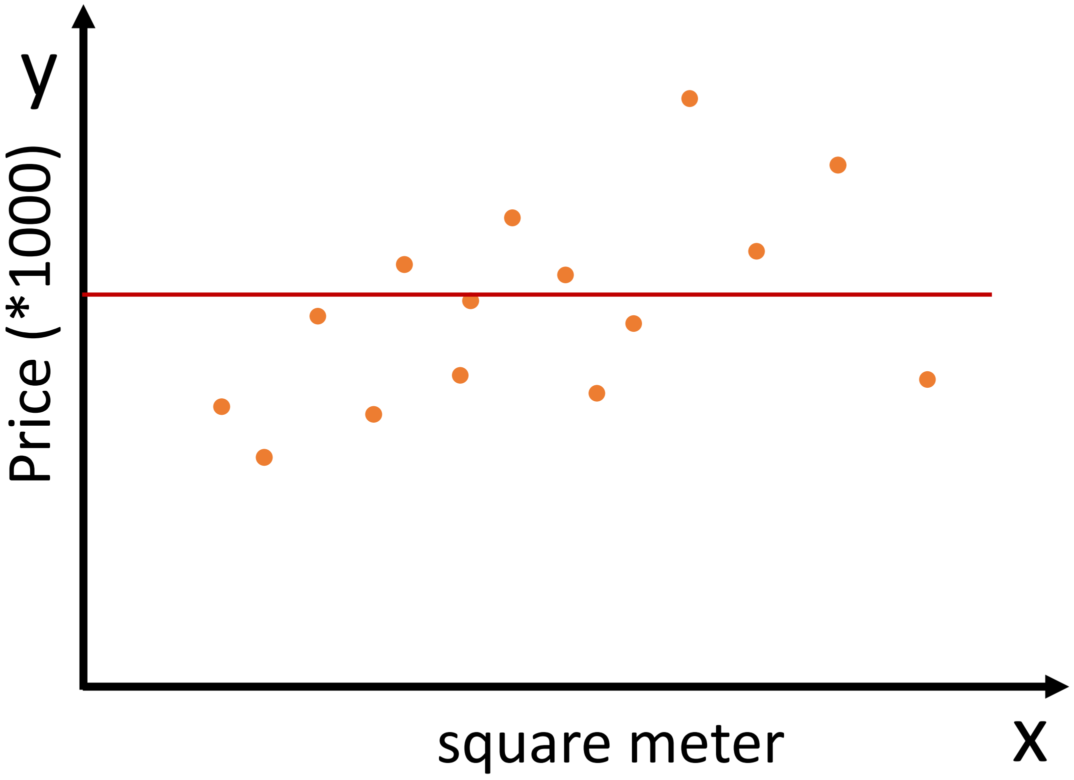
Overfitting



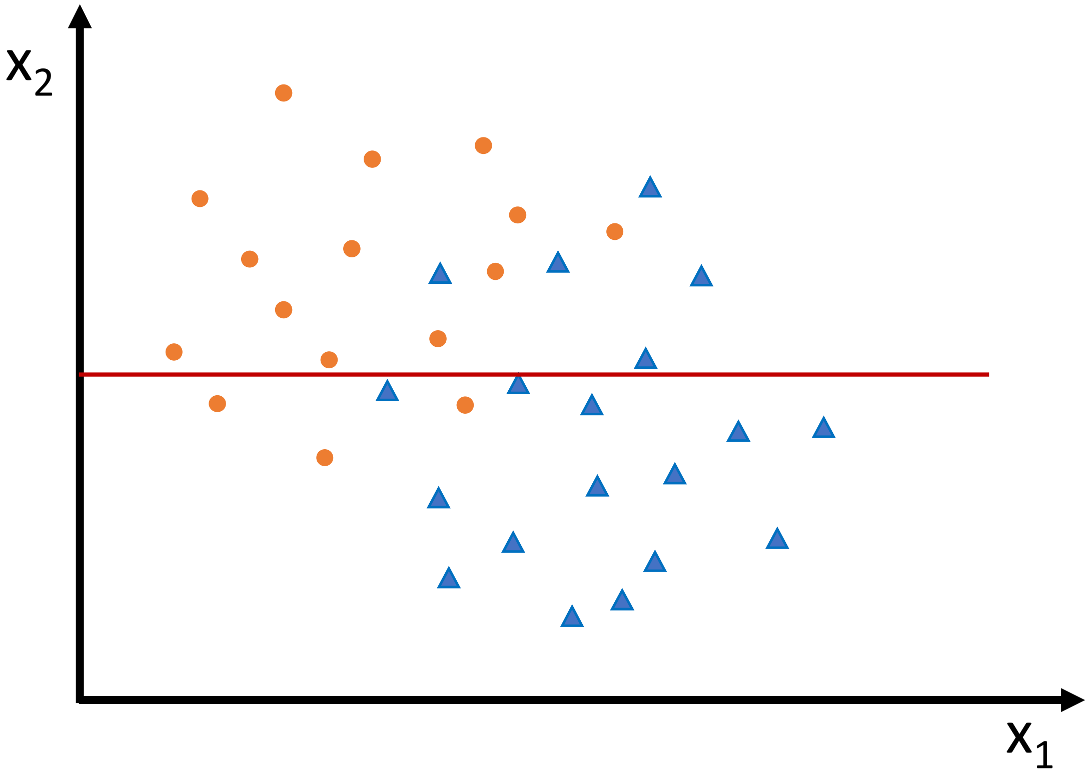
High Variance



Underfitting



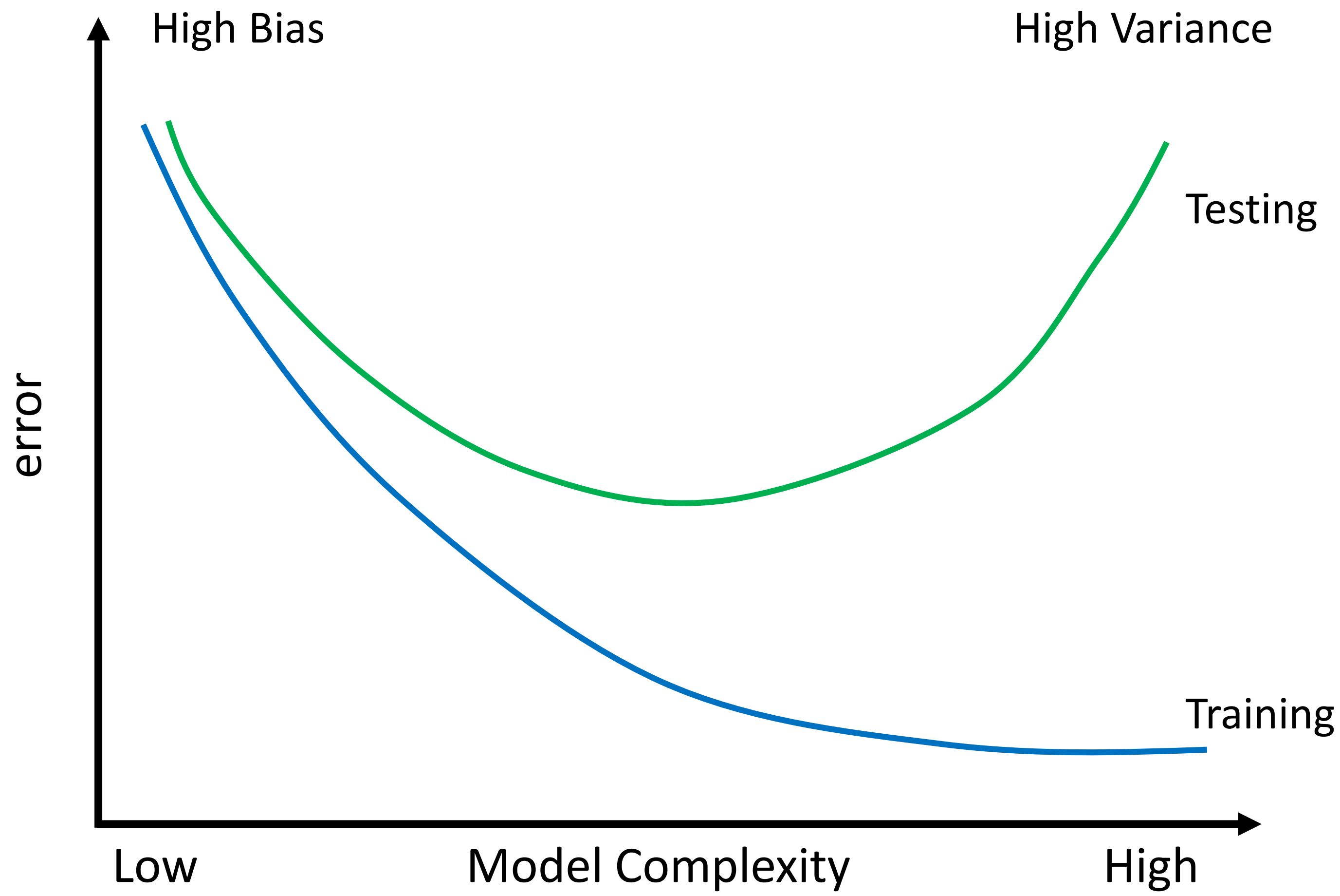
High Bias



Training vs. Testing Errors

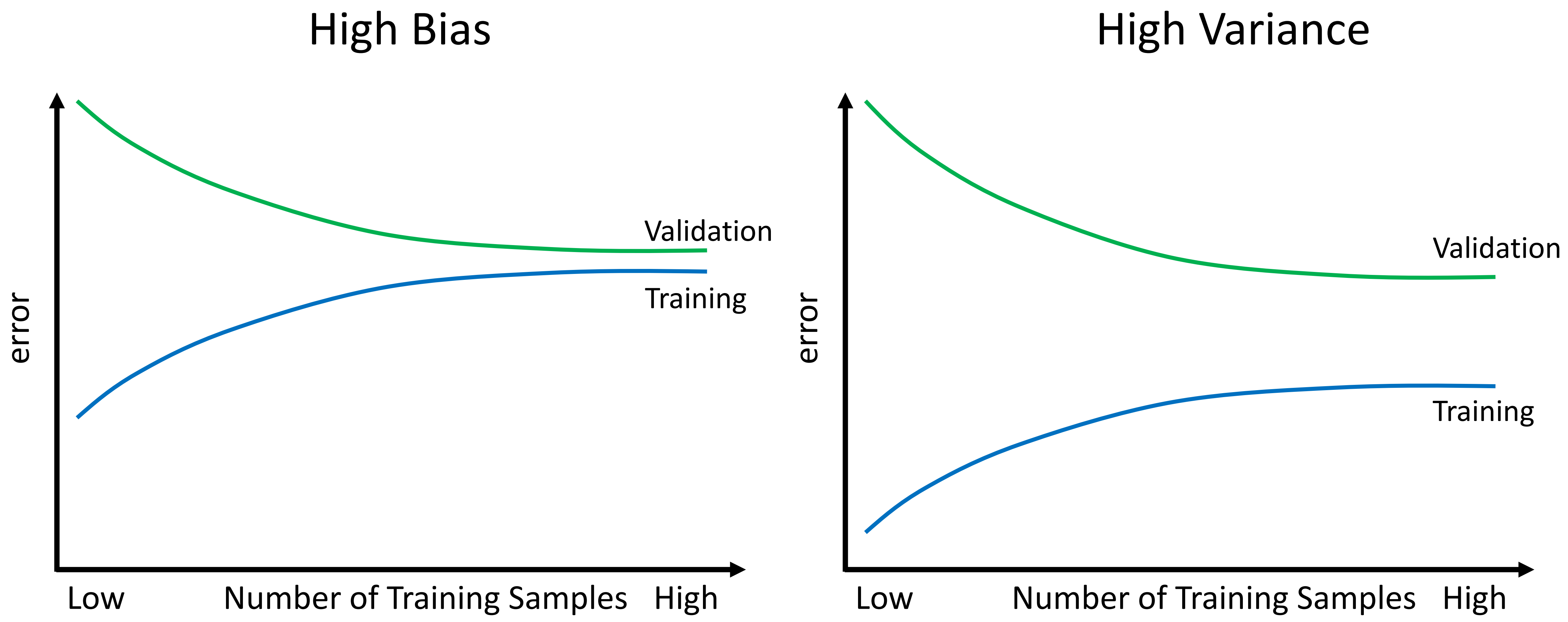
- Accuracy on training set is not representative of model performance
- We need to calculate the accuracy on the test set (a new unseen examples)
- The goal is to generalize the model to work on unseen data

Bias and variance trade-off



The optimal is to have low bias and low variance

Learning Curves



Regularization

- To prevent overfitting
- Decrease the complexity of the model
- Example of regularized regression model (Ridge Regression)

$$RSS(w_0, w_1) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 + \lambda \sum_{j}^{k} w_j^2,$$

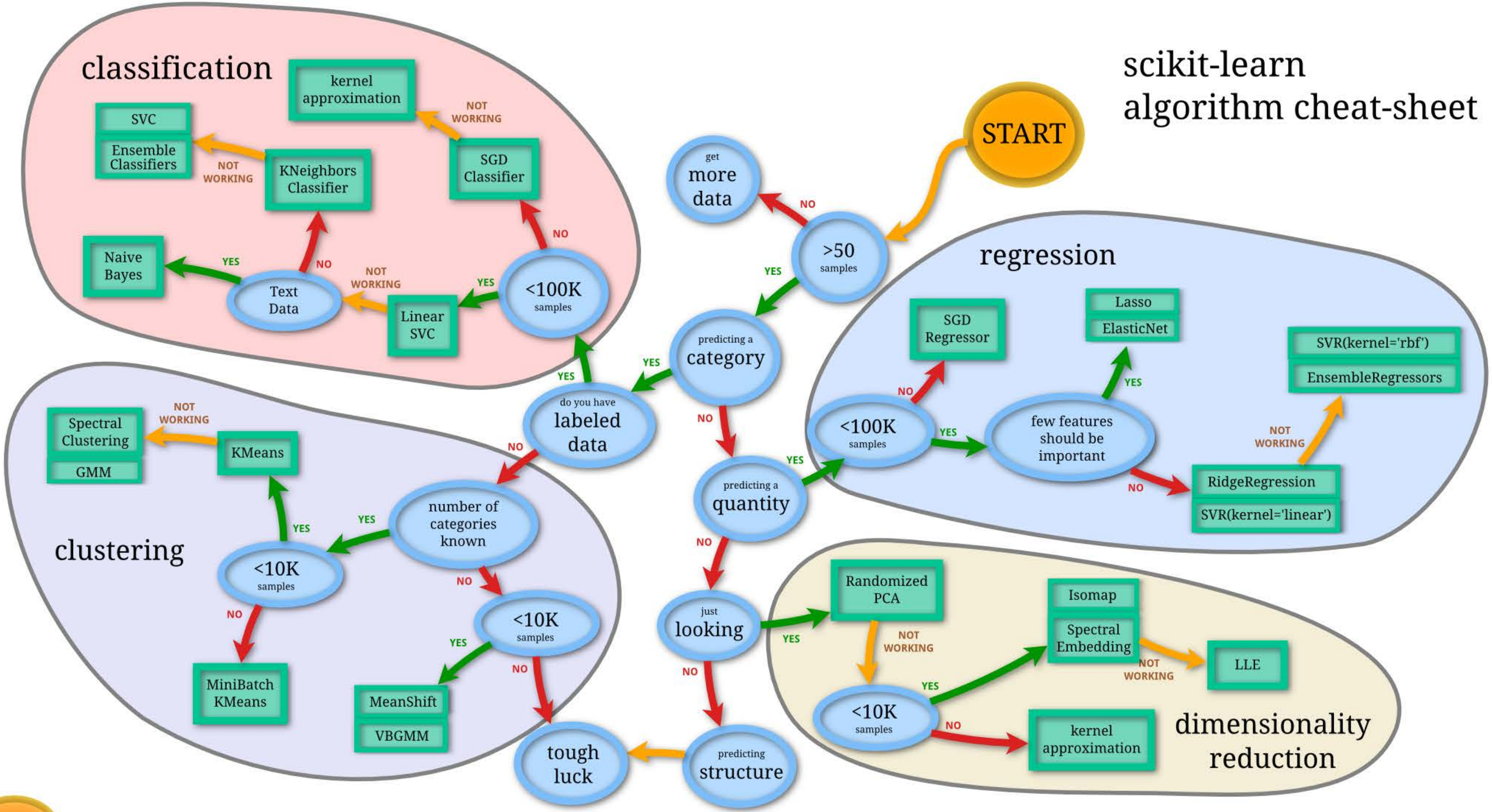
$$k = number\ of\ weights$$

- λ is a very important hyper-parameter

Debugging a Learning Algorithm

- From "Machine Learning" course on coursera.org, by Andrew Ng

- Get more training examples → fixes high variance
- Try smaller sets of features → fixes high variance
- Try getting additional features → fixes high bias
- Try adding polynomial features (e.g., $x_1^2, x_2^2, x_1, x_2, etc$) → fixes high bias
- Try decreasing λ → fixes high bias
- Try increasing λ → fixes high variance



scikit-learn
algorithm cheat-sheet
START
classification
regression
clustering
dimensionality reduction
>50 samples
get more data
predicting a category
do you have labeled data
<100K samples
Linear SVC
Text Data
Naive Bayes
KNeighbors Classifier
SVC
Ensemble Classifiers
SGD Classifier
kernel approximation
predicting a quantity
SGD Regressor
few features should be important
Lasso
ElasticNet
RidgeRegression
SVR(kernel='linear')
SVR(kernel='rbf')
EnsembleRegressors
number of categories known
<10K samples
KMeans
Spectral Clustering
GMM
MiniBatch KMeans
MeanShift
VBGMM
tough luck
just looking
predicting structure
Randomized PCA
Isomap
Spectral Embedding
LLE
YES
NO
NOT WORKING

What is the best ml algorithm?

- **“No free lunch” theorem:** there is no one model that works best for every problem
- We need to try and compare different models and assumptions
- Machine learning is full of uncertainty

متلازمة ويكا
Weka syndrome

نصائح لتطبيق تعلم الآلة

- افهم المشكلة التي أنت بصدد حلها جيداً
  - ما هي المشكلة؟
  - ما المطلوب حله بالضبط؟
  - هل تحتاج أن تتعلم أمور متعلقة بالمجال؟ (طبي، تسويق، ...)
- افهم البيانات المتاحة جيداً
  - حجم البيانات
  - إجراء بعض الإحصاءات الوصفية (descriptive statistics) على البيانات
  - هل توجد أجزاء ناقصة؟ كيف تتعامل معها؟

نصائح لتطبيق تعلم الآلة

- حدد عدّة خوارزميات لاختبارها بناءً على وصف المشكلة والبيانات المتاحة
  - Regression? Classification? Clustering? Other?
  - هل تحتاج regularization؟
- الخصائص المميزة features
  - هل هي جاهزة، أم تحتاج أن تستخلصها؟ (مثلاً من صورة أو نصوص)
  - هل تحتاج لتقليلها؟ (feature selection or projection)
  - هل تحتاج إلى scaling؟

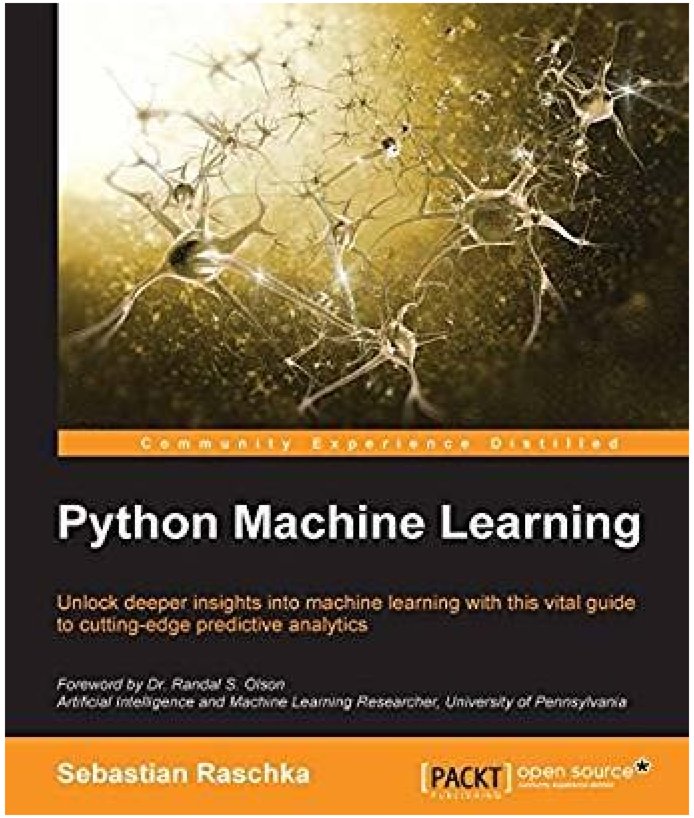
نصائح لتطبيق تعلم الآلة

- صمم الاختبار
  - كيف تقسم البيانات؟ (60% training, 20% validation, 20% testing)
  - Evaluation Measures
  - Hyper-parameters selection (using validation split)
  - Plot learning curves to asses bias and variance
- ماذا تفعل؟
  - المزيد من البيانات؟
  - تقليل الخصائص أو زيادتها أو مزجها؟
- بعد أن تنتهي من كل هذا، طبق الخوارزميات التي اخترتها على بيانات الاختبار testing split، واختر منها ما تعتقد أنه الأنسب

برنامج مقترح لتعلم المجال

- مراجعة المواضيع التالية في الرياضيات والإحصاء
  - Descriptive Statistics
  - Inferential Statistics
  - Probability
  - Linear Algebra
  - Basics of differential equations

- قراءة كتاب: Python Machine Learning



برنامج مقترح لتعلم المجال

- التسجيل في دورة “Machine Learning” في coursera.org
  - www.coursera.org/learn/machine-learning
  - وحل جميع التمارين

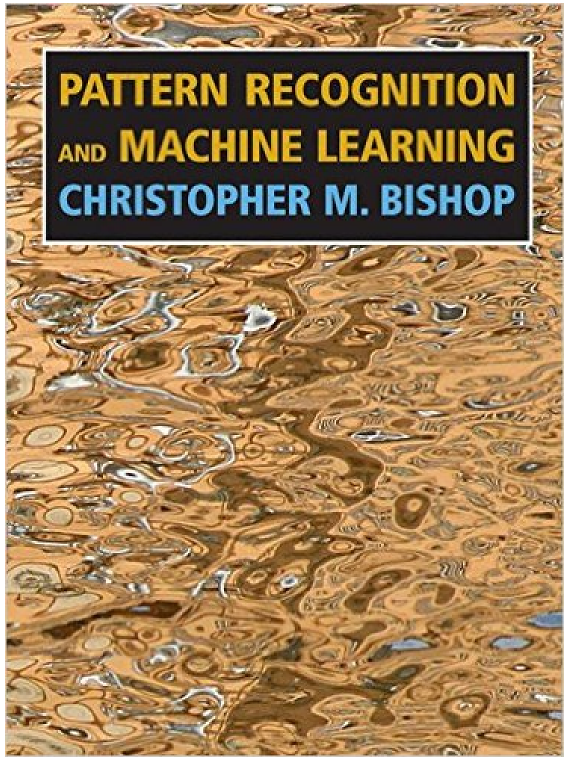
- تعلم لغة برمجة والمكتبات ذات العلاقة بتعلم الآلة
  - Matlab
  - Python
  - R

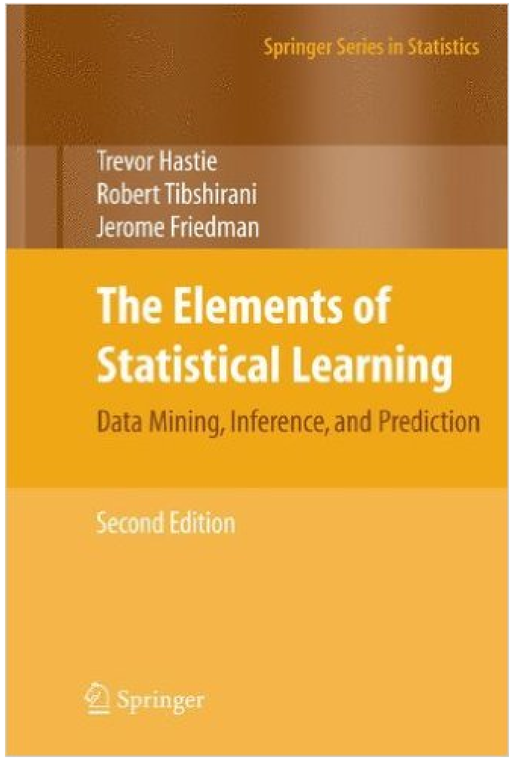
برنامج مقترح لتعلم المجال

- التسجيل في كاجل (www.kaggle.com)، والعمل على بعض البيانات والتحديات المتاحة.
- مقترح للبداية:
  - Titanic: Machine Learning from Disaster
    https://www.kaggle.com/c/titanic
  - House Prices: Advanced Regression Techniques
    https://www.kaggle.com/c/house-prices-advanced-regression-techniques
  - Digit Recognizer
    https://www.kaggle.com/c/digit-recognizer

برنامج مقترح لتعلم المجال

- مراجعة أخرى للإحصاء والرياضيات لتقوية الأمور التي تعرف الآن أنك تحتاجها
- كتب أخرى مقترحة لتعلم الآلة (أكثر تقدماً)





برنامج مقترح لتعلم المجال

- التركيز على مجال تريده
  - التنبؤ المستقبلي للأعمال (كالتسويق)
  - معالجة اللغات الطبيعية
  - رؤية الحاسب
- اعرض نتائج أعمالك
- شارك الناس عملك (مثل الكود والنتائج) واطلب رأيهم
- أقم دورة في المجال الذي تخصصت به ☺

شكراً لكم على حضوركم وإنصاتكم