CHULA ENGINEERING
Foundation toward Innovation

COMPUTER

Chula Big Data and IoT
Center of Excellence
(CUBIC)

DataMind
Data Mining Group
Machine Intelligence and Knowledge Discovery Lab
Chulalongkorn University

Data Analytics (Part1)
Python for Data Analytics

Peerapon Vateekul, Ph.D.
Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University
Peerapon.v@chula.ac.th

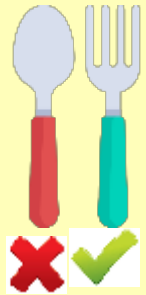
Outlines

- Fundamental terminology
- Data analytics tasks
- Scikit-learn: Machine learning library in Python
- Demo

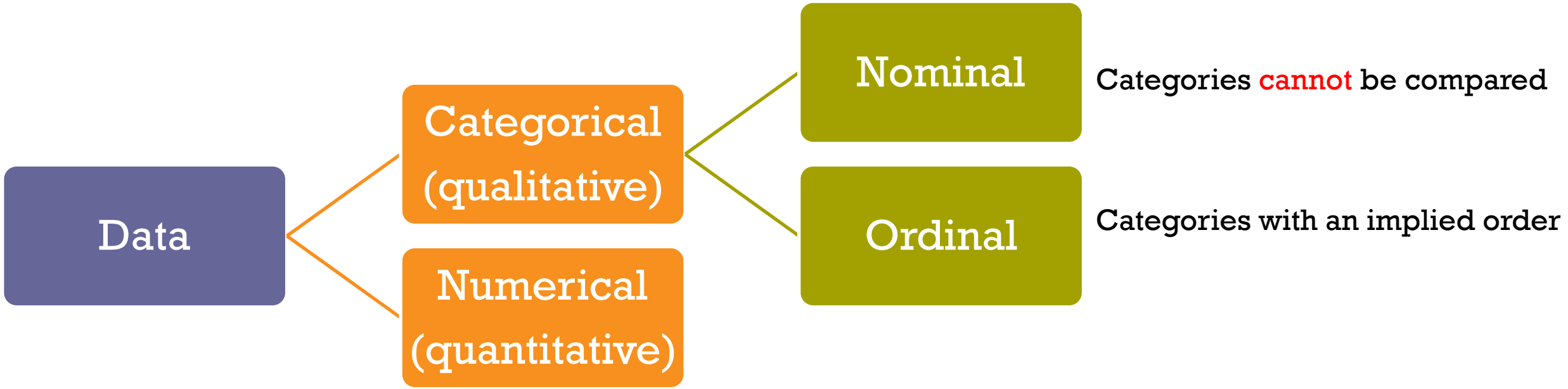
PIERIAN DATA

https://www.pieriandata.com/

- Understand data analytics **tasks**
- Be able to identify tasks and **tools** (technique) from a given problem



Terminology: Kinds of data



Terminology: Data table

		inputs		target
Age	**Income**	**Gender**	**Province**	**Purchase**
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	No

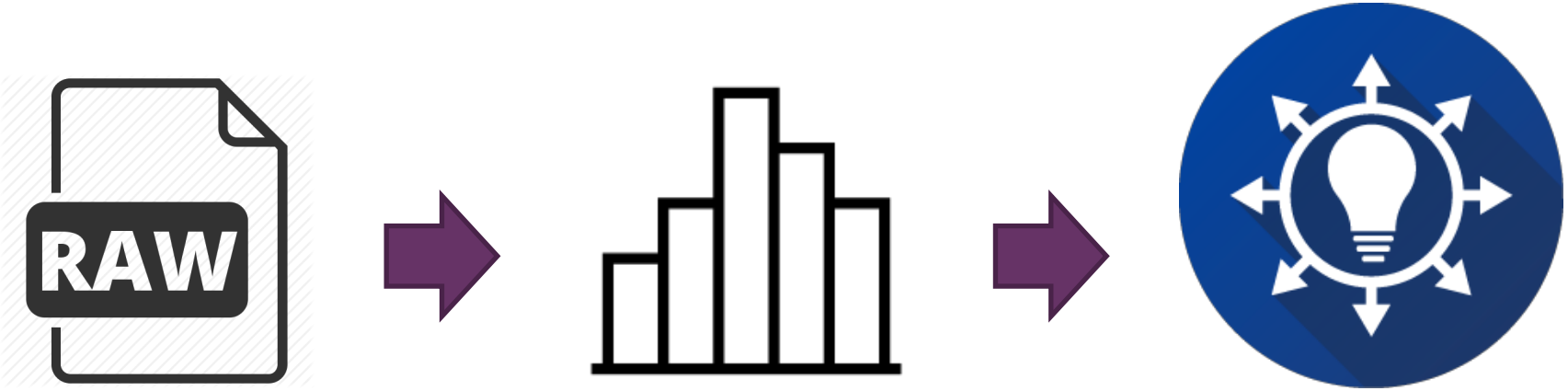
- Row
  - Example, instance, case, observation, subject
- Column
  - Feature, variable, attribute
- Input
  - Predictor, independent, explanatory variable
- Target
  - Output, outcome, response, dependent variable

+

Data Analytics Tasks

Data analytics

- **Data analytics** refers to the science of examining and exploring data in order to (i) understand it and (ii) discover useful information.
- This can lead to **better decision** and also **data product**.



Data analytics process

4 • Data Visualization

3 • Data Analytics

2 • Data Storage

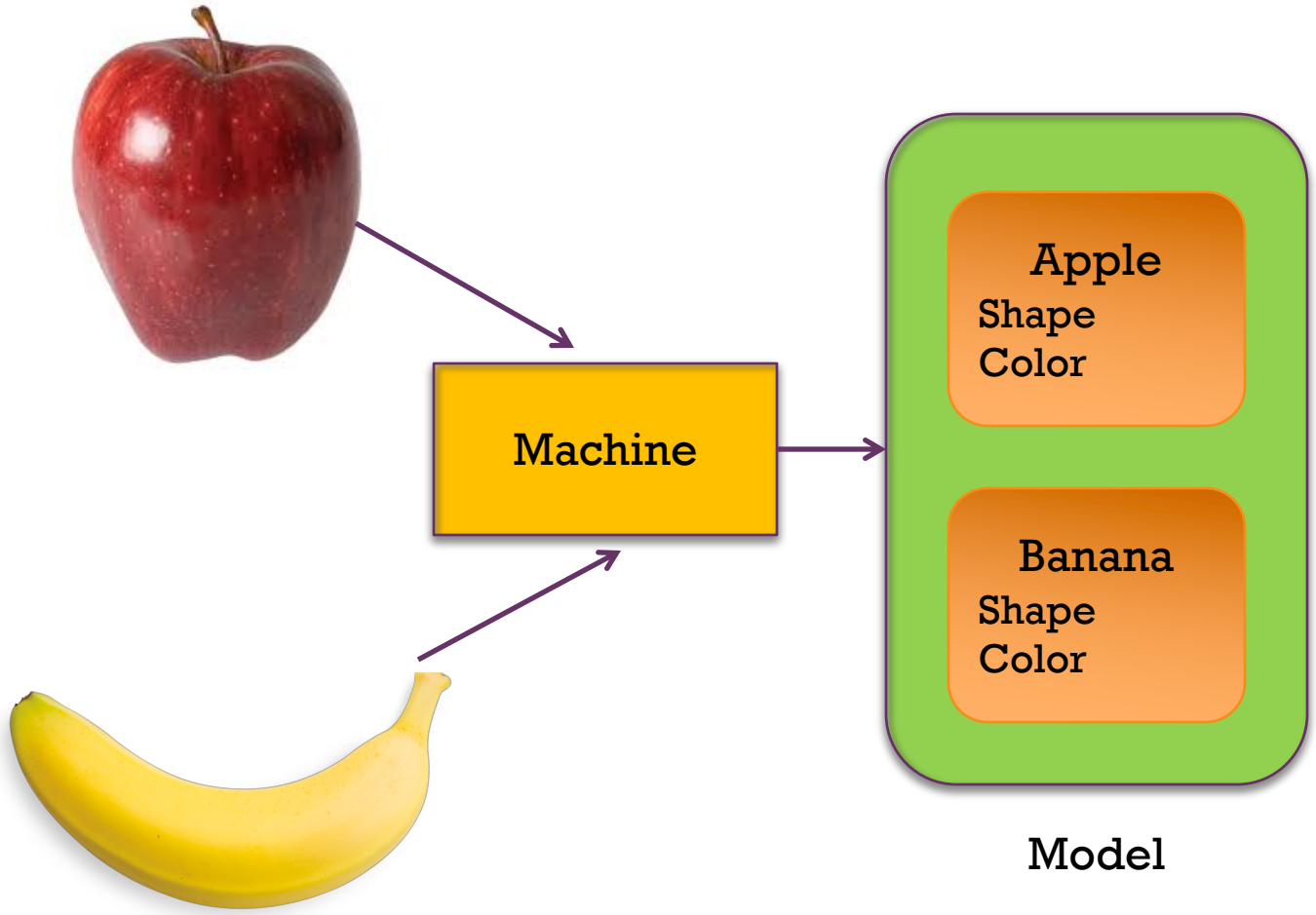
1 • System Infrastructure

Data Mining

- An automatic process of
- discovering useful information
- in large data repositories
- with sophisticated algorithm

- Supervised Learning (predictive task)
- Unsupervised Learning (descriptive task)
- Special task

Task1: Supervised learning (predictive task)

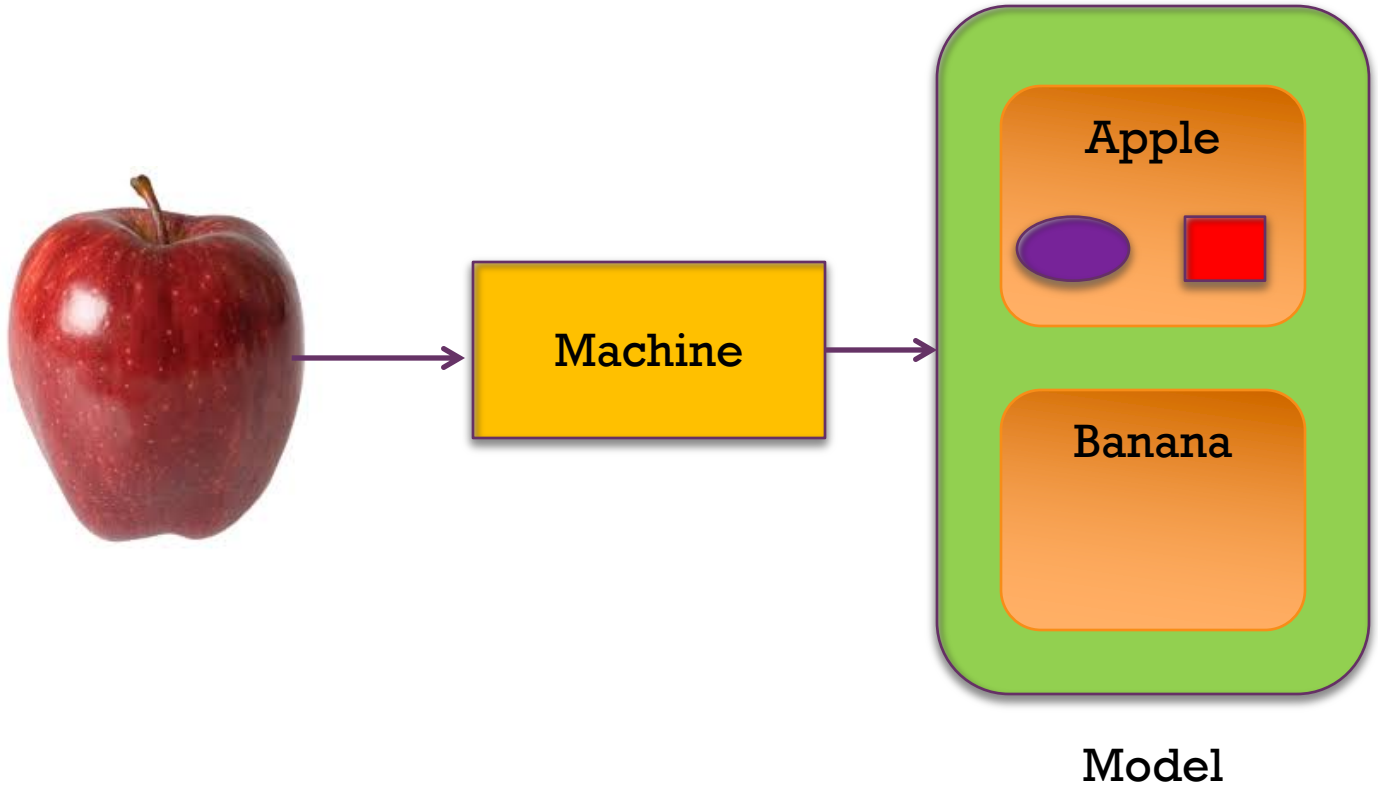


Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

1. Training Phase
2. Testing Phase

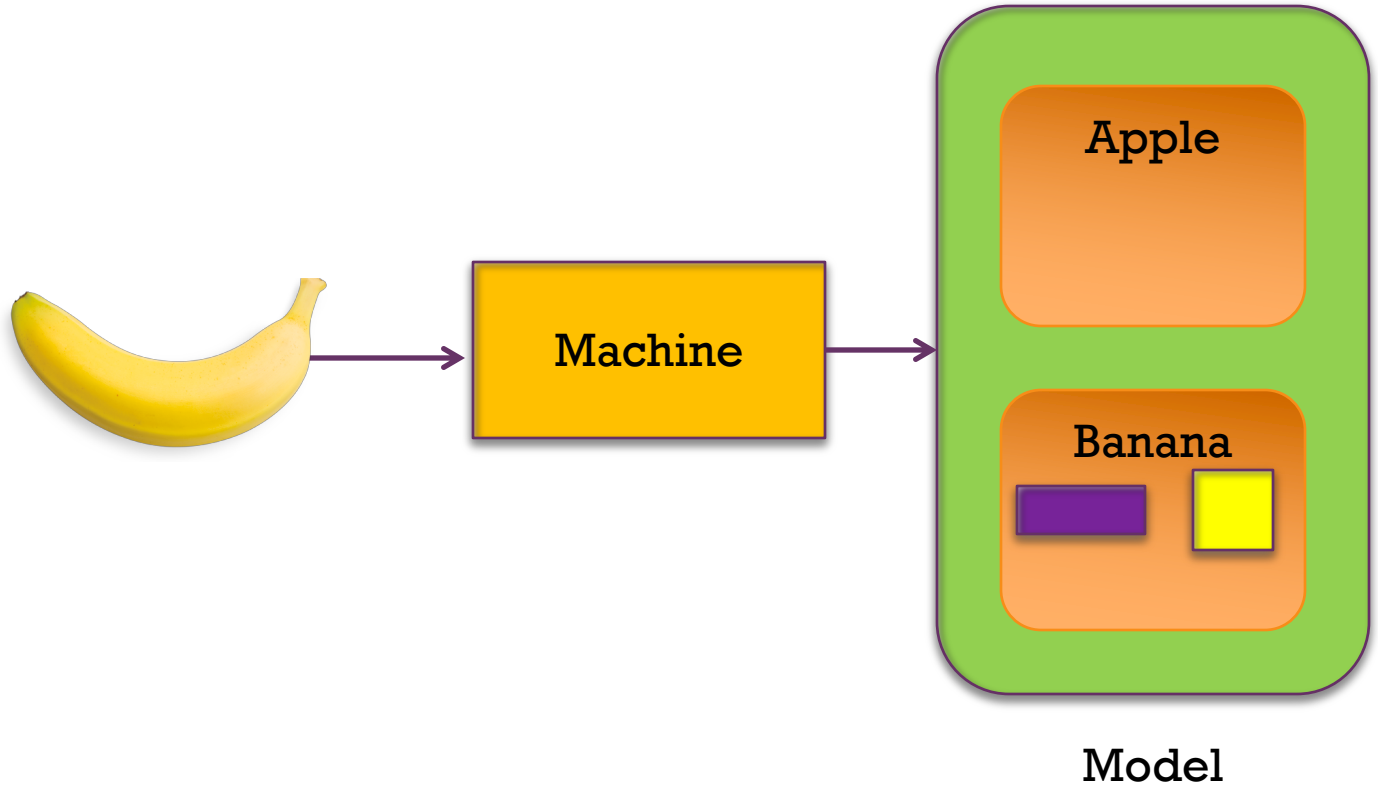
Supervised learning (cont.): Training Phase



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

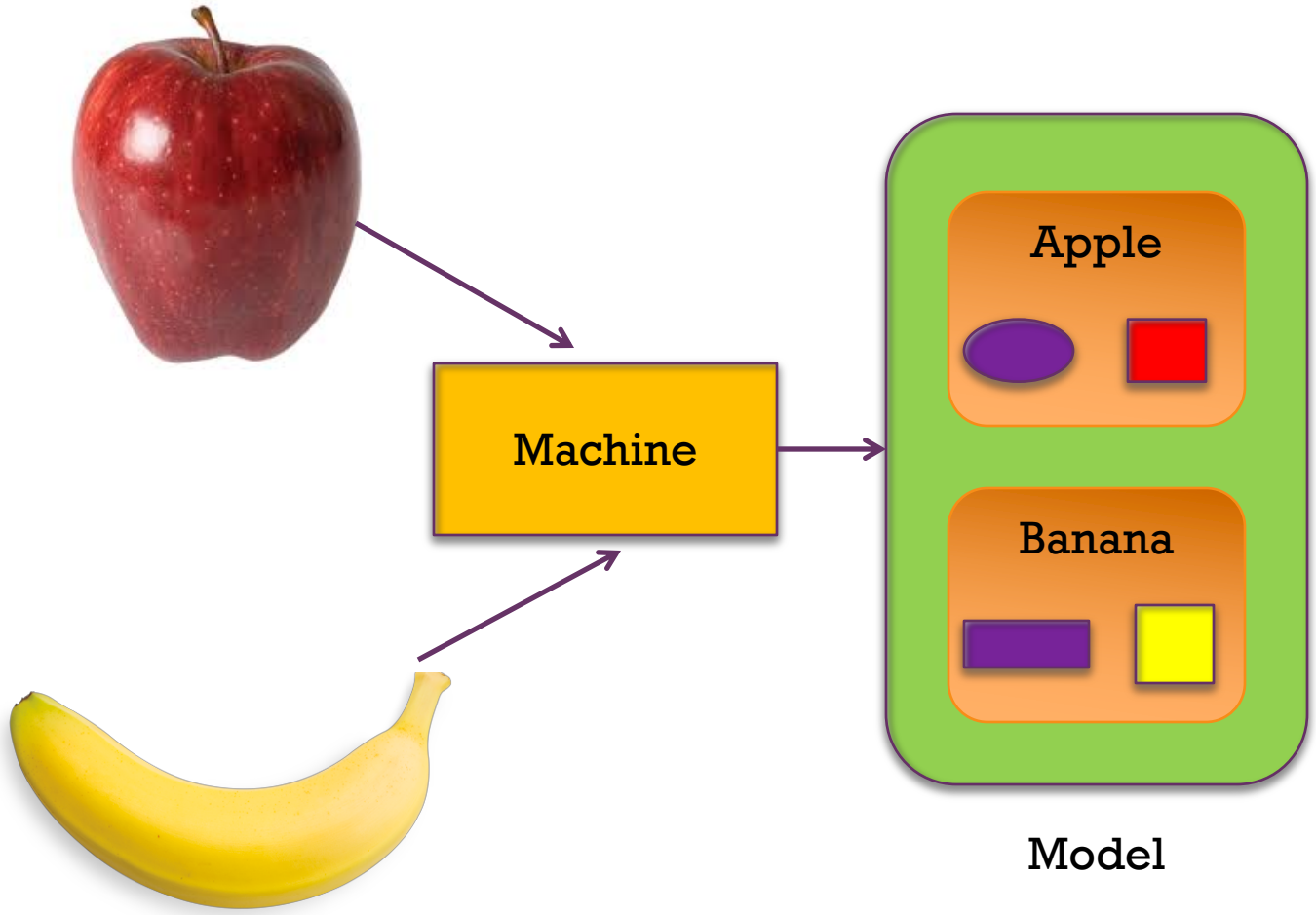
Supervised learning (cont.)
Training Phase



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

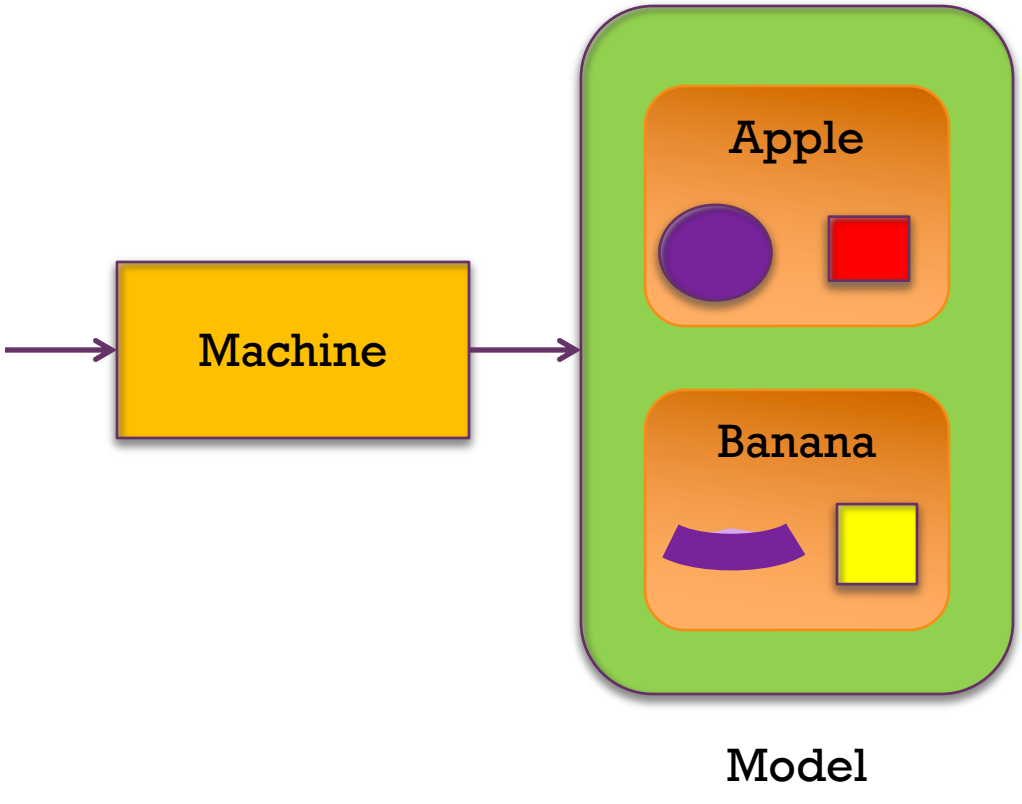
Supervised learning (cont.): Training Phase



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

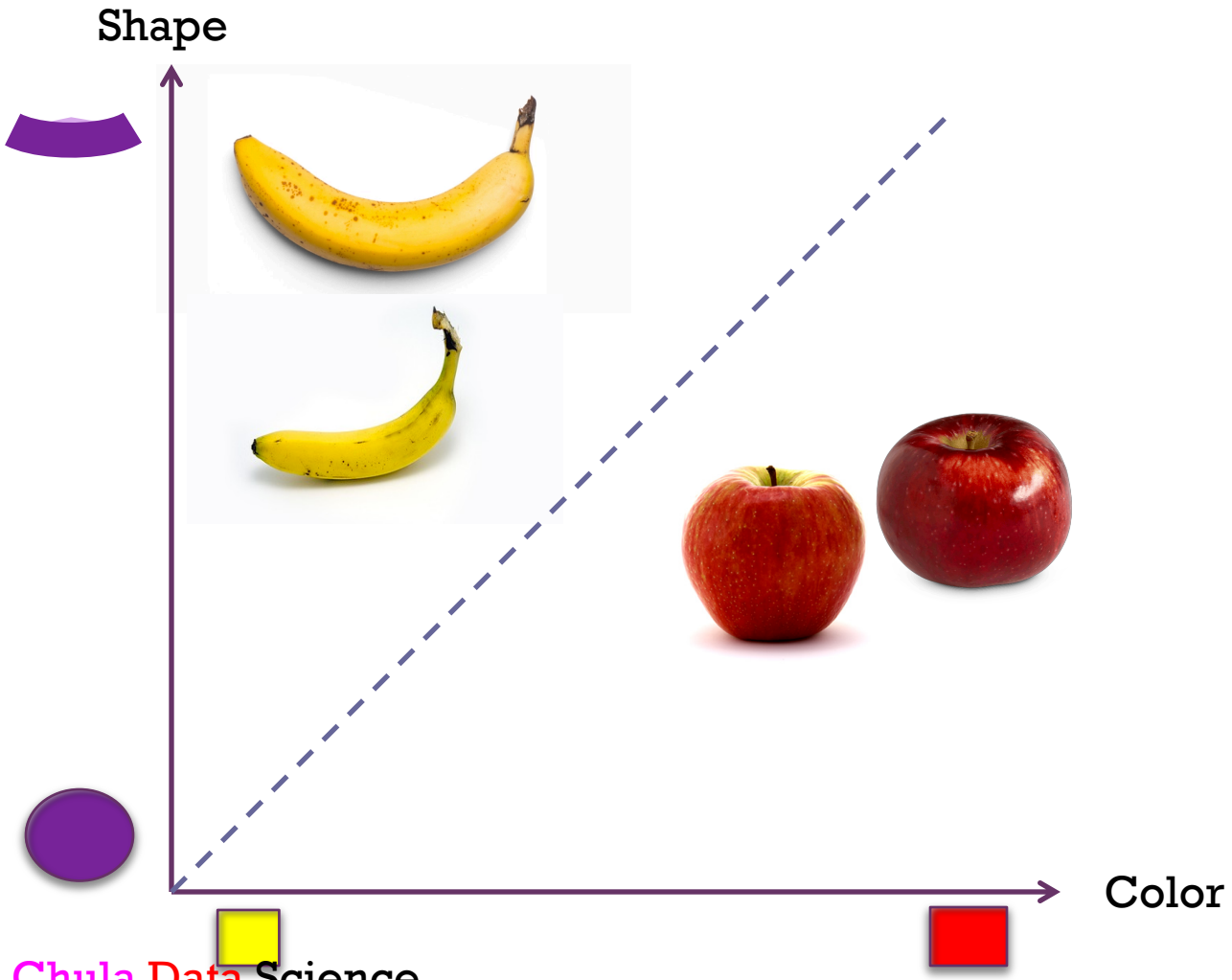
Supervised learning (cont.):
Training Phase → more examples



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

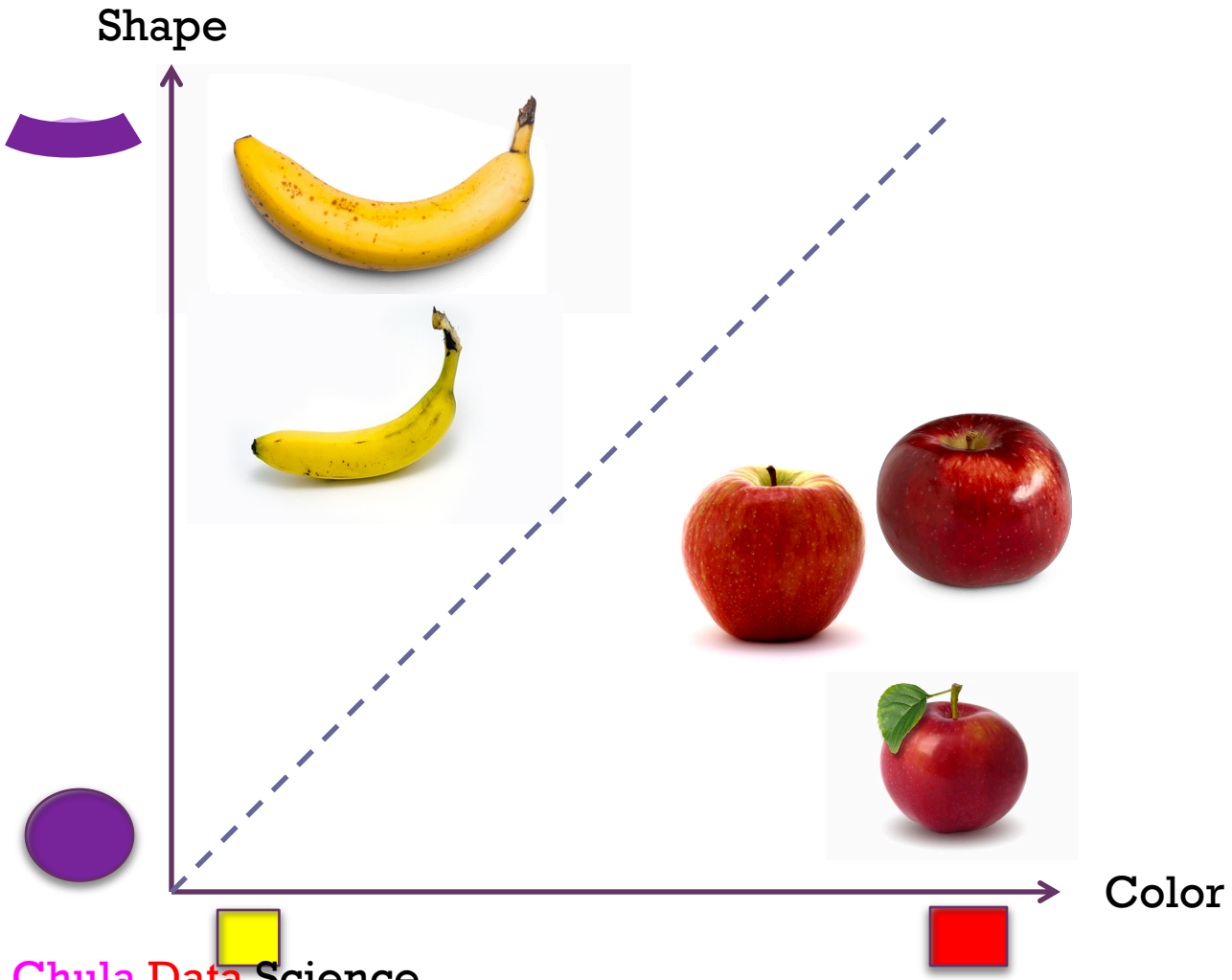
Supervised learning (cont.):
Training Phase → classification model



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

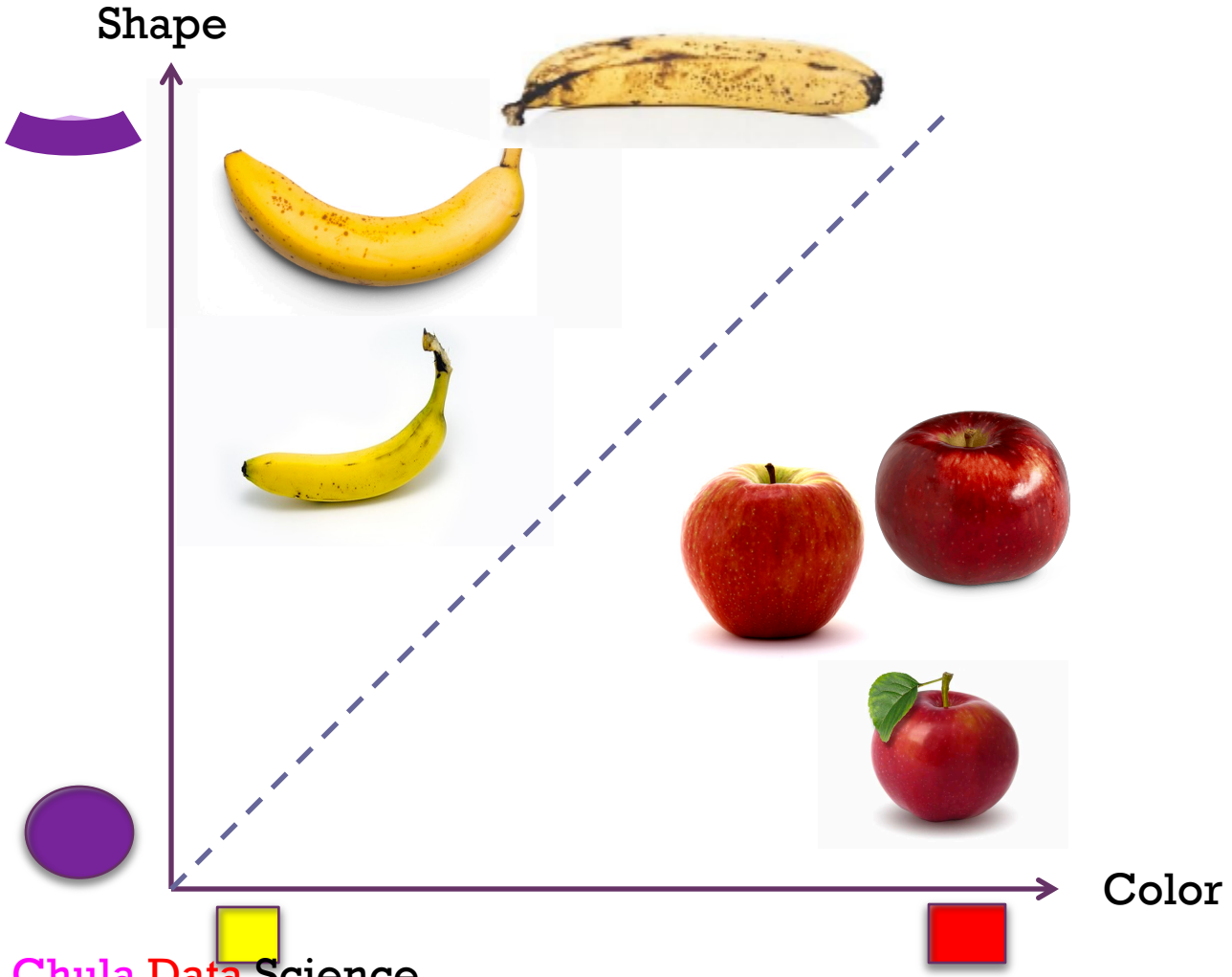
Supervised learning (cont.):
Testing Phase: case 1



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

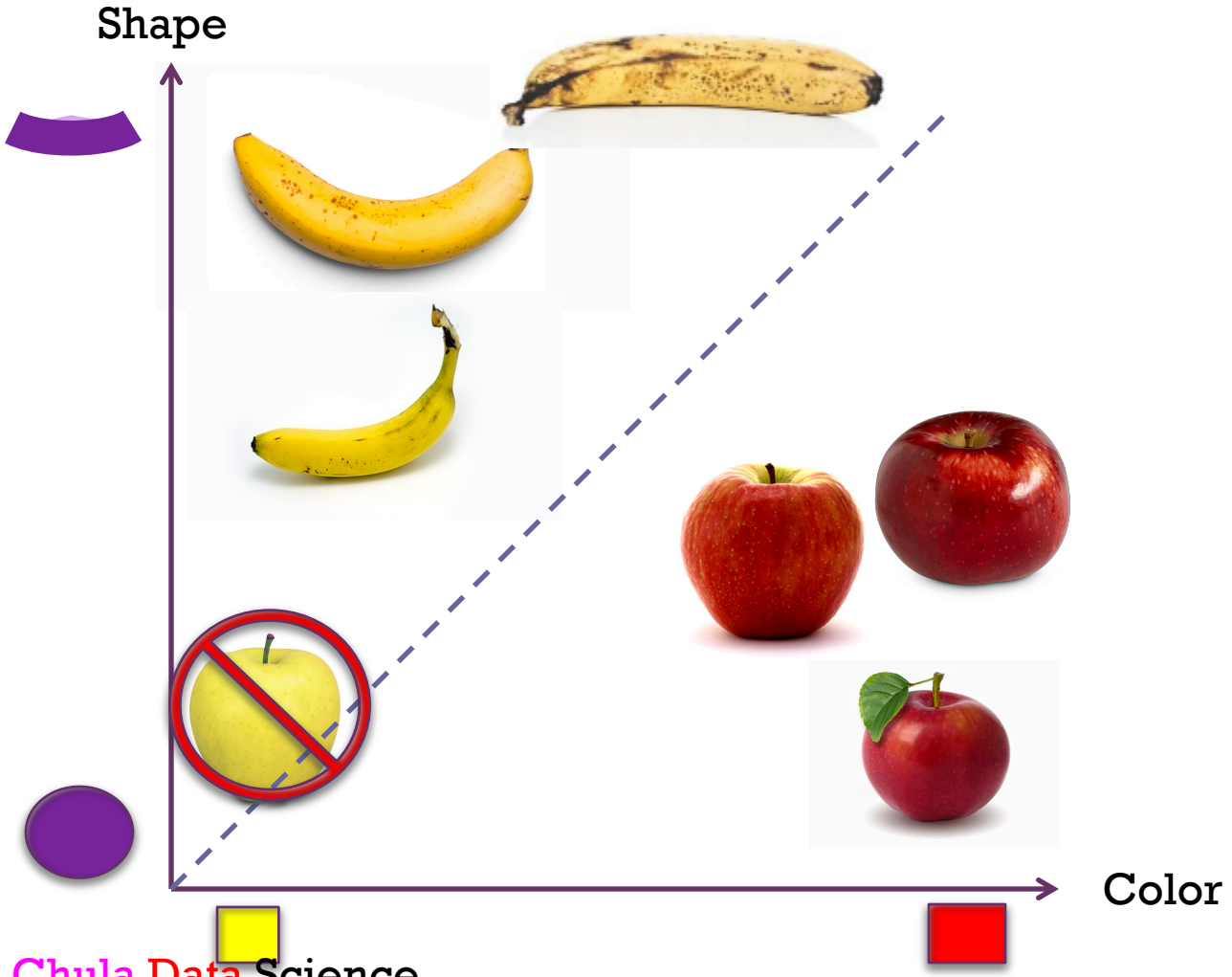
Supervised learning (cont.):
Testing Phase: case2



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

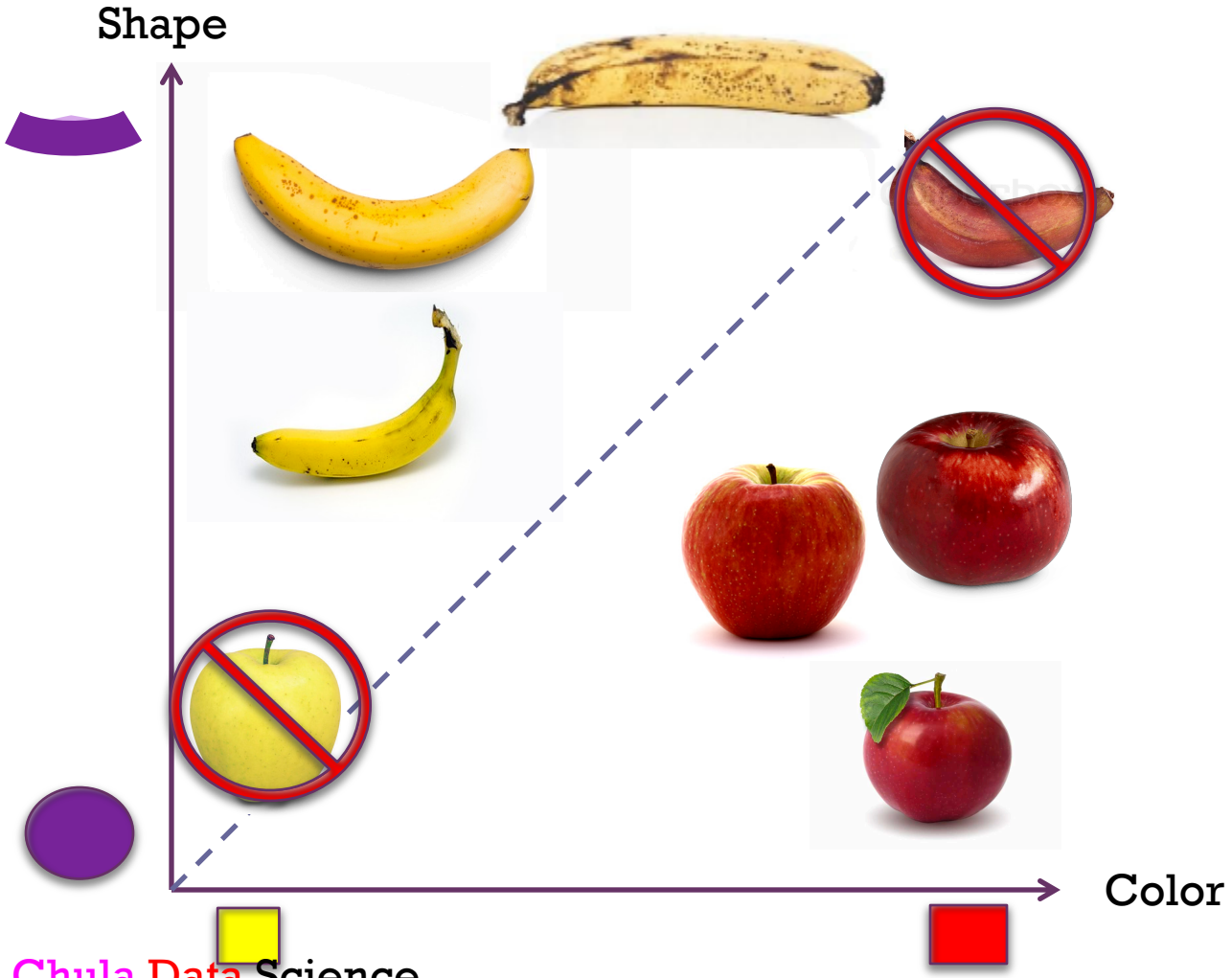
Supervised learning (cont.):
Testing Phase: case3



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

Supervised learning (cont.):
Testing Phase: case4



Supervised Learning

- Learn from labeled examples
- Example: Learn to classify apples from bananas
- Model: Criteria for classification

Supervised learning (recap)

Training Data

inputs

target

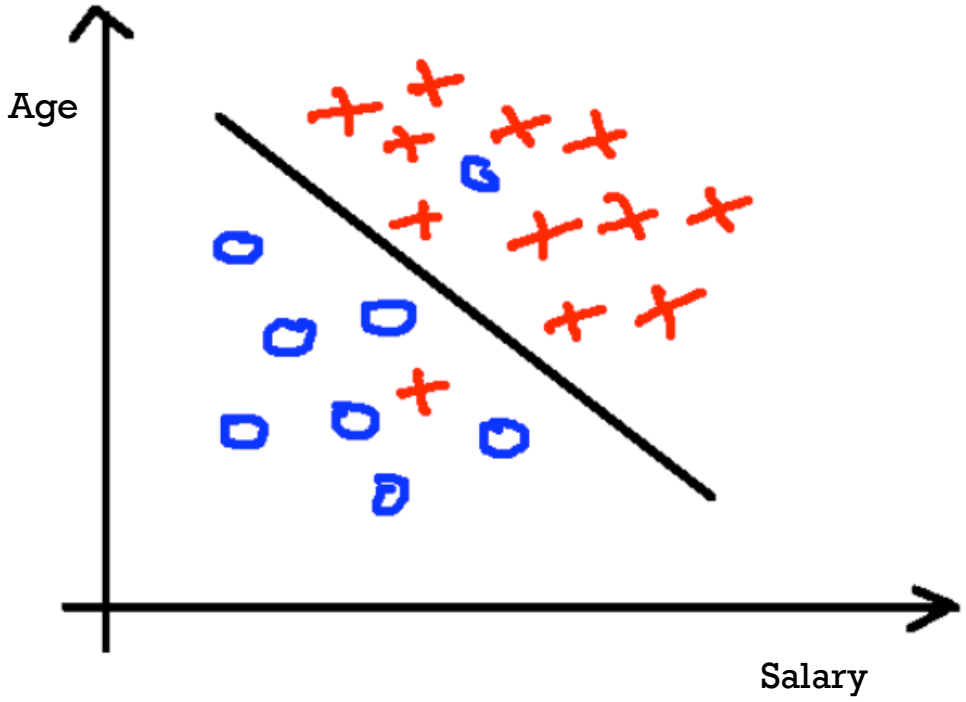
Age	Income	Gender	Province	Purchase
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	No

Testing Data

Age	Income	Gender	Province	Purchase
25	25,000	Female	Bangkok	?

Application: Direct Target Customer

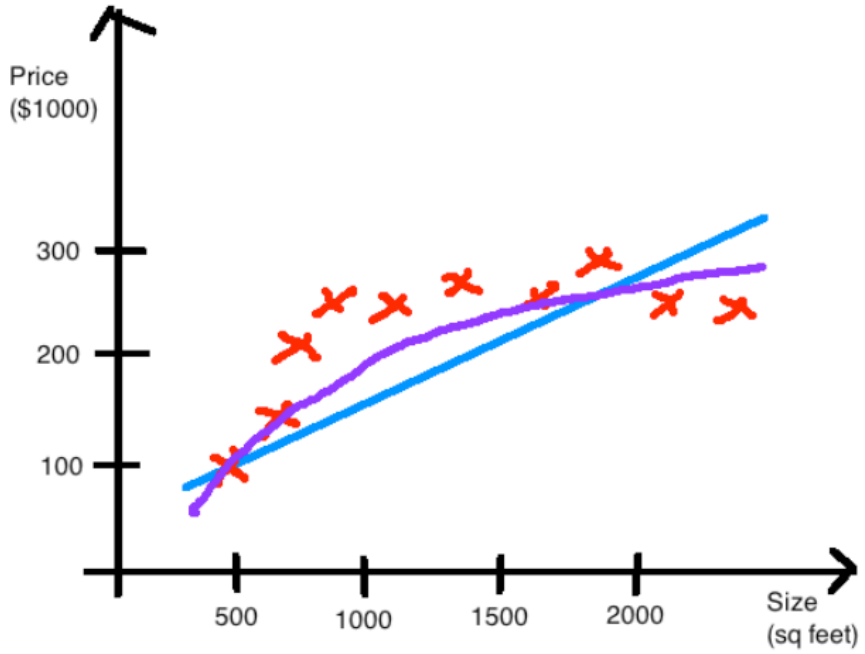
Classification: Predicting a categorical target



Predict targeted customers who tend to buy our product (yes/no)

- **Sample Applications**
  - Database marketing
  - Fraud detection
  - Pattern detection
  - Churn customer detection

Regression: Predicting a continuous target



Predict a sale price of each house

- **Sample Applications**
  - Financial risk management
  - Revenue forecasting
  - Loss reserving

inputs		target
Gender	**Province**	**Amount**
Female	Bangkok	$7,800
Female	Nontaburi	$500
Male	Bangkok	$12,000

Prediction algorithms

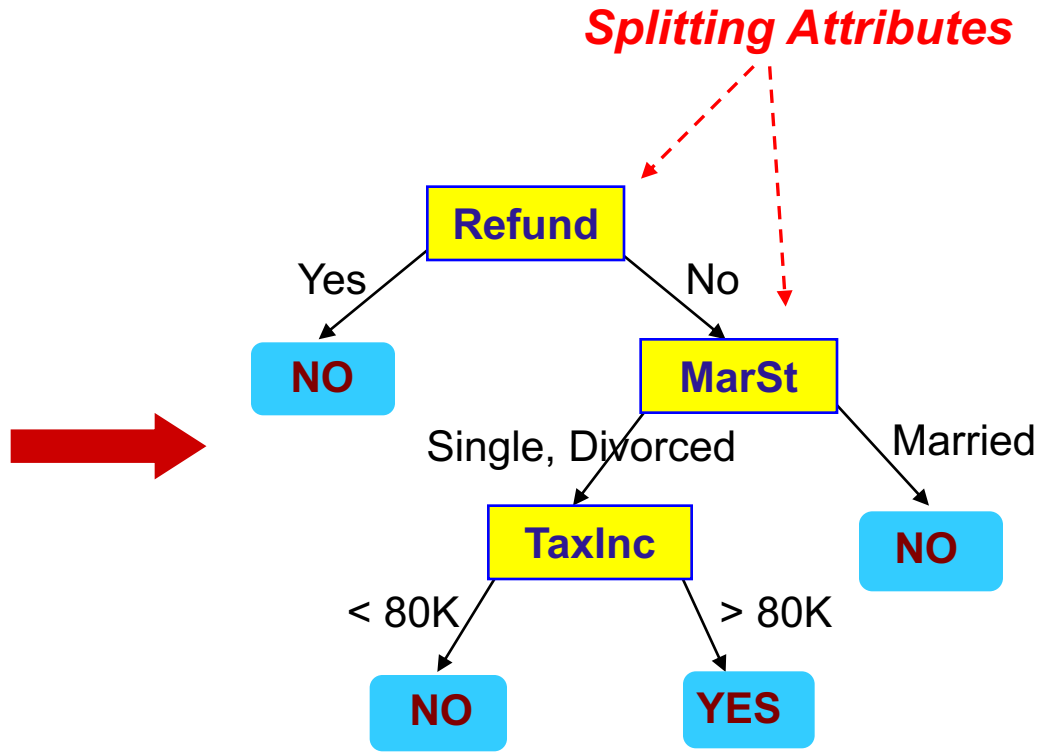
- Decision Tree
- (Logistic) Regression
- kNN
- Support Vector Machine
- Neural Networks (NN)
- Deep Learning

Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

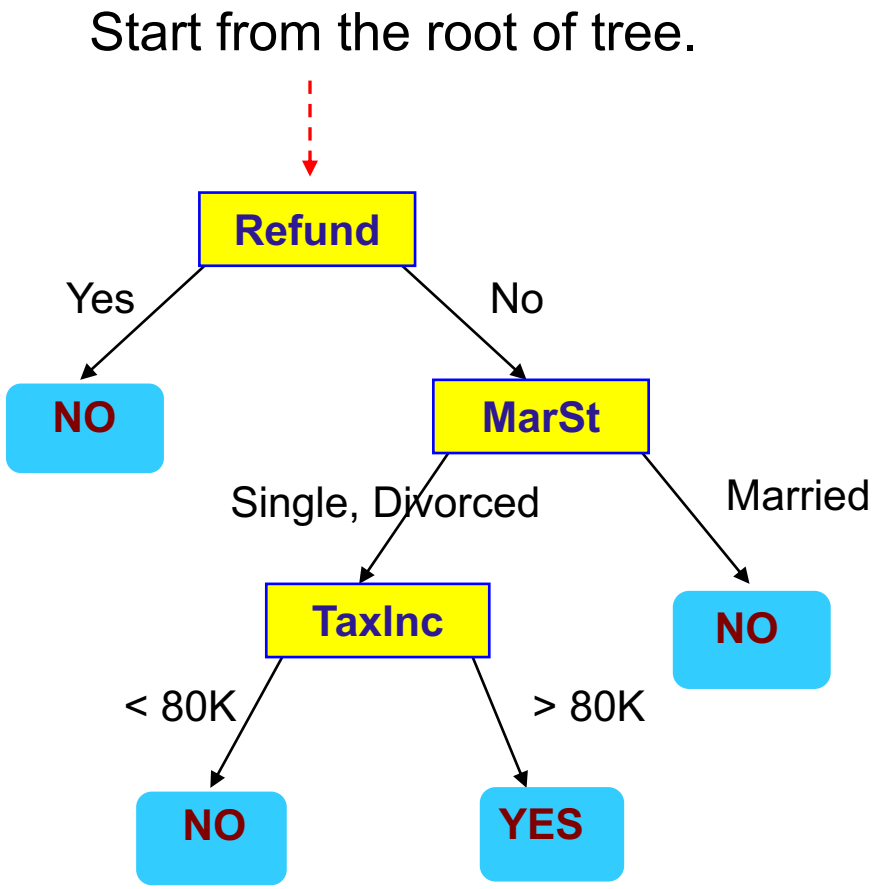
categorical categorical continuous class

Training Data



Model: Decision Tree

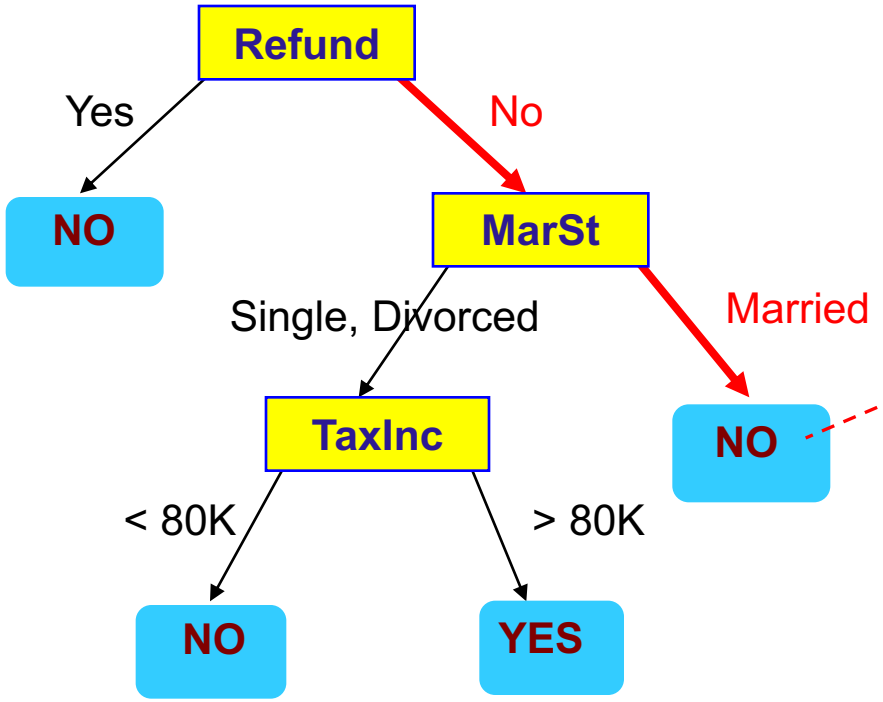
Decision Tree (cont.)



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Decision Tree (cont.)

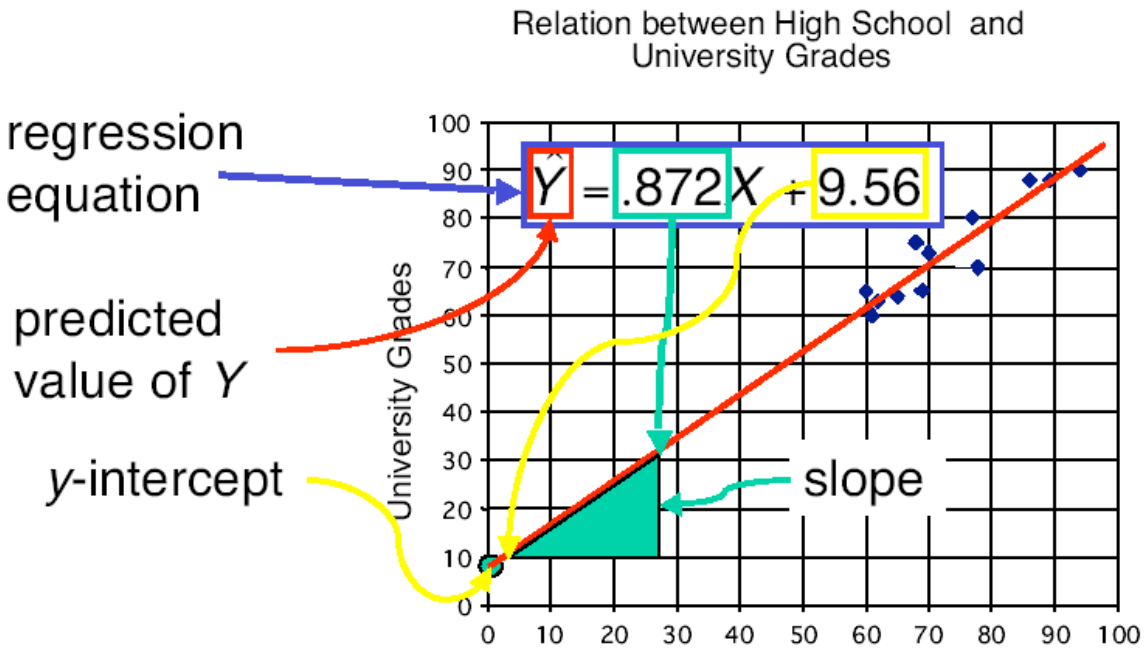


Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

Regression



weight, coefficient

$$\underset{\text{target}}{\hat{y}} = \underset{\text{intercept}}{\hat{w}_0} + \hat{w}_1 x_1 + \hat{w}_2 x_2$$

input

- The least square method aims to minimize the following term

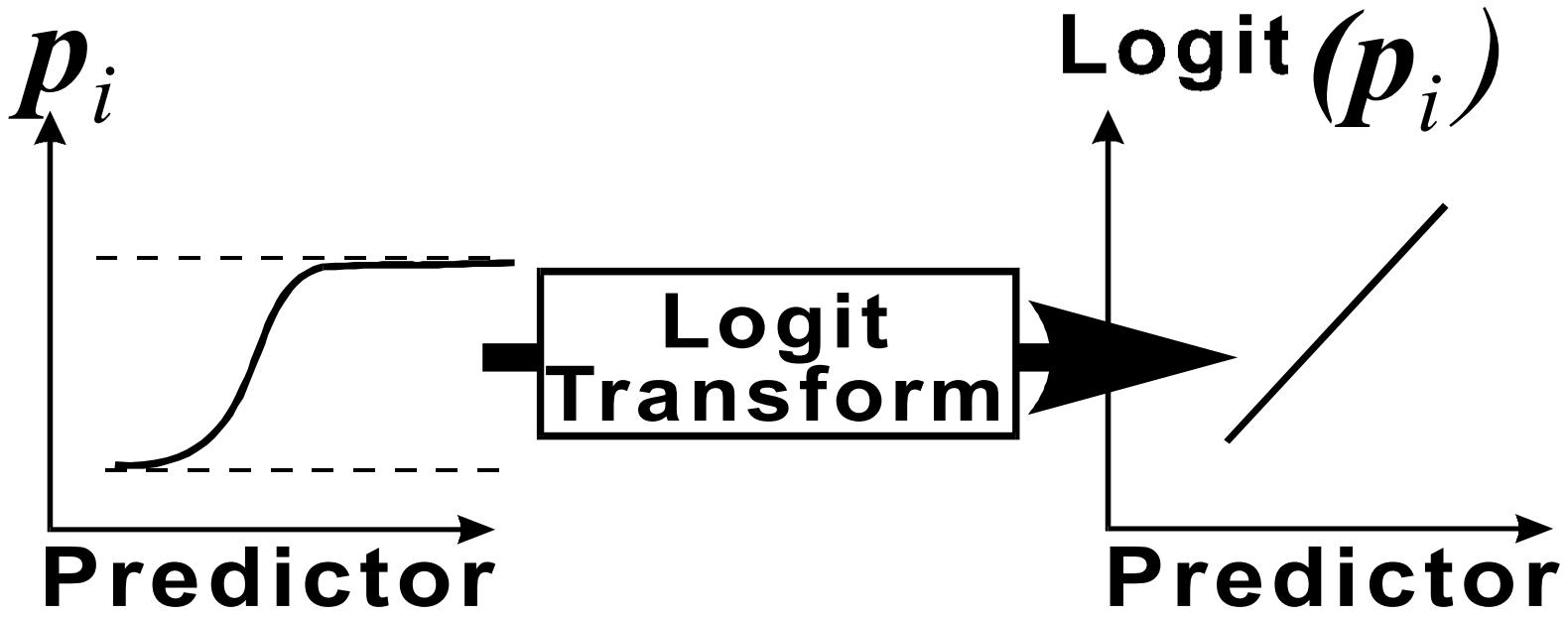
$$\sum_{\substack{training \\ data}} (y_i - \hat{y}_i)^2$$

Logistic regression

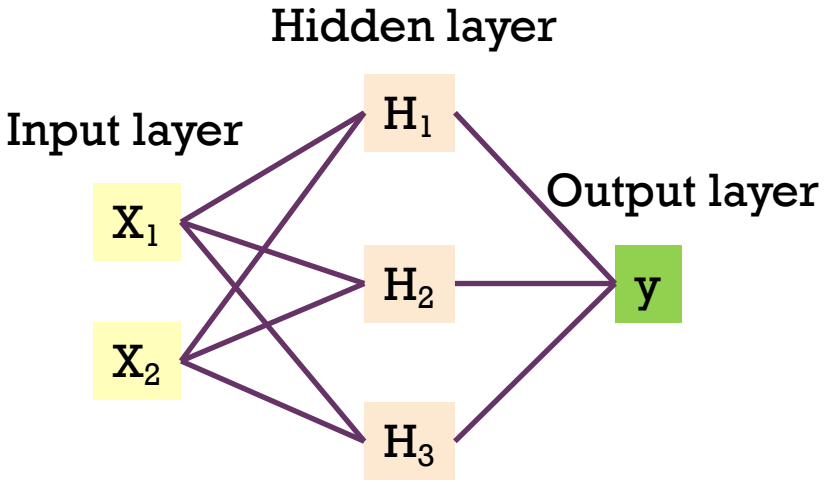
$$log\left(\frac{\hat{p}}{1-\hat{p}}\right) = \hat{w}_0 + \hat{w}_1 x_1 + \hat{w}_2 x_2 = logit(\hat{p})$$

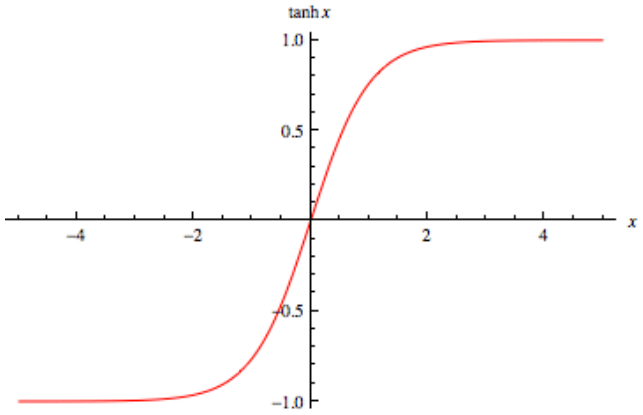
$$\hat{p} = \frac{1}{1 + e^{logit(\hat{p})}}$$

- Maximum likelihood estimates



Neural Networks (universal approximator)





$$log\left(\frac{\hat{p}}{1-\hat{p}}\right) = \hat{w}_0 + \hat{w}_1 H_1 + \hat{w}_2 H_2 + \hat{w}_3 H_3$$

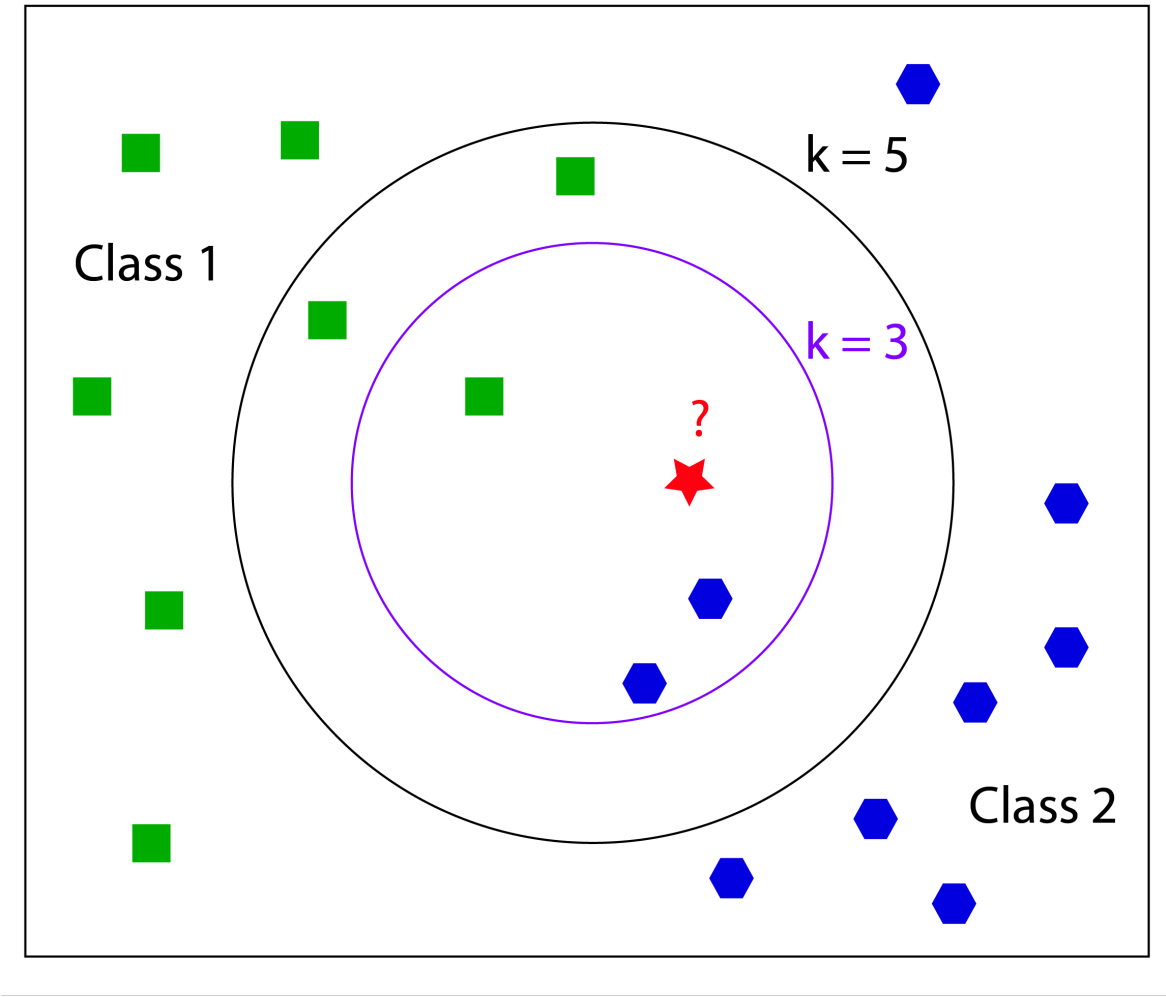
$$H_1 = tanh(\hat{w}_{10} + \hat{w}_{11} x_1 + \hat{w}_{12} x_2)$$

$$H_2 = tanh(\hat{w}_{20} + \hat{w}_{21} x_1 + \hat{w}_{22} x_2)$$

$$H_3 = tanh(\hat{w}_{30} + \hat{w}_{31} x_1 + \hat{w}_{32} x_2)$$

k-Nearest Neighbors (kNN)

- Memory based learning
- Suitable for small data sets
- Merge
  - Voting
  - Average
  - Maximum prob

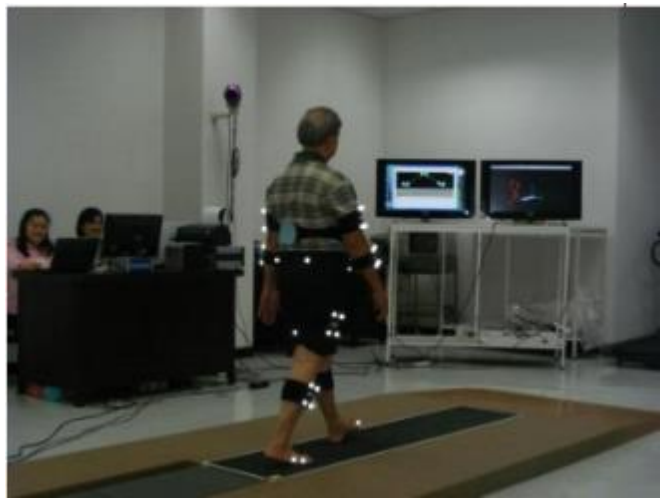


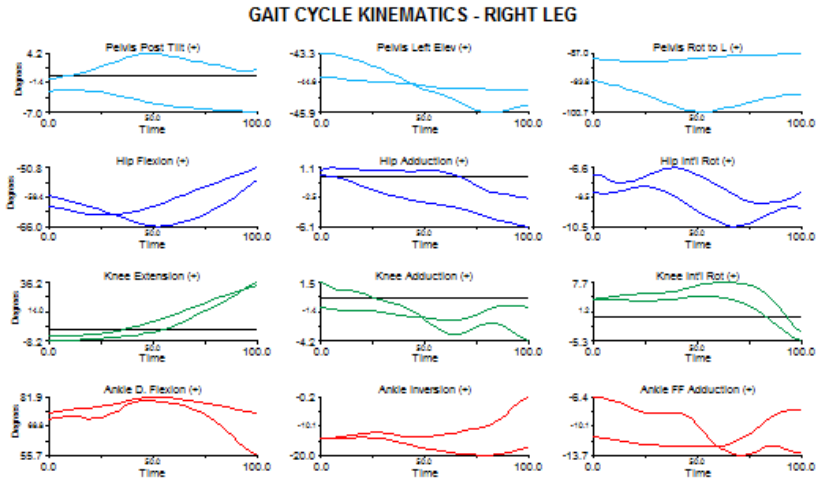
Model Evaluation

- Estimate
  - Sum Squared Error (SSE)
  - Average Squared Error (ASE)
- Decision
  - **Accuracy**
  - Misclassification
  - Precision
  - Recall
  - F1
- Ranking
  - ROC Curve
  - Area Under ROC (c-statistic)
  - **Lift**
  - Gain
  - Response

Case Study: Automated Medical Diagnosis on Movement Disorder Using Gait Data

- This work proposed an automated medical diagnosis in order to classify patients into three classes:
  - Normal, Sick/Knee OA, Sick/Parkinson.





Task2: Unsupervised learning (descriptive task)

1. Clustering
2. Association Rule Mining

Training Data

inputs

target

Age	Income	Gender	Province	Purchase
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	

Testing Data

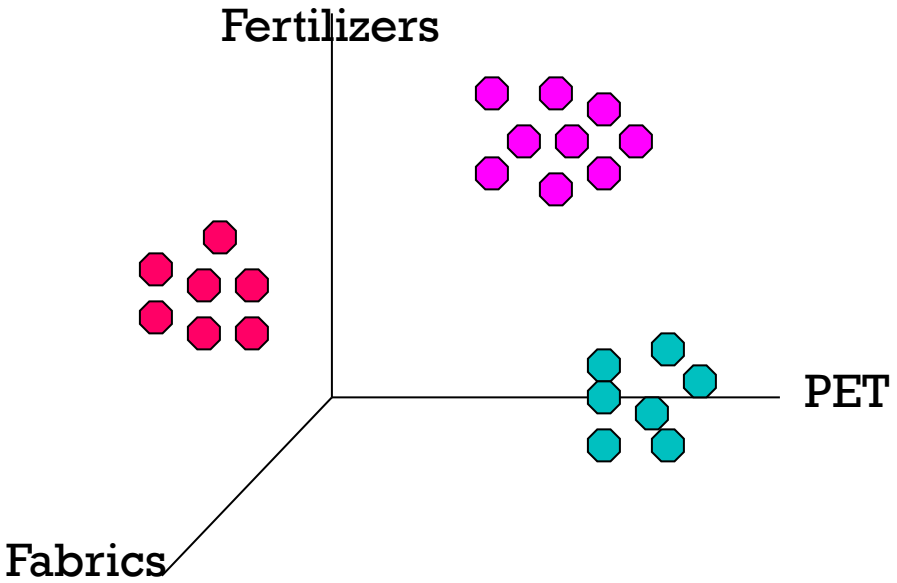
Age	Income		Province	Purchase
25	25,000		Bangkok	?

Clustering

- In our class, there are many participants. Should we teach them using the same method?
- May be not! Since they may have different learning behaviors and backgrounds.
- Inputs
  - Education field
  - Level of English communication
  - Level of computer skills
  - Age range
  - Gender

Clustering (cont.)

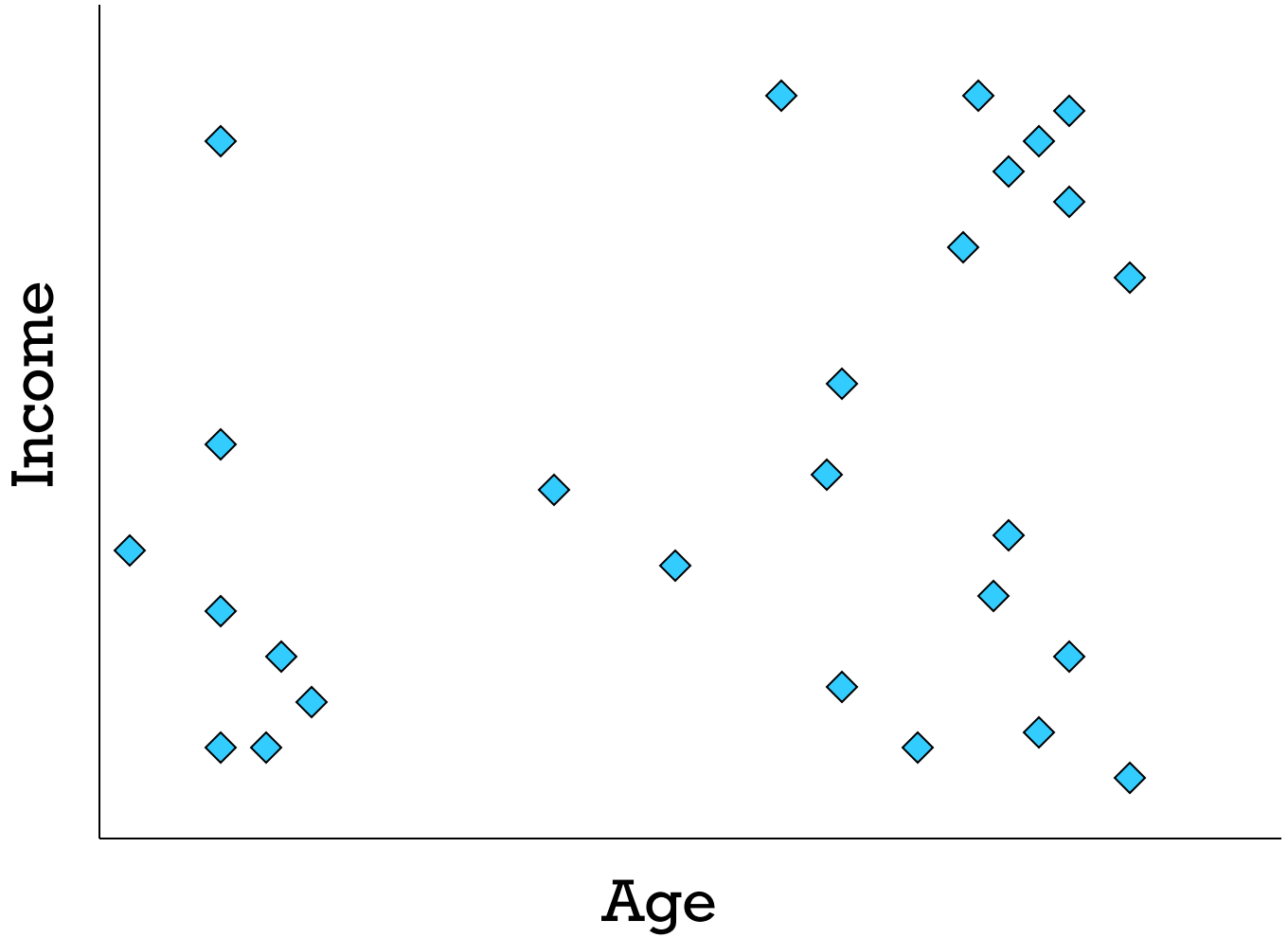
Company	Sedan	Truck	Motorcycles
C1	70M	2M	80M
C2	90M	120M	100M
C3	1M	8M	70M



- **Some techniques:**
  - k-means
  - DB-scan
  - Hierarchical clustering

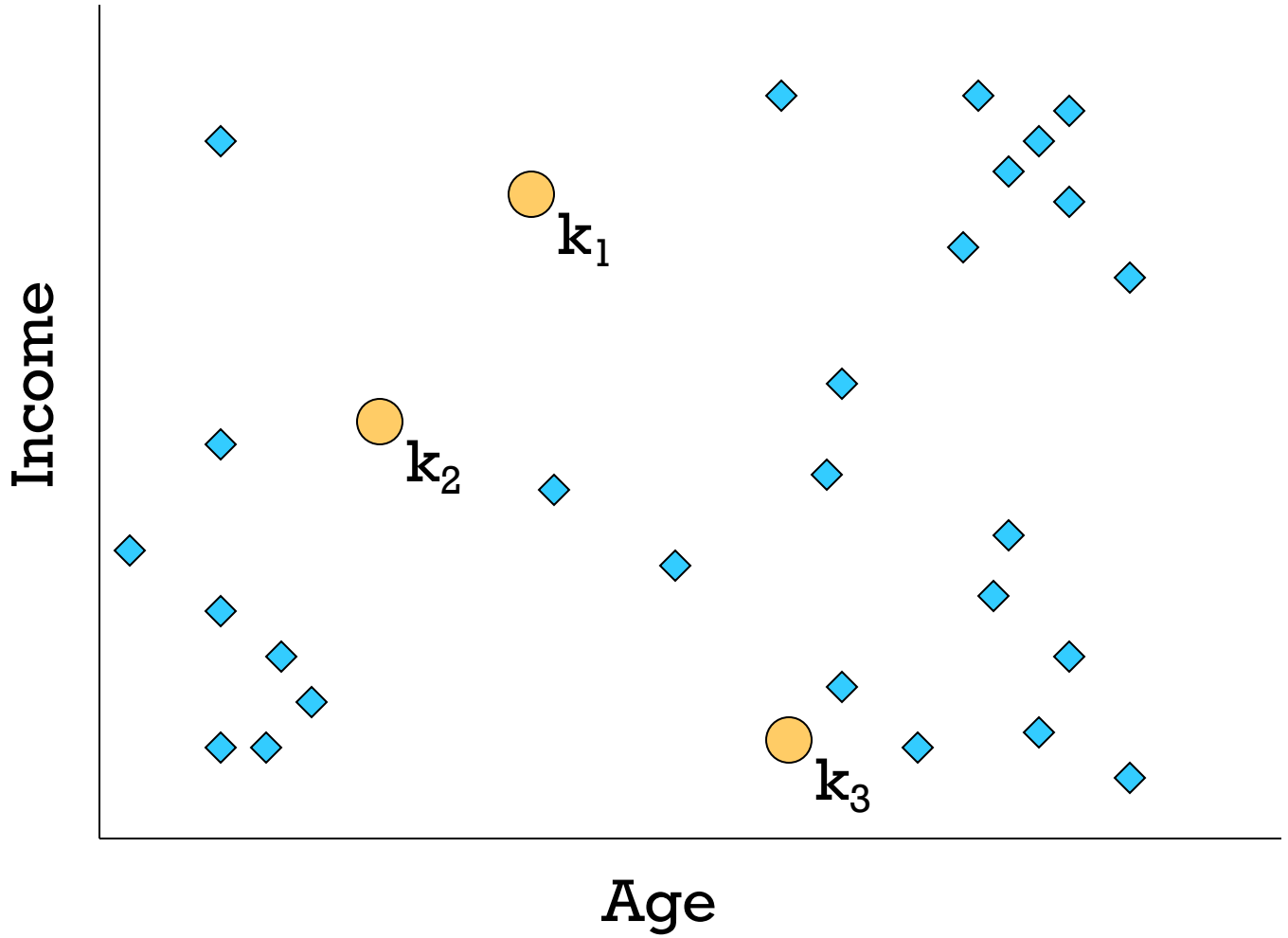
Example: Customer Segmentation

K-means: Step0



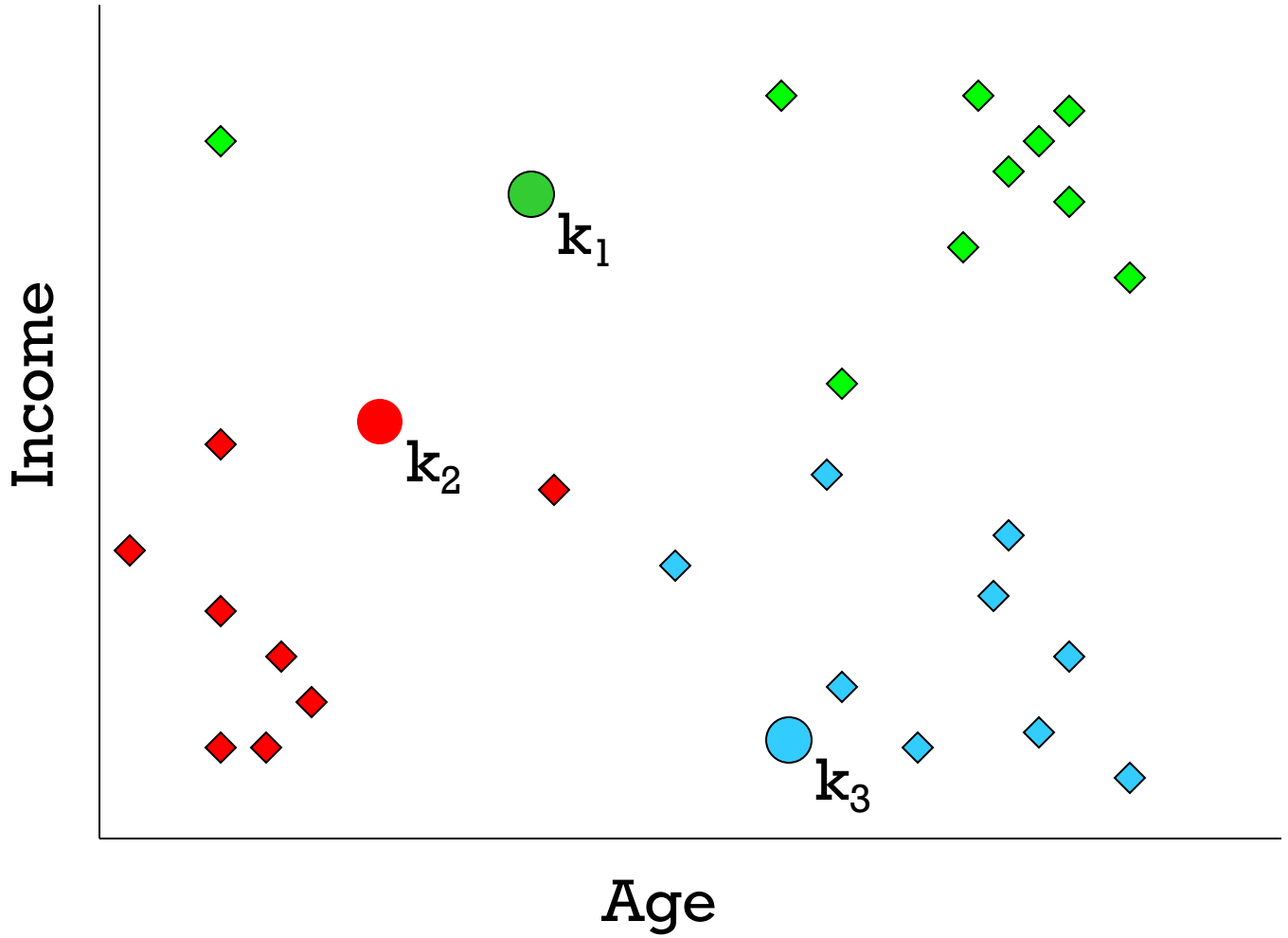
K-means: Step 1

Pick 3 initial cluster centers (randomly)



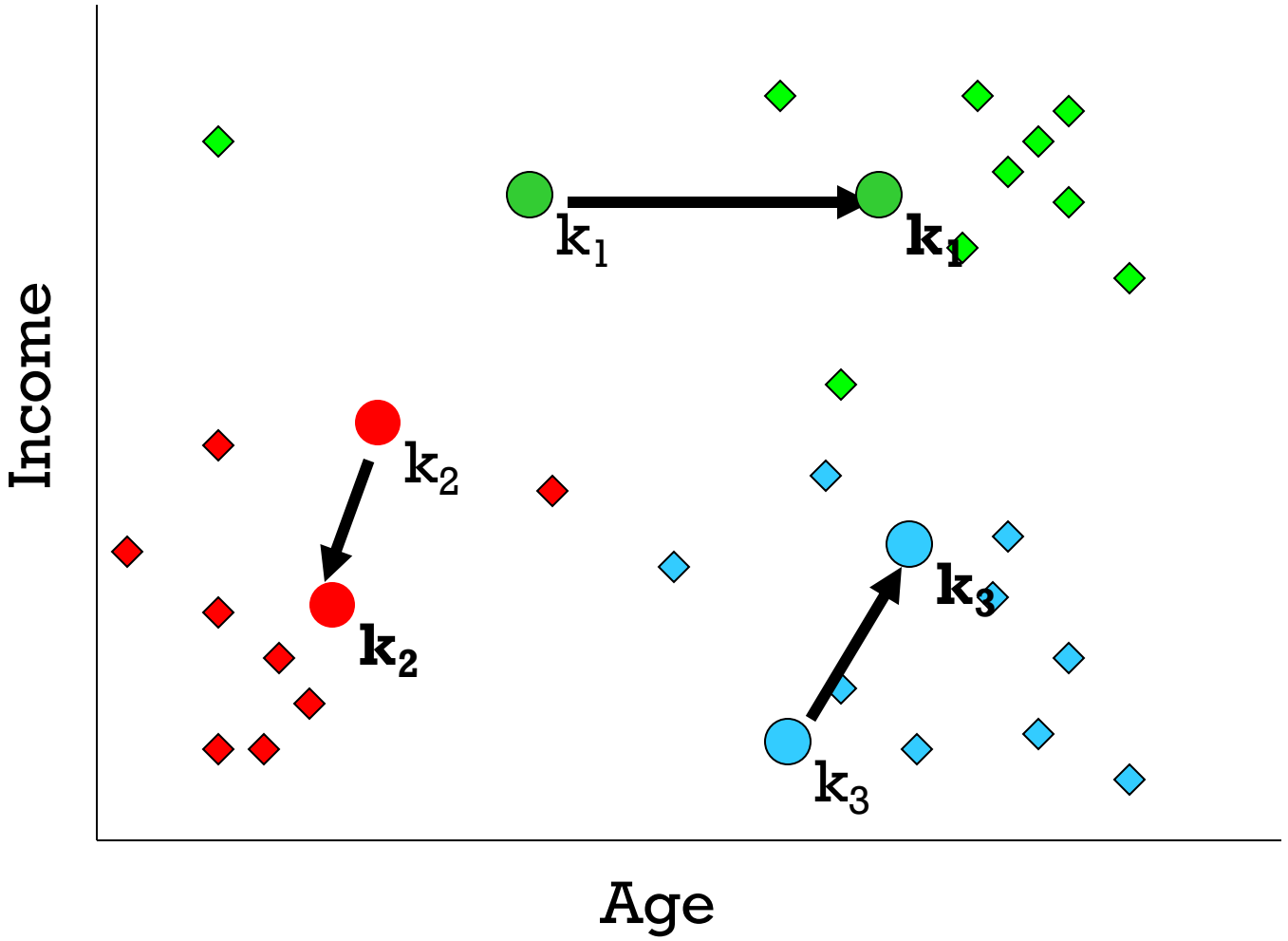
K-means: Step2

Assign each point to the closest cluster center



K-means: Step3

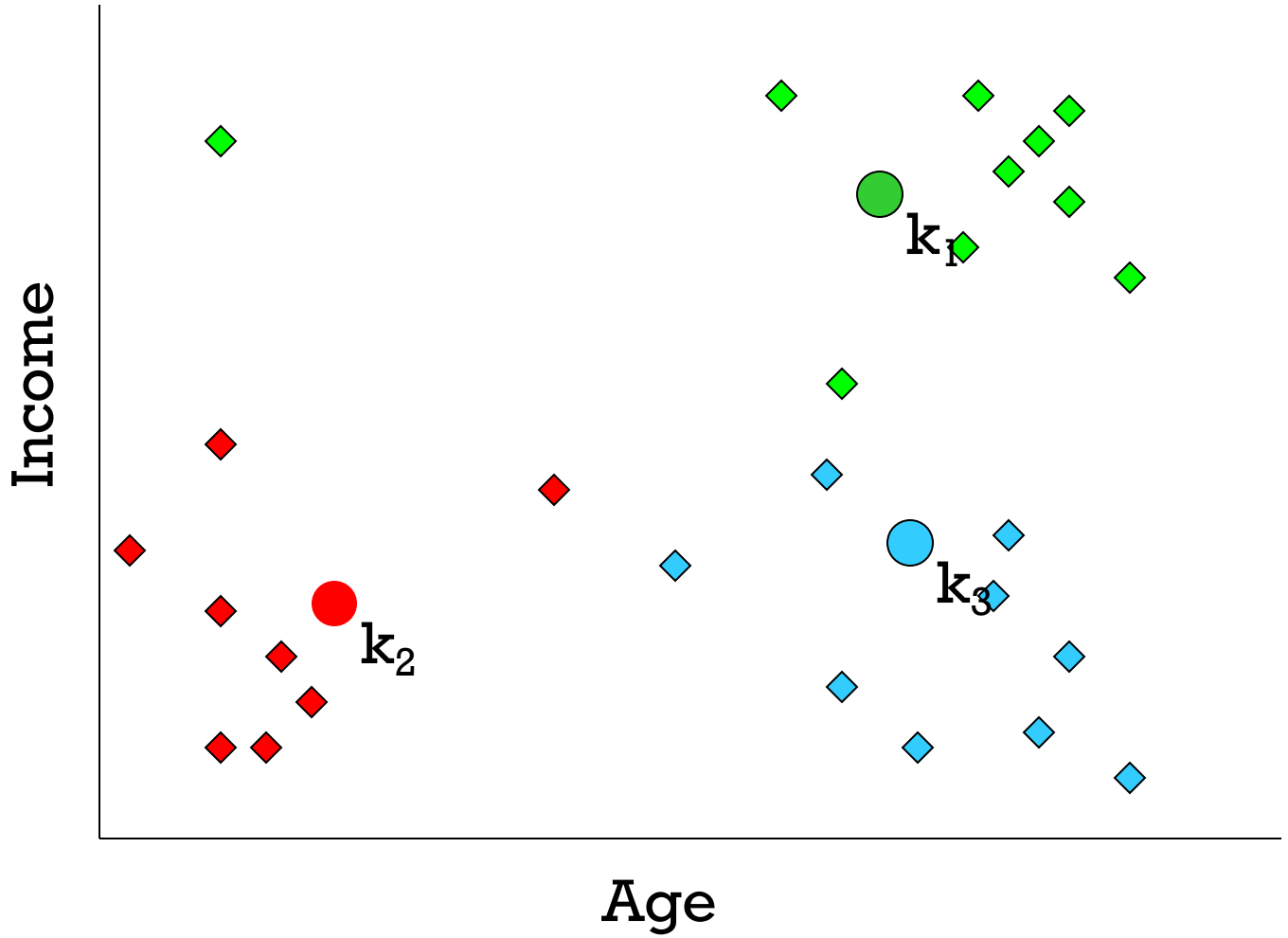
Move
each cluster
center
to the mean
of each cluster



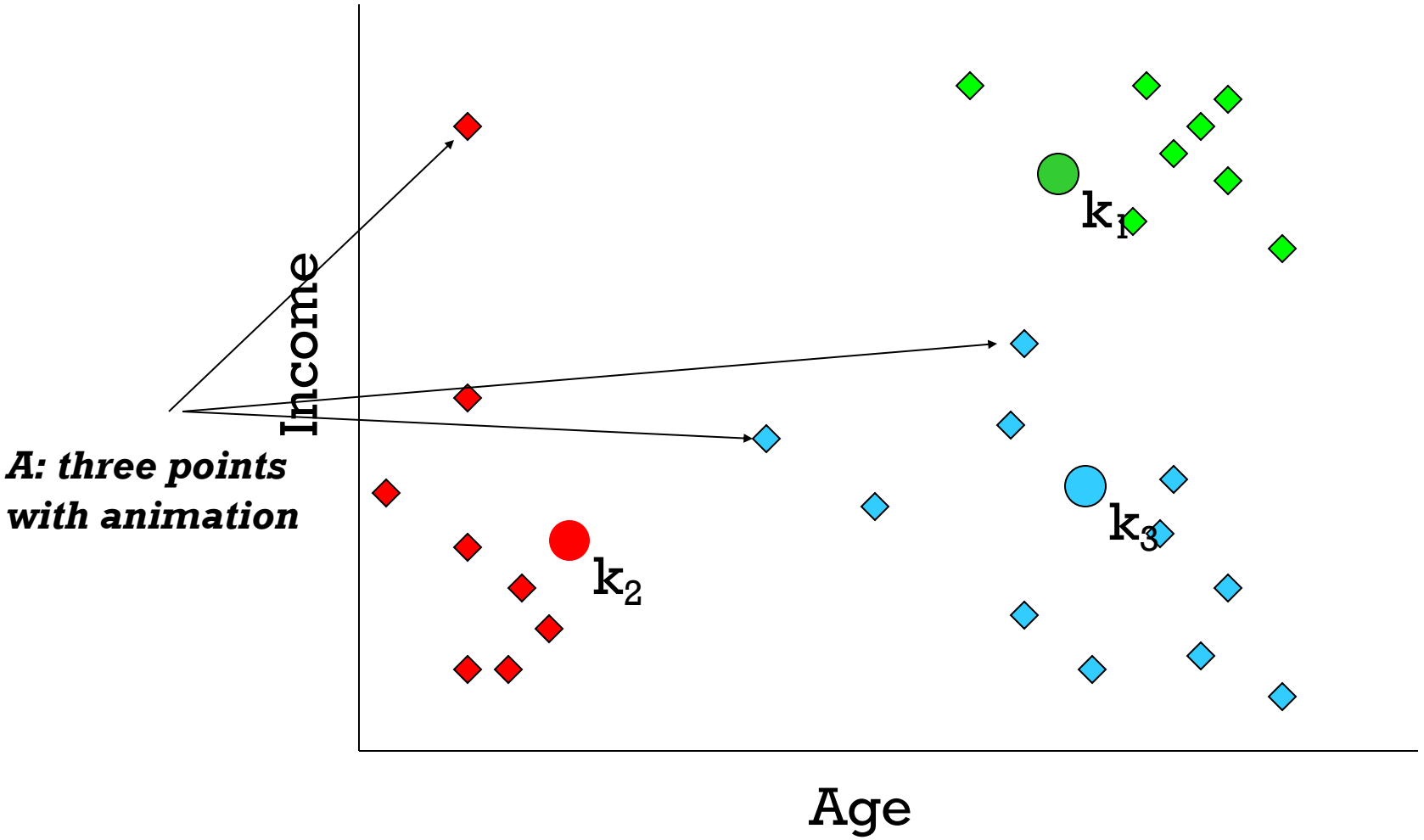
K-means: Step4

Reassign points closest to a different new cluster center

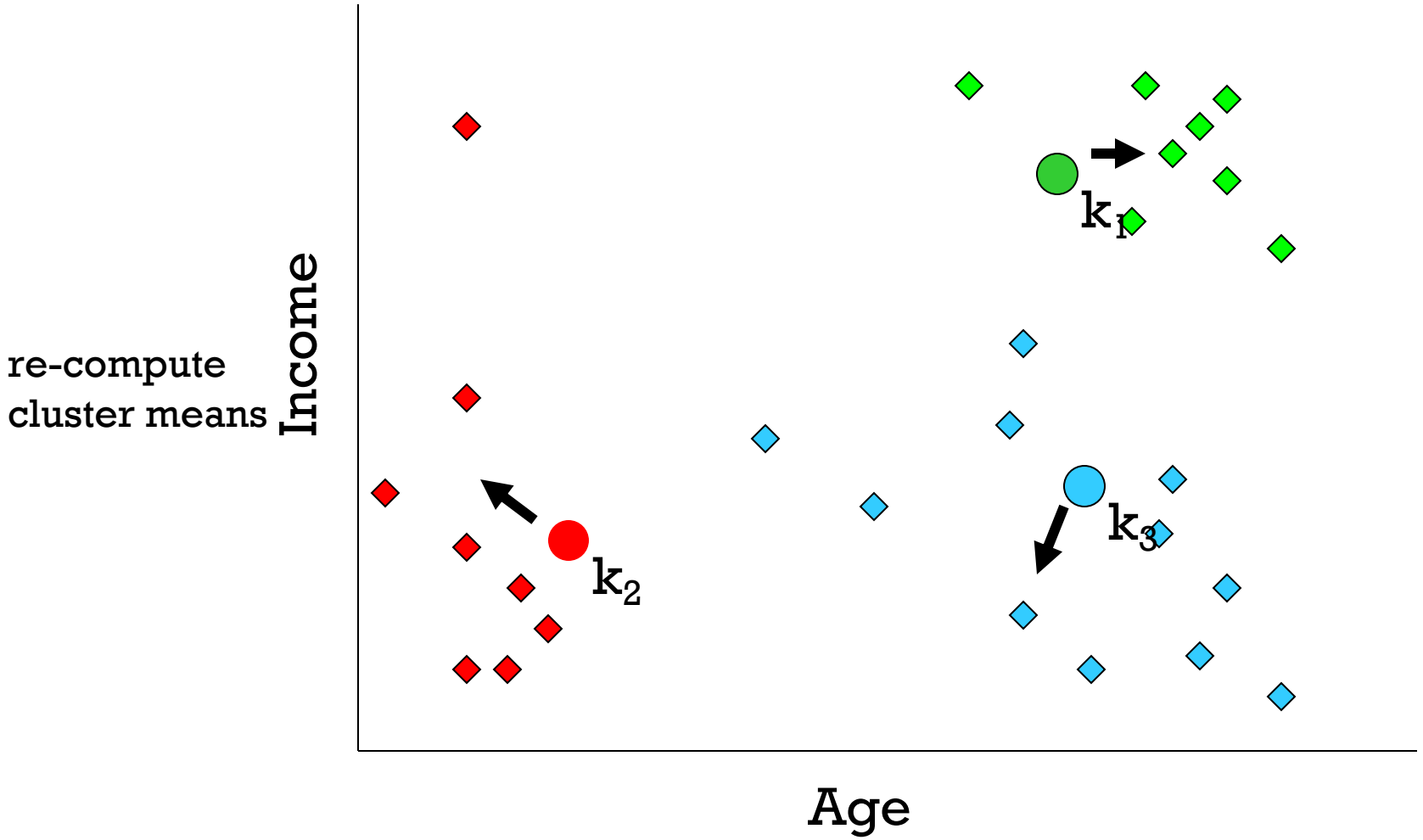
Q: Which points are reassigned?



K-means: Step4(a)

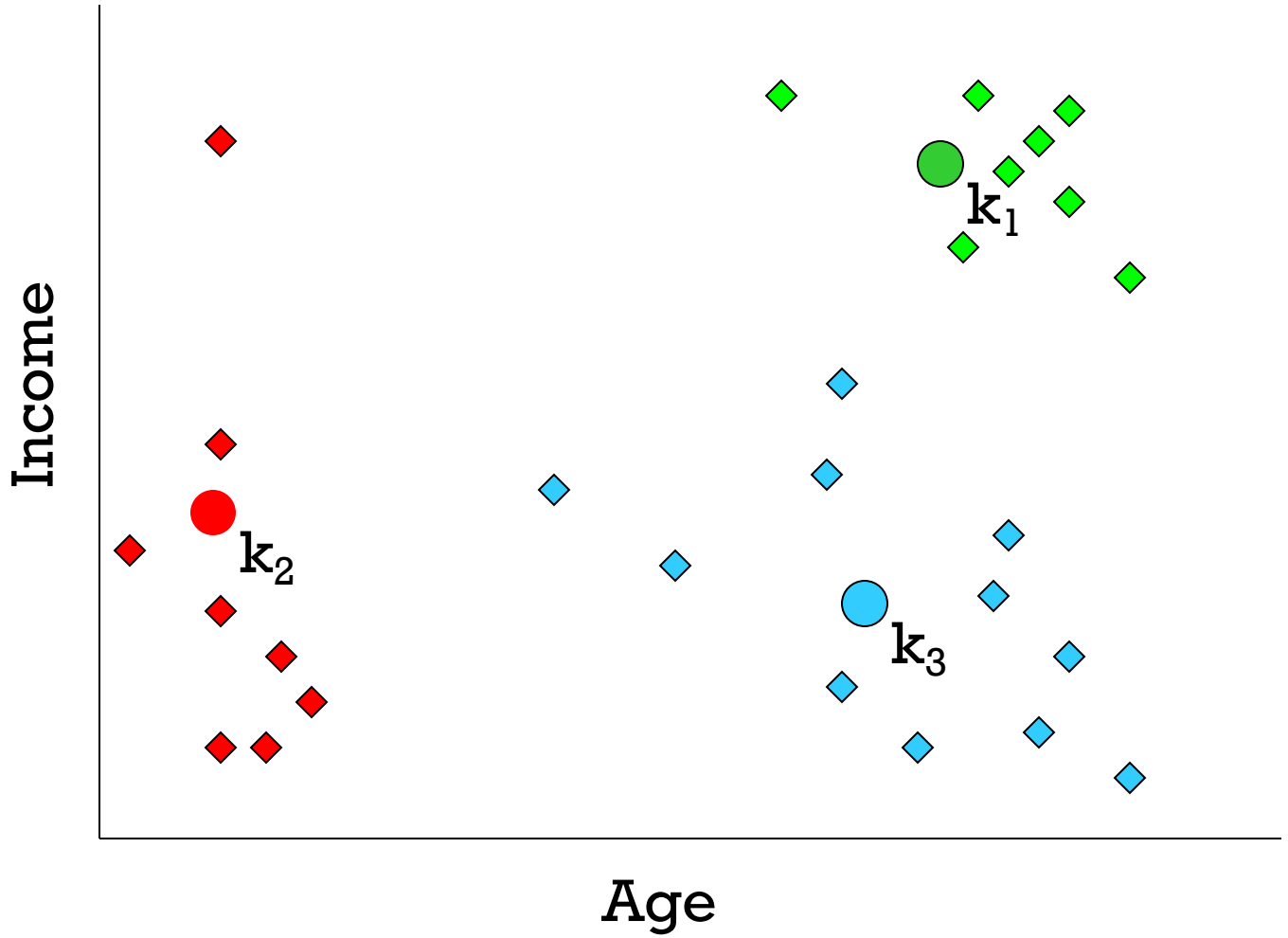


K-means: Step5



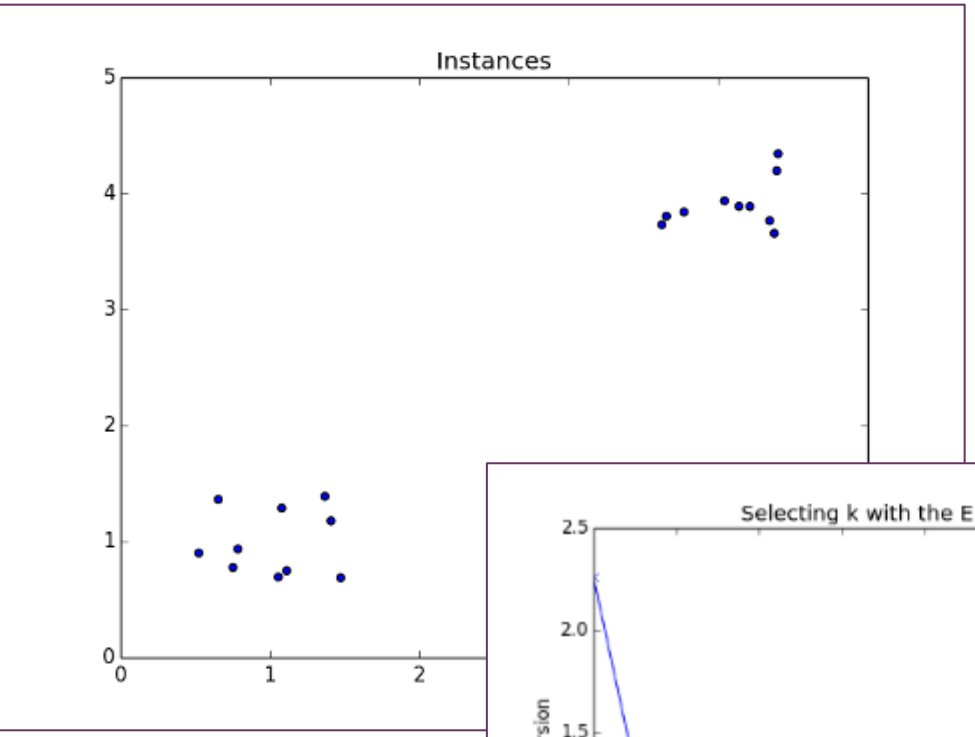
K-means: Step5(a)

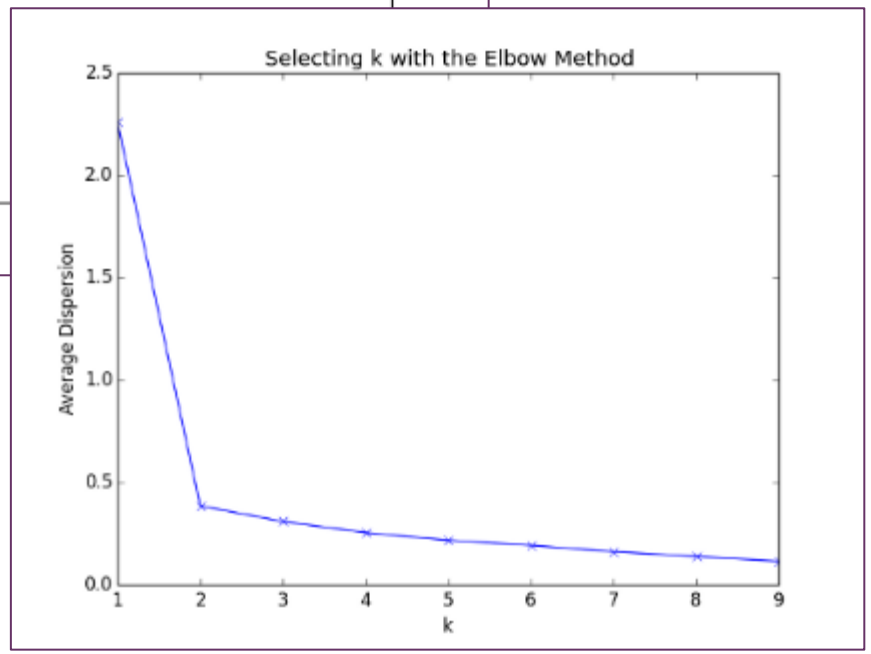
move cluster centers to cluster means

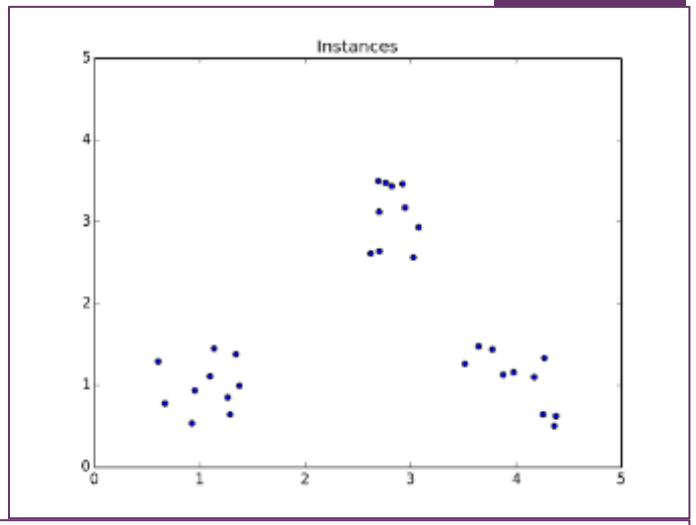


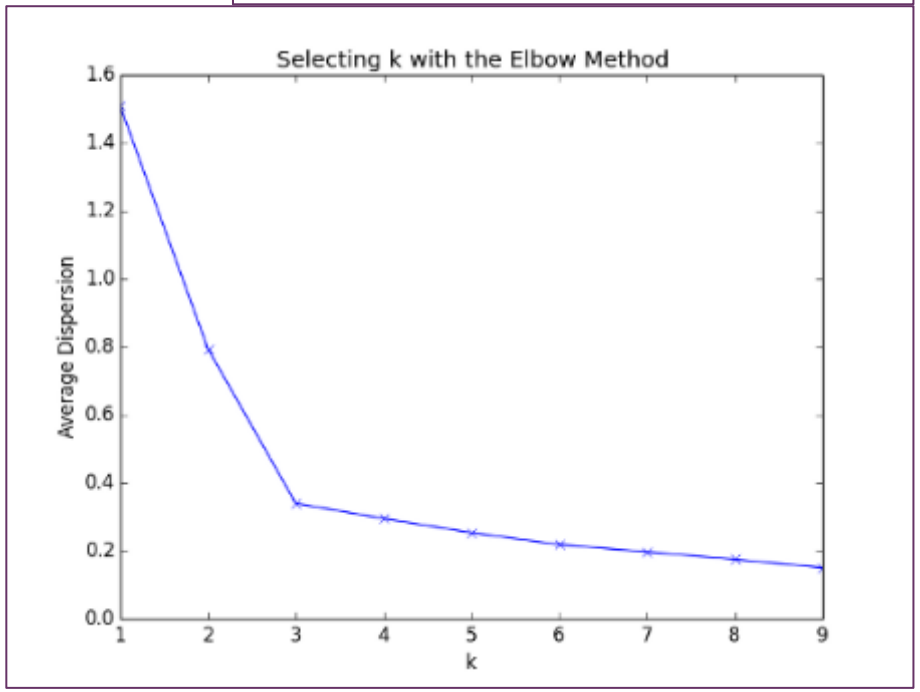
Determine the number of *k*: Elbow Method

https://www.packtpub.com/books/content/clustering-k-means









Market Basket Analysis

Rule	Support	Confidence
Apple => Donut	2/5	2/3
Coconut > Apple	2/5	2/4
Apple => Coconut	2/5	2/3
Banana & Coconut => Donut	1/5	1/3

Implication

	No	Yes	Pasta Total
No	500	3,500	**4,000**
Yes	1,000	5,000	**6,000**
Pizza Total	**1,500**	**8,500**	**10,000**

- Support(Pizza => Pasta) = 50%
- Confidence(Pizza => Pasta) = 83%
- Expected Confidence(Pizza => Pasta) = 85%
- **Lift(Pizza => Pasta) = 83%/85% < 1**

Association Rule Mining (cont.)

- Wal-Mart customers who purchase Barbie dolls have a 60% likelihood of also purchasing one of three types of candy bars [Forbes, Sept 8, 1997]

- Strategies?
  1. Put them closer together in the store.
  2. Put them far apart in the store.
  3. Package candy bars with the dolls.
  4. Package Barbie + candy + poorly selling item.
  5. Raise the price on one, and lower it on the other.
  6. Offer Barbie accessories for proofs of purchase.
  7. Do not advertise candy and Barbie together.
  8. Offer candies in the shape of a Barbie doll.

Caution in Association Rule Mining

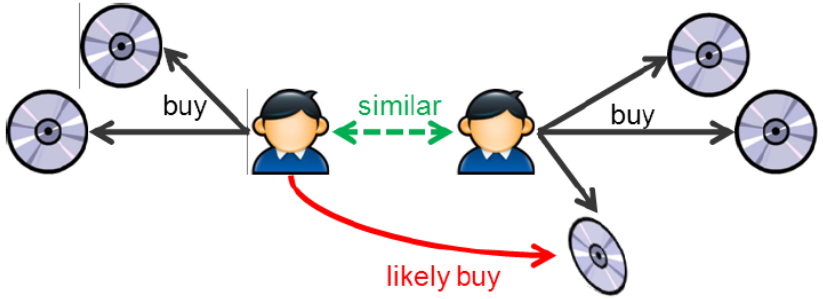
1. • Basket size: per bill, customer, day
2. • Item level: SKU, product category

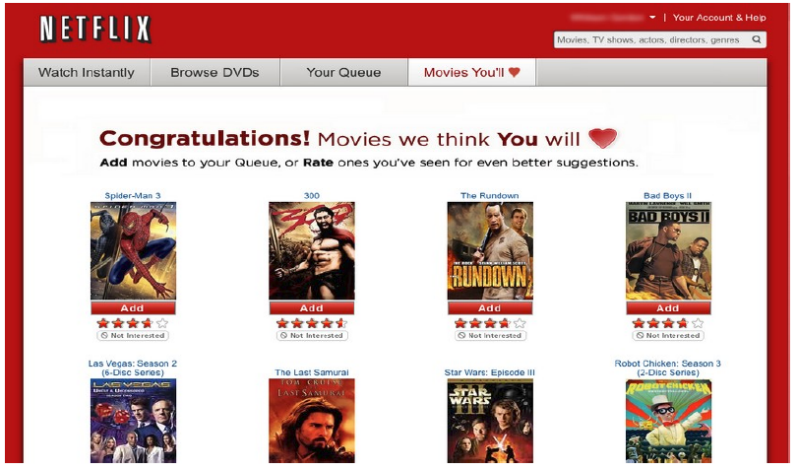
Association Rule Mining (cont.)

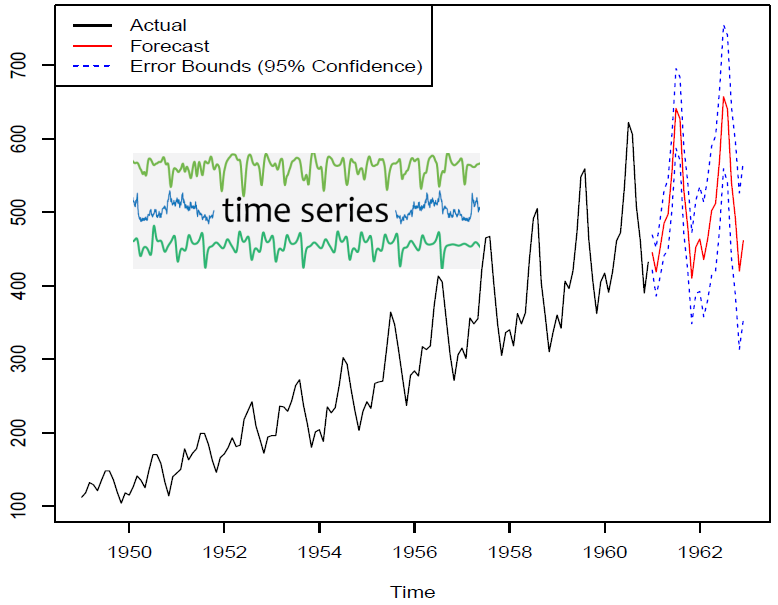
- Wal-Mart customers who purchase Barbie dolls have a 60% likelihood of also purchasing one of three types of candy bars [Forbes, Sept 8, 1997]

- Strategies?
  1. Put them closer together in the store.
  2. Put them far apart in the store.
  3. Package candy bars with the dolls.
  4. Package Barbie + candy + poorly selling item.
  5. Raise the price on one, and lower it on the other.
  6. Offer Barbie accessories for proofs of purchase.
  7. Do not advertise candy and Barbie together.
  8. Offer candies in the shape of a Barbie doll.

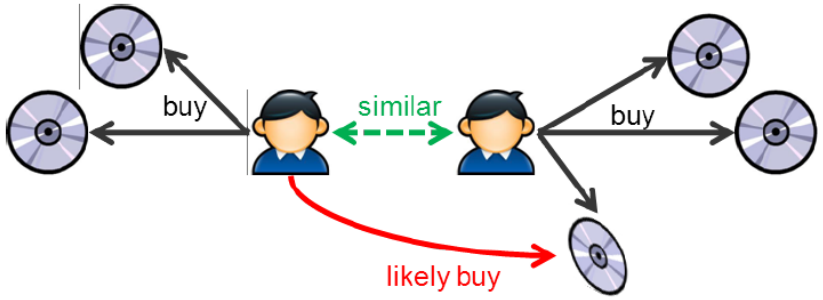
Task3: Special task

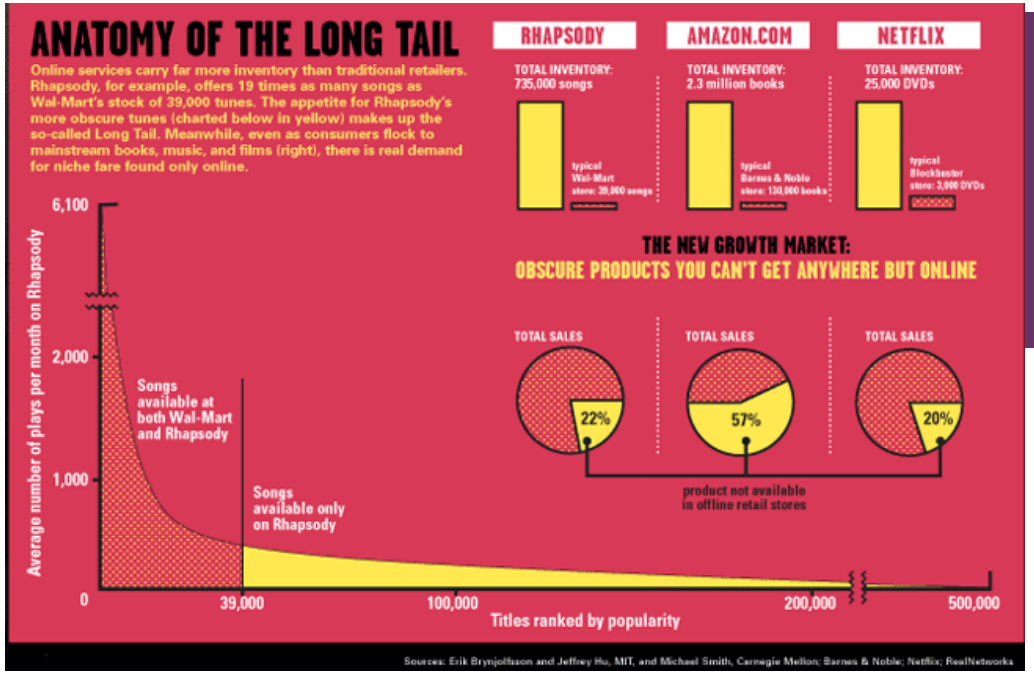


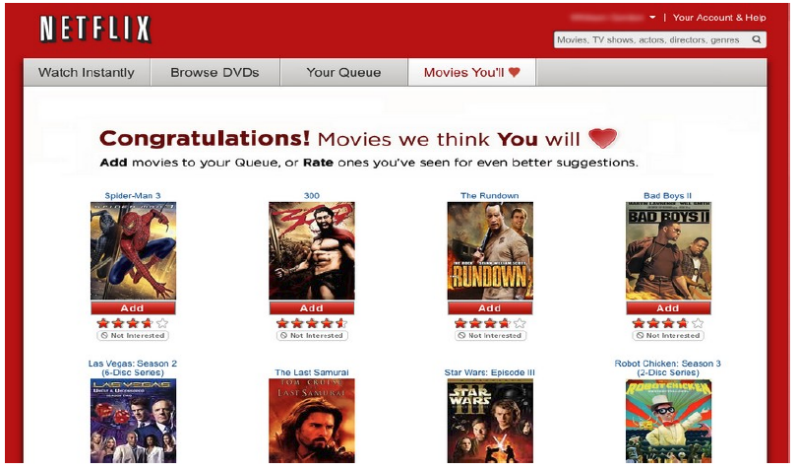




Recommendation system





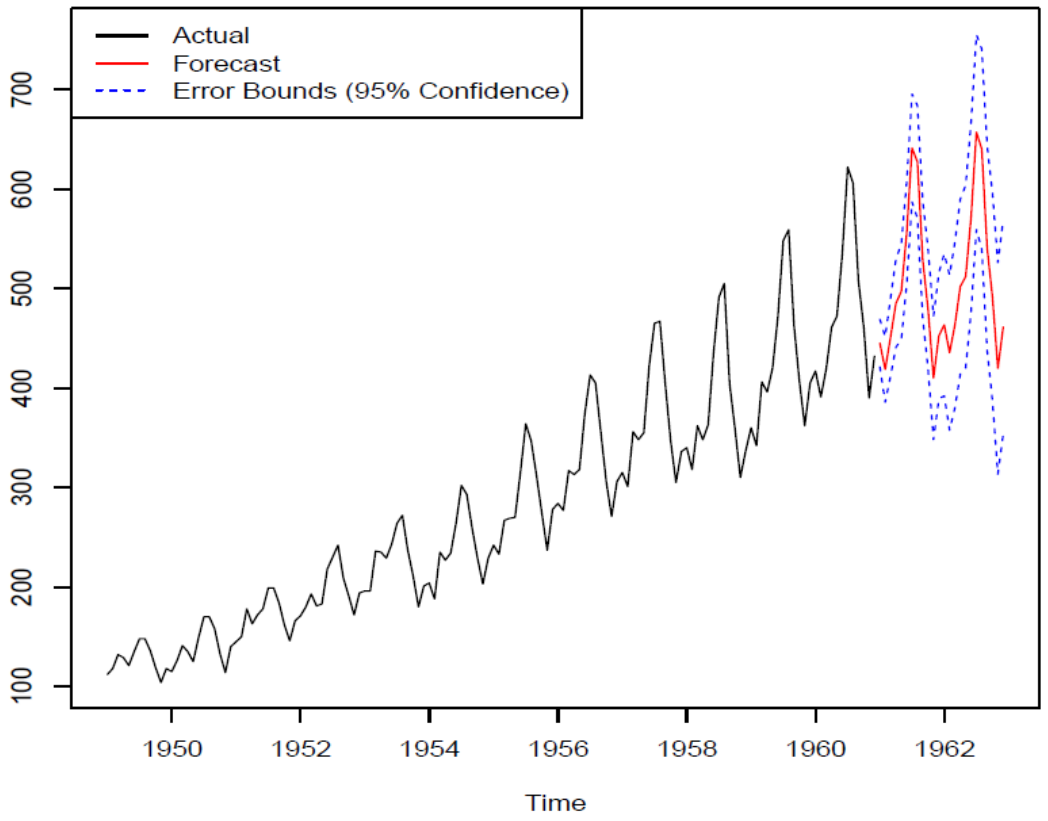


	Harry potter	X-Men	Hobbit	Argo	Pirates
101	5	2	4	?	?
102	?	?	5	2	?
103	1	2	?	?	3
104					
105					
…					

→

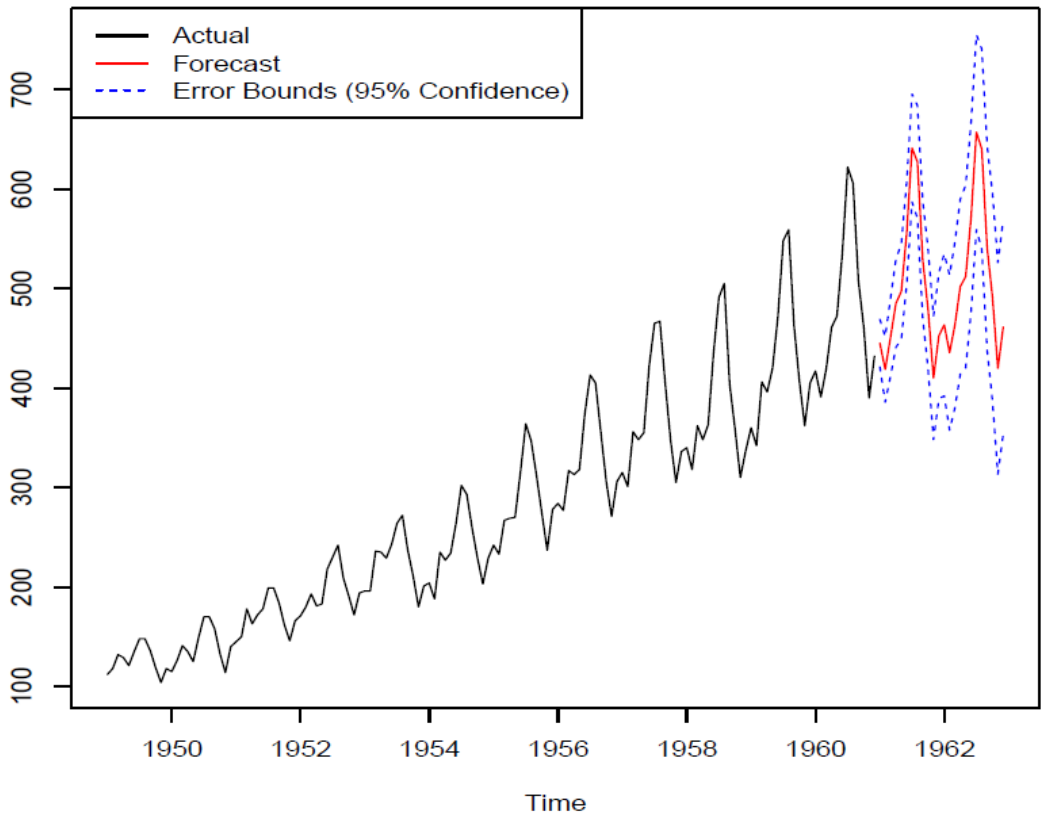
	Harry potter	X-Men	Hobbit	Argo	Pirates
101	5	2	4	1	3
102	4	1	5	2	3
103	1	2	4	1	3
104					
105					
…					

Time Series Analysis (Trend Forecasting)



- Techniques
  - **ARIMA (Autoregressive integrated moving average)**
  - Exponential Smoothing
  - Neural Networks
  - Deep Learning
- Sample Applications
  - Customer trend forecasting
  - Revenue trend forecasting
  - Rainfall forecasting
  - Remaining useful life forecasting (preventive maintenance)

Time Series Analysis (Trend Forecasting)



- Techniques
  - **ARIMA (Autoregressive integrated moving average)**
  - Exponential Smoothing
  - Neural Networks
  - Deep Learning
- Sample Applications
  - Customer trend forecasting
  - Revenue trend forecasting
  - Rainfall forecasting
  - Remaining useful life forecasting (preventive maintenance)

Text Mining

https://ischool.syr.edu/infospace/2013/04/23/what-is-text-mining/

- Text mining, which is sometimes referred to "text analytics" is one way to make qualitative or "unstructured" data **usable by a computer**.
- Convert from unstructured to structured data

Parse → Stop → Stem

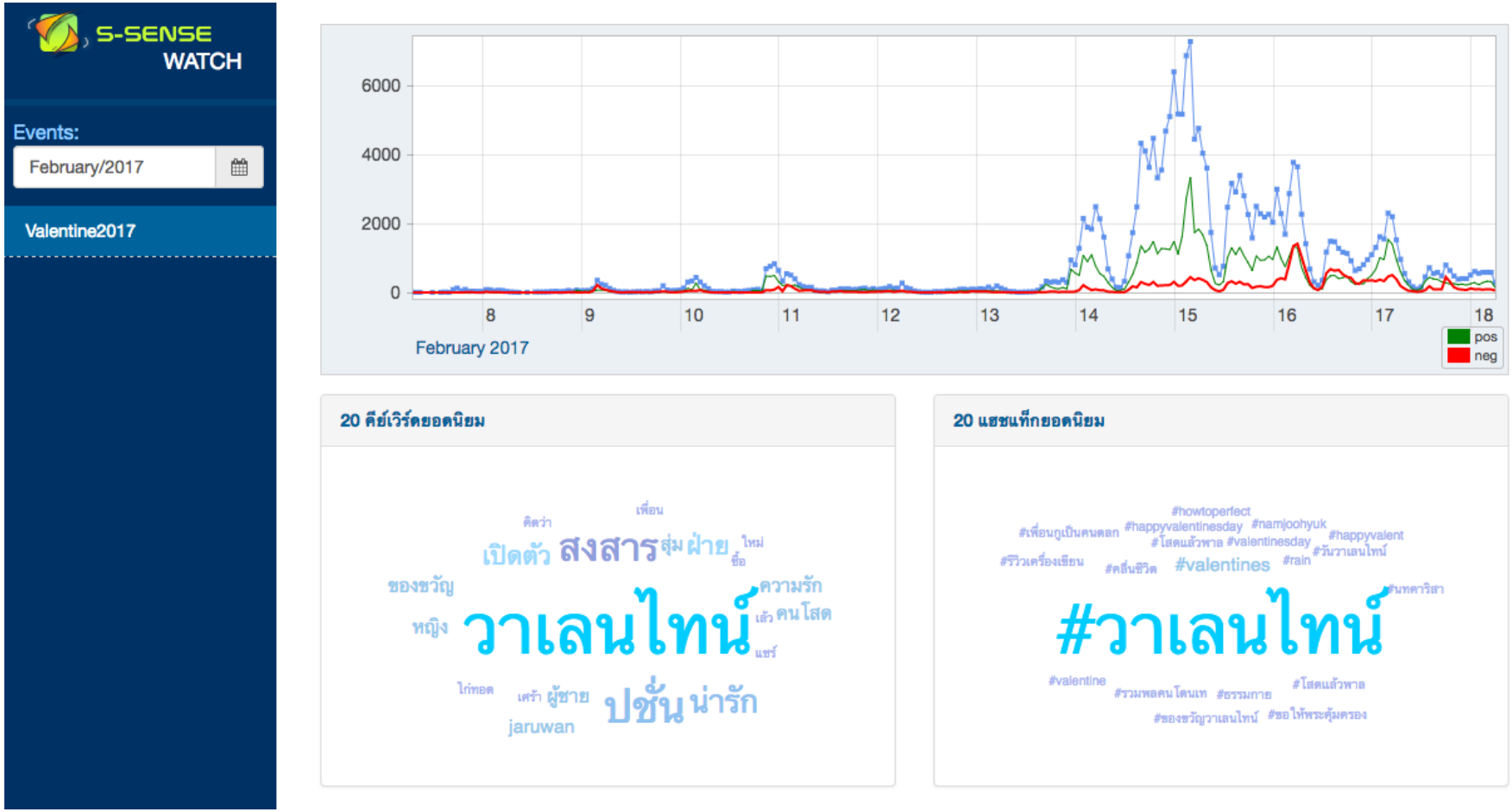
Comments	Good	Like	Hate	Sentiment
Tweet1	7	8	0	😄
Tweet2	1	0	10	😡
Tweet3	2	9	1	😄

Text Mining (cont.): Sentiment Analysis

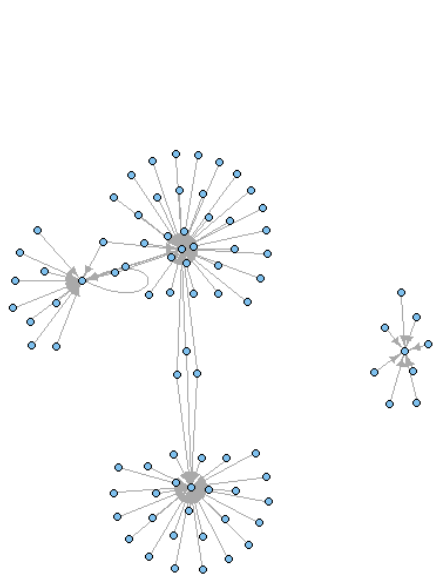
http://pop.ssense.in.th/

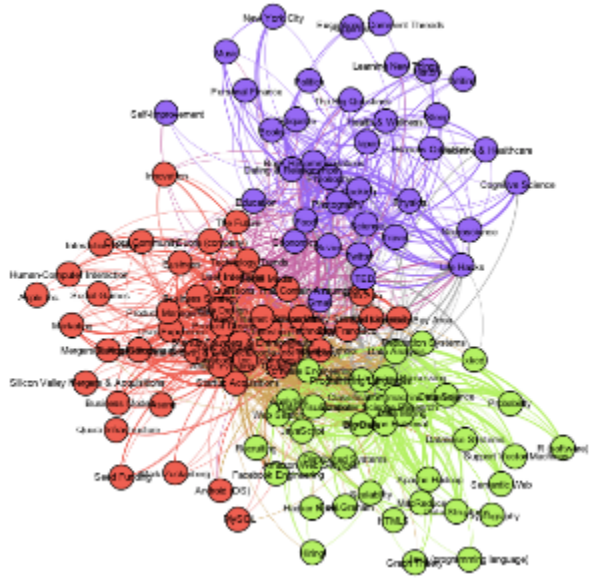
Text Mining (cont.): Emerging Trend Analysis

http://www.ssense.in.th/watch/



Social Network Analysis





- Techniques
  - Centrality: degree, closeness, betweenness, transitivity
  - Community detection
  - Graph Clustering
- Sample Applications
  - Influencer detection
  - Community detection

Scikit-learn:
Machine learning library in Python

Scikit-learn: Machine learning library in Python

- Provides many machine learning tools with a common Estimator interface
- Built in helpers for common ML tasks (e.g., metrics, preprocessing)
- Easily combine algorithms to make a complex pipeline
- Relies heavily on numpy and scipy, often used with pandas

How do you pronounce the project name?

sy-kit learn. sci stands for science!

Why scikit?

There are multiple scikits, which are scientific toolboxes built around SciPy. You can find a list at https://scikits.appspot.com/scikits. Apart from scikit-learn, another popular one is scikit-image.

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, … — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, … — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, … — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

http://scikit-learn.org/stable/index.html

Estimator Interface

Decision Trees

We'll start just by training a single decision tree.

```
In [8]: from sklearn.tree import DecisionTreeClassifier

In [9]: dtree = DecisionTreeClassifier(min_samples_leaf=10, criterion='entropy')

In [10]: dtree.fit(X_train,y_train)

Out[10]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=10, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                                splitter='best')
```

Prediction and Evaluation

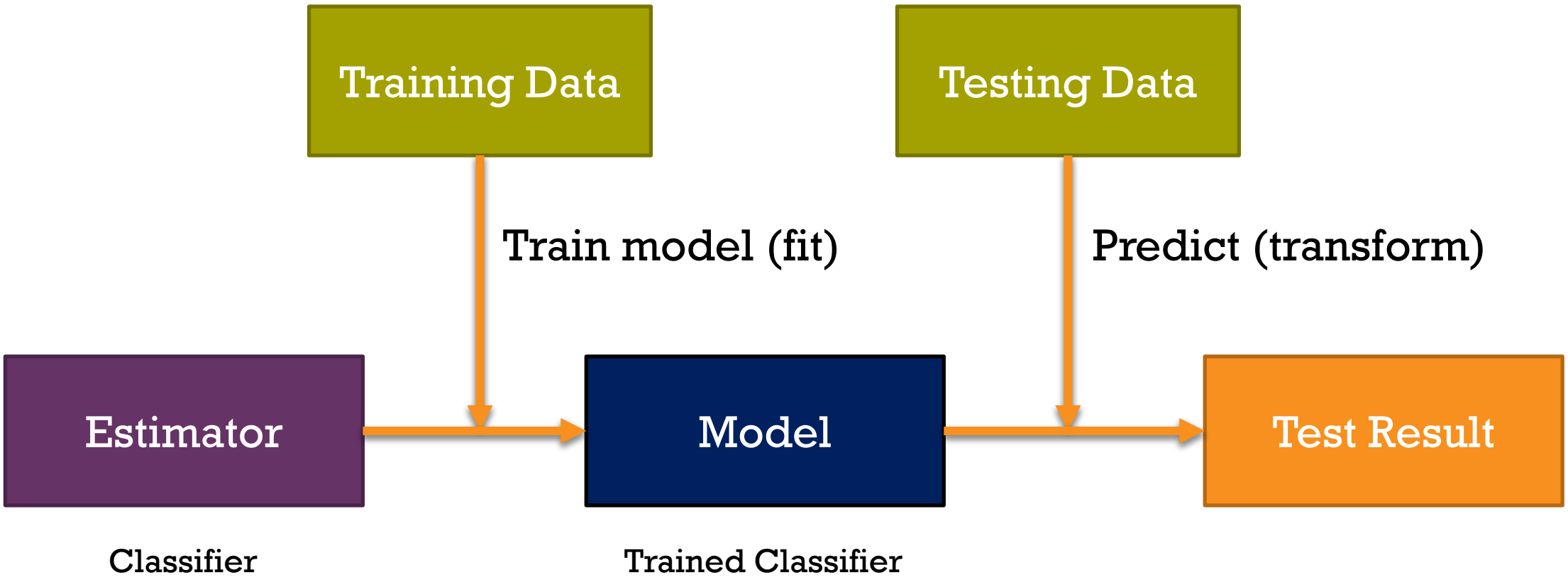
Let's evaluate our decision tree.

```
[11]: predictions = dtree.predict(X_test)

[12]: from sklearn.metrics import classification_report,confusion_matrix

[13]: print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
absent	0.85	0.85	0.85	20
present	0.40	0.40	0.40	5
avg / total	0.76	0.76	0.76	25



Example: Learning to Predict Breast Cancer

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

cancer = load_breast_cancer()      # Get some data
X_train, X_test, y_train, y_test = train_test_split(
        cancer.data, cancer.target,
        stratify=cancer.target, random_state=1337)

tree = DecisionTreeClassifier(random_state=7331)
tree.fit(X_train, y_train)  # Learn a Decision Function
```

Ref: C. Travis Johnson, Mines Linux Users Group on April 27, 2017

Example (cont.): Evaluating Accuracy of a Model

```
# How well did we do?
train_acc = tree.score(X_train, y_train)
test_acc = tree.score(X_test, y_test)
print("Training Accuracy: {:.3f}".format(train_acc))
print("Testing Accuracy: {:.3f}".format(test_acc))
# Training Accuracy: 1.000
# Testing Accuracy: 0.923
```

Ref: C. Travis Johnson, Mines Linux Users Group on April 27, 2017

Example (cont.): Pipeline

```
from sklearn.pipeline import make_pipeline
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import GridSearchCV
pipe = make_pipeline(PCA(), StandardScaler(), SVC())
params = dict(pca__n_components=[2, 5, 10],
              svc__C=[0.1, 10, 100])
grid = GridSearchCV(pipe, param_grid=params)
# Next, call grid.fit on some training data
# This will use cross validation to estimation performance using each
# combination of parameters for pipeline in params dict

# With fitted model
print(grid.best_params_)
```

Ref: C. Travis Johnson, Mines Linux Users Group on April 27, 2017

Demo

Demo Outlines

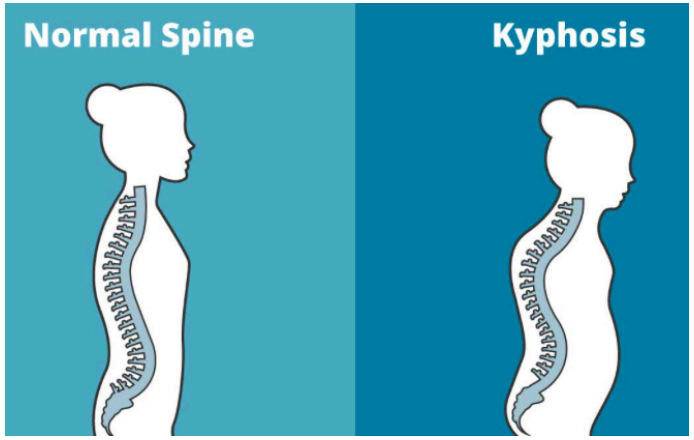
1. Decision Tree
2. Linear Regression
3. Logistic Regression
4. Neural Networks
5. Cross Validation (optional)
6. Grid Search (optional)
7. Kmeans

```
# installation
conda install -c anaconda pydot=1.2.3
conda install -c anaconda pyparsing=2.2.0
conda install -c anaconda graphviz

conda install -c conda-forge mlxtend
```

http://bit.ly/cu-analytics-1

Lab 1: Decision Tree
Kyphosis data set



- A factor with levels absent present indicating if a kyphosis (a type of deformation) was present after the operation.
- 81 rows and 4 columns
- Predictors (3 variables):
  - Age: in months
  - Number: the number of vertebrae involved
  - Start: the number of the first (topmost) vertebrae operated on.
- Target (nominal):
  - Present (1) and Absent (0)

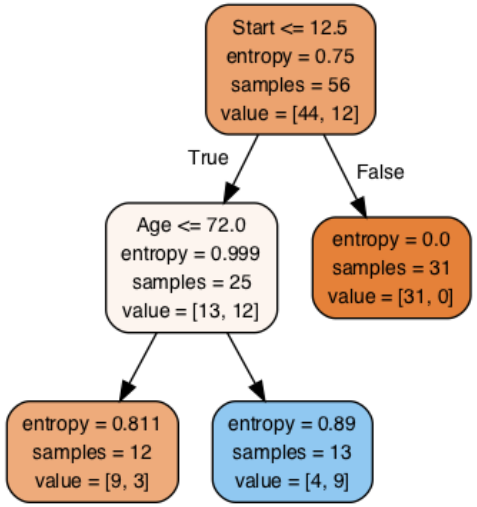
- Library

```
from sklearn.tree import DecisionTreeClassifier
```

```
dtree = DecisionTreeClassifier(min_samples_leaf=10, criterion='entropy')
```

```
dtree.fit(X_train,y_train)
```

- Output image



Lab2: Linear Regression
Housing Prices data set

- Predicting Housing Prices for regions in the USA.
- 5000 rows and 7 columns
- Predictors (5 variables):
  - 'Avg. Area Income': Avg. Income of residents of the city house is located in.
  - 'Avg. Area House Age': Avg Age of Houses in same city
  - 'Avg. Area Number of Rooms': Avg Number of Rooms for Houses in same city
  - Etc.
- Target (numeric):
  - Price

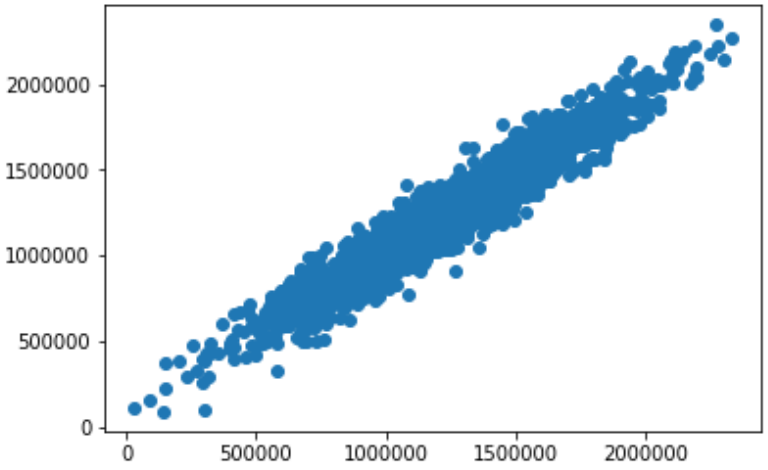
- Library

```
from sklearn.linear_model import LinearRegression
```

```
lm = LinearRegression()
```

```
lm.fit(X_train,y_train)
```

- Output image




```
MAE: 82288.22251914965
MSE: 10460958907.210062
RMSE: 102278.82922291427
```

Lab3: Logistic Regression
Titanic data set

- In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive.
- 891 rows and 12 columns
- Predictors (9 variables):
  - PassengerId
  - Pclass: Ticket class
    - pclass: A proxy for socio-economic status (SES) 1st = Upper, 2nd = Middle, 3rd = Lower
  - Age: Age in years
  - etc.
- Target (nominal):
  - Survival: Survive (1), Dead (0)

- Library

```
from sklearn.linear_model import LogisticRegression
```

```
logmodel = LogisticRegression()
logmodel.fit(X_train,y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

```
predictions = logmodel.predict(X_test)
```

- Output image

	precision	recall	f1-score	support
0	0.81	0.93	0.86	163
1	0.85	0.65	0.74	104
avg / total	0.82	0.82	0.81	267

Lab4: Neural Network
Give Me Some Credit

- Improve on the state of the art in credit scoring by predicting the probability that somebody will experience financial distress in the next two years.
- Predictors (14 variables):
  - MonthlyIncome
  - Age
  - MonthlyIncome
  - NumberOfDependents
  - Etc.
- Target (nominal):
  - Subscribe; (1) and Not Sybscribe (0)

- Library

```
In [0]: from sklearn.neural_network import MLPClassifier

In [0]: mlp = MLPClassifier(hidden_layer_sizes=(22),max_iter=500, random_state=12345)

In [15]: mlp.fit(X_train,y_train)

Out[15]: MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
         beta_2=0.999, early_stopping=False, epsilon=1e-08,
         hidden_layer_sizes=22, learning_rate='constant',
         learning_rate_init=0.001, max_iter=500, momentum=0.9,
         nesterovs_momentum=True, power_t=0.5, random_state=12345,
         shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1,
         verbose=False, warm_start=False)
```

- Output image

```
In [18]: print(confusion_matrix(y_test,predictions))

[[27693   303]
 [ 1728   344]]

In [19]: print(classification_report(y_test,predictions))

             precision    recall  f1-score   support

          0       0.94      0.99      0.96     27996
          1       0.53      0.17      0.25      2072

avg / total       0.91      0.93      0.92     30068
```

Lab5: k Nearest Neighbors (kNN)
Classified Data data set

- 1000 rows and 4 columns
- Predictors (3 variables):
  - WTT,PTI,EQW,SBI,LQE,QWG,FDJ,PJF,HQE,NXJ
  - Etc.
- Target (nominal):
  - TARGET CLASS (1) and Absent (0)

- Library

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier(n_neighbors=1)
```

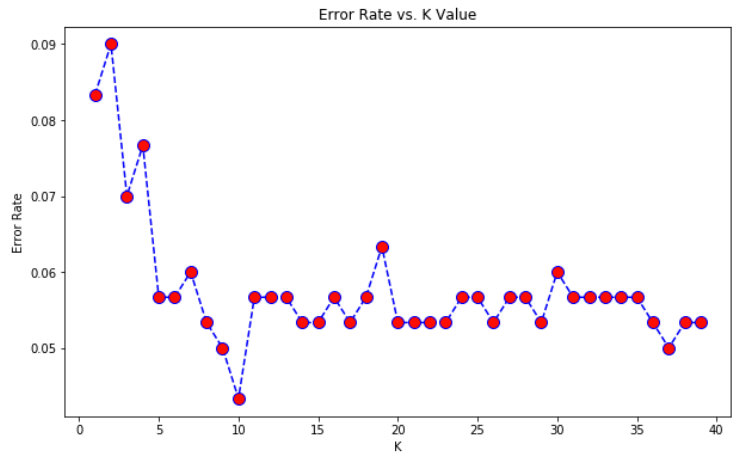
```
knn.fit(X_train,y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=1, p=2,
           weights='uniform')
```

```
pred = knn.predict(X_test)
```

- Output image

	precision	recall	f1-score	support
0	0.93	0.90	0.91	145
1	0.91	0.94	0.92	155
avg / total	0.92	0.92	0.92	300



Lab6: Grid Search and Cross Validation

- Library and Output image

```
from sklearn.grid_search import GridSearchCV
```

```
/Library/Frameworks/Python.framework/Versions/3.6/lib/python
DeprecationWarning: This module was deprecated in version 0.
to which all the refactored classes and functions are moved.
  DeprecationWarning)
```

```
# define the parameter values that should be searched
k_range = list(range(1, 31))
print(k_range)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
29, 30]
```

```
# create a parameter grid: map the parameter names to the values
param_grid = dict(n_neighbors=k_range)
print(param_grid)
```

```
{'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
25, 26, 27, 28, 29, 30]}
```

```
# instantiate the grid
grid = GridSearchCV(knn, param_grid, cv=10, scoring='accuracy')
```

Lab7: Sequential Feature Selector

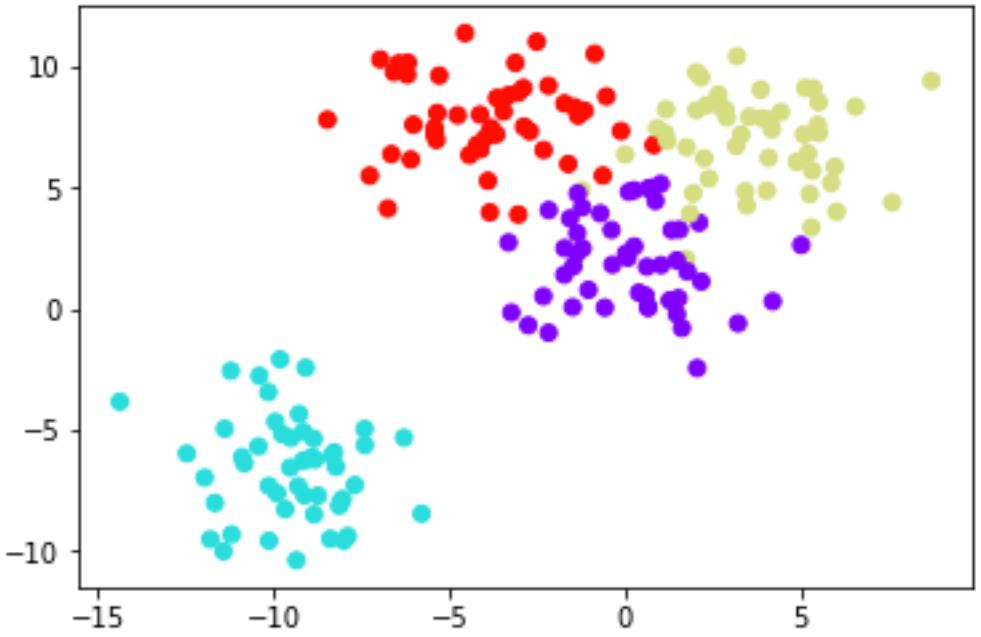
- Sequential Forward Selection (SFS)
- Sequential Backward Selection (SBS)
- Sequential Forward Floating Selection (SFFS)

- Example 1 - A simple Sequential Forward Selection example
- Example 2 - Toggling between SFS, SBS and SFFS
- Example 3 - Visualizing the results in Data Frames
- Example 4 - Sequential Feature Selection for Regression
- Example 5 -- Using the Selected Feature Subset For Making New Predictions
- Example 6 -- Sequential Feature Selection and Grid Search
- Example 7 -- Selecting the "best" feature combination in a k-range

Lab8: K-Means Clustering
Make blobs (Sklearn) data set

```
from sklearn.datasets import make_blobs
```

```
# Create Data
data = make_blobs(n_samples=200, n_features=2,
            centers=4, cluster_std=1.8,random_state=101)
```



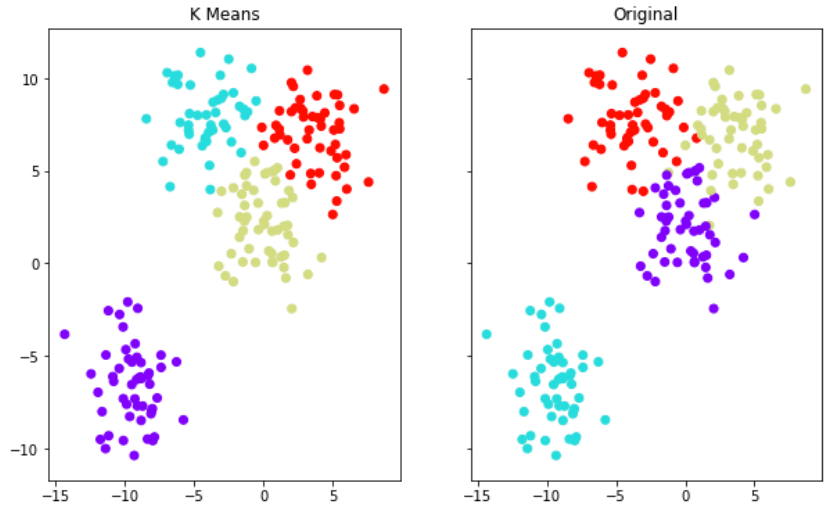
- Library

```
from sklearn.cluster import KMeans
```

```
kmeans = KMeans(n_clusters=4)
```

```
kmeans.fit(data[0])
```

- Output image



Non-performing Loan (NPL)

Input

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
PRODUCT	90302	5	Class C Loan	30911	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Delinquent_Status	90302	8	A.CURRENT	29184	NaN	NaN	NaN	NaN	NaN	NaN	NaN
REGION	90302	7	ภาคตะวันออกเฉียงเหนือ	21927	NaN	NaN	NaN	NaN	NaN	NaN	NaN
PROVINCE	90302	78	กรุงเทพมหานคร	7206	NaN	NaN	NaN	NaN	NaN	NaN	NaN
AGE	90302	8	03.31-40	20462	NaN	NaN	NaN	NaN	NaN	NaN	NaN
ZIP_CODE	90302	3104	11000	234	NaN	NaN	NaN	NaN	NaN	NaN	NaN
REGION_ENG	90302	8	4.North East	21541	NaN	NaN	NaN	NaN	NaN	NaN	NaN
ACTIVE_ACCTS	90302	NaN	NaN	NaN	12.7853	158.874	0	0	1	2	9209
ACTIVE_OS	90302	NaN	NaN	NaN	803407	5.63203e+06	0	0	42036.9	185821	2.96747e+08
TOTAL_WO_ACCTS	90302	NaN	NaN	NaN	12.3521	247.755	0	0	0	1	18328
TOTAL_WO_OS	90302	NaN	NaN	NaN	589169	6.21974e+06	0	0	0	0	4.11723e+08
REMAINING_WO_ACCTS	90302	NaN	NaN	NaN	6.46752	142.409	0	0	0	0	10330
REMAINING_WO_OS	90302	NaN	NaN	NaN	395565	4.11305e+06	0	0	0	0	2.64296e+08
TDR_ACCTS	90302	NaN	NaN	NaN	0.283405	0.613134	0	0	0	0	4
TDR_OS	90302	NaN	NaN	NaN	21951.4	96621.8	0	0	0	0	4.62866e+06

Prediction

	index	PRODUCT	Delinquent_Status	REGION	PROVINCE	AGE	ZIP_CODE	REGION_ENG	ACTIVE_ACCTS	ACTIVE_OS	TOTAL_WO_ACCTS	TOTAL_WO_OS	REMAINING_WO_ACCTS	REMAINING_WO_OS	TDR_ACCTS	TDR_O	ปกติ	ผิดนัดชำระ
5113	85113	Class C Loan	Z.WO	ภาคกลาง	พิจิตร	06.56-60	66230	2.Central	0	0.00	3	183435.67	2	161805.49	0	0.	0.0	1.0
10115	90115	Class C Loan	B.X DAY	ภาคเหนือ	เพชรบูรณ์	05.51-55	67220	6.North	1	94838.95	0	0.00	0	0.00	0	0.	0.0	1.0
10096	90096	Class C Card	B.X DAY	ภาคกลาง	นครนายก	06.56-60	26120	2.Central	1	18840.35	0	0.00	0	0.00	0	0.	0.0	1.0
5424	85424	Class C Loan	B.X DAY	ภาคตะวันออกเฉียงเหนือ	ศรีสะเกษ	03.31-40	33210	4.North East	2	192908.28	0	0.00	0	0.00	0	0.	0.0	1.0
10098	90098	Class B Card	A.CURRENT	ภาคตะวันออกเฉียงเหนือ	บึงกาฬ	07.>60	38210	7.NA	0	0.00	0	0.00	0	0.00	0	0.	0.0	1.0
93	80093	Class C Loan	F.120 DAY	ภาคตะวันออก	ชลบุรี	03.31-40	20170	3.East	2	160261.35	0	0.00	0	0.00	0	0.	0.0	1.0
5267	85267	Class C Loan	F.120 DAY	ภาคกลาง	เพชรบุรี	03.31-40	76120	2.Central	2	163087.38	0	0.00	0	0.00	0	0.	0.0	1.0
5262	85262	Class C Card	A.CURRENT	กรุงเทพและปริมณฑล	กรุงเทพมหานคร	03.31-40	10243	1.BKK & Surround	1	38145.77	0	0.00	0	0.00	0	0.	0.0	1.0
10117	90117	Class C Loan	D.60 DAY	ภาคตะวันออกเฉียงเหนือ	นครพนม	02.21-30	48120	4.North East	1	19066.19	0	0.00	0	0.00	0	0.	0.0	1.0
10116	90116	Class B Card	E.90 DAY	กรุงเทพและปริมณฑล	ปทุมธานี	04.41-50	12000	1.BKK & Surround	1	177518.74	0	0.00	0	0.00	0	0.	0.0	1.0

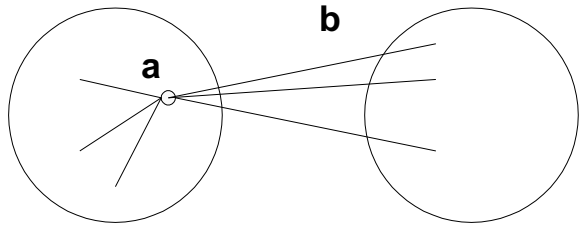
Appendix

Silhouette Coefficient

- Silhouette Coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, *I*
  - ***a*** = average distance of ***i*** to the points in the same cluster
  - ***b*** = min (average distance of ***i*** to points in another cluster)
  - silhouette coefficient of **i**:

    s = 1 – a/b if **a < b**

  - Typically between 0 and 1.
  - The closer to 1 the better.



- Can calculate the Average Silhouette width for a cluster or a clustering