深度學習Deep Learning (4)

W4: **scikit-learn**程式練習

朱學亭老師

深度學習(1)

- Part 1: Python程式
  - Jupyter notebook & Colab
  - Python Variables (int, float, string)
  - Python Basics (IPO model)
  - Python Data Structures (Lists & Dictionaries)
  - Python Function & Lambda
  - Python Date & Time
  - Python Crawler
  - Python Database

[3]

```
#收集學校新聞的大數據
import re
from urllib import request
count = 0
sss = ["2008", "2009","2010", "2011", "2012","2013","2014","2015","2016","2017","2018"]
titles=list()
for i in range(len(sss)):
    year = sss[i]
    with request.urlopen('http://www.asia.edu.tw/news1.php?y='+year) as response:
        html = response.read().decode('utf-8')
        #print(html)
        pattern = '<font color="#446666" face="新細明體" style="font-weight: 700;" size="2">'
        for pos in re.finditer(pattern, html):
            pos2 = html.find('</font>', pos.end())
            sub = html[pos.end():pos2]
            titles.append(sub)
            count = count + 1
print (count)
```

```
4100
```

深度學習(2)

- Part 2: Machine learning
  - Numpy & Pandas
  - scikit-learn
  - PCA &t-SNE
  - Logistic regression
  - One-hot encoding
  - Supervised learning
  - Reinforcement learning
- Part 3: Deep learning basics
  - Training and Loss
  - Gradient Descent/Optimizer
  - ROC Curve and AUC
  - Overfitting & Regularization
  - Activation Functions
  - Loss Functions
  - Confusion matrix
  - Transfer learning

深度學習-3

- Part 4: Deep learning with Tensorflow 2
  - TF2 Hello World/TF2 Keras layers
  - Convolutional Neural Network (CNN)
  - Recurrent Neural Networks (RNN)
  - Generative Adversarial Network (GAN)
  - Tensorboard
  - Interpretability(tf-explain)
  - Encoder-Decoder
  - Attention mechanism
  - Transformers

深度學習(4)

- Part 5: Deep learning topics for Image processing (IP)
  - Image Classification
  - Image Segmentation
  - Image Captioning
  - Image Generation
- Part 6: Deep learning topics for Natural Language Processing (NLP)
  - Authorship Attribution
  - Sentiment analysis
  - Text Summarization
  - Question Answering
- Part 7: Deep Learning for Games
  - Q-Learning
  - Deep Reinforcement Learning
  - Deep Q-Learning

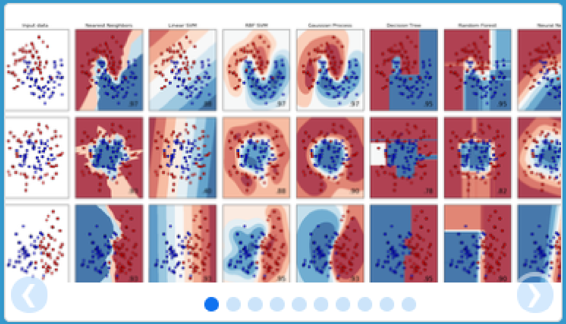
scikit-learn

Machine Learning in Python

scikit learn

Home Installation Documentation ▾ Examples

Google Custom Search Search ×



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

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

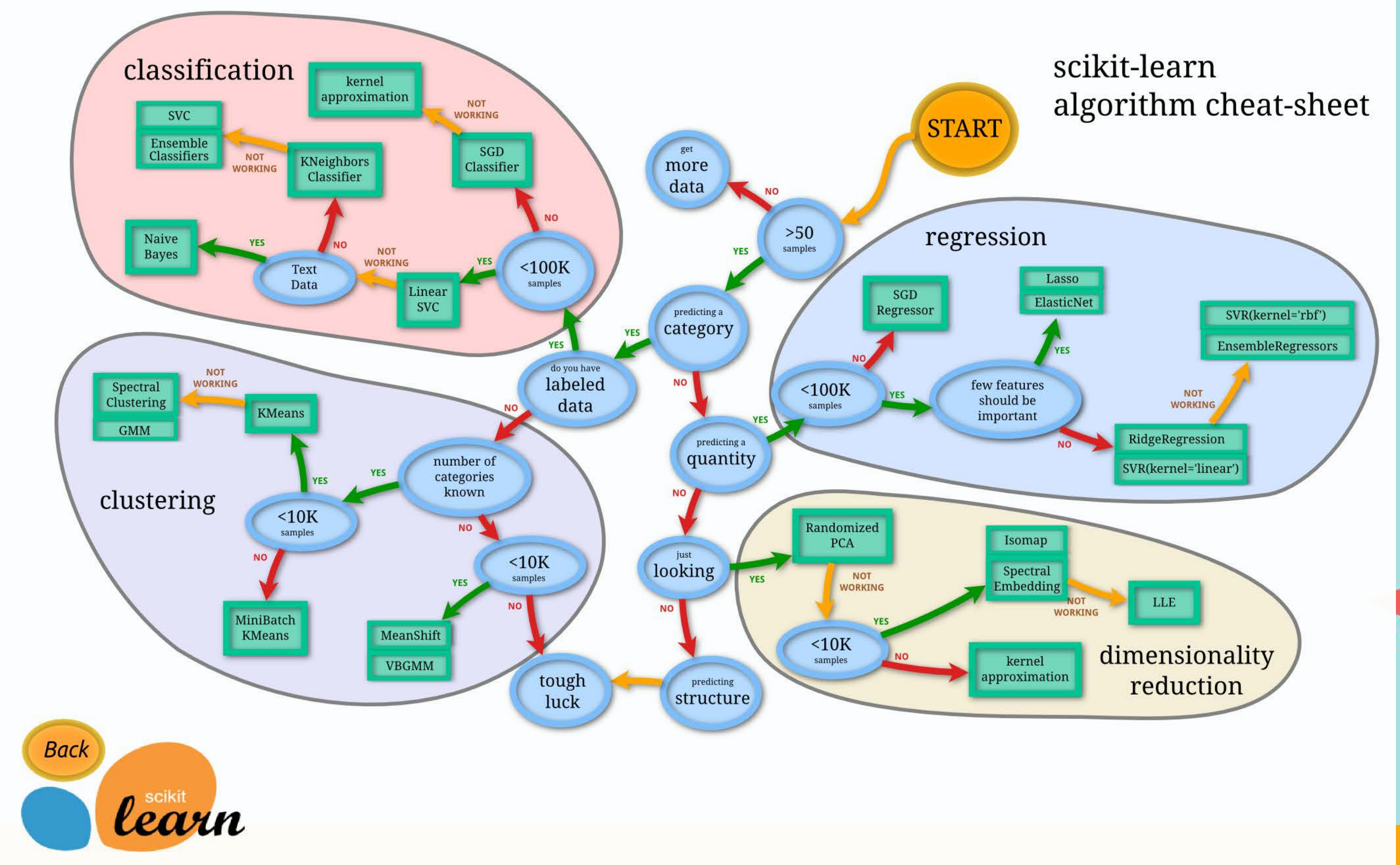
Preprocessing

Feature extraction and normalization.

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

— Examples

Choosing the right estimator



Scikit-learn datasets

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

- Toy datasets
  - load_boston([return_X_y]) Load and return the boston house-prices dataset (regression).
  - load_iris([return_X_y]) Load and return the iris dataset (classification).
  - load_diabetes([return_X_y]) Load and return the diabetes dataset (regression).
  - load_digits([n_class, return_X_y]) Load and return the digits dataset (classification).
  - load_linnerud([return_X_y]) Load and return the linnerud dataset (multivariate regression).
  - load_wine([return_X_y]) Load and return the wine dataset (classification).
  - load_breast_cancer([return_X_y]) Load and return the breast cancer wisconsin dataset (classification).

Boston house prices dataset

Number of Instances:	
	506
Number of Attributes:	
	13 numeric/categorical predictive. Median Value (attribute 14) is usually the target.
Attribute Information (in order):	

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS proportion of non-retail business acres per town
- CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per $10,000
- PTRATIO pupil-teacher ratio by town
- B 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT % lower status of the population
- MEDV Median value of owner-occupied homes in $1000's

Missing Attribute Values:	
	None
Creator:	Harrison, D. and Rubinfeld, D.L.

Diabetes dataset

Ten baseline variables, age, sex, body mass index, average blood pressure, and six blood serum measurements were obtained for each of n = 442 diabetes patients, as well as the response of interest, a quantitative measure of disease progression one year after baseline.

Data Set Characteristics:

Number of Instances:	
	442
Number of Attributes:	
	First 10 columns are numeric predictive values
Target:	Column 11 is a quantitative measure of disease progression one year after baseline
Attribute Information:	
	• Age • Sex • Body mass index • Average blood pressure • S1 • S2 • S3 • S4 • S5 • S6

Note: Each of these 10 feature variables have been mean centered and scaled by the standard deviation times n_samples (i.e. the sum of squares of each column totals 1).

Wine recognition dataset

Number of Instances:	
	178 (50 in each of three classes)
Number of Attributes:	
	13 numeric, predictive attributes and the class
Attribute Information:	

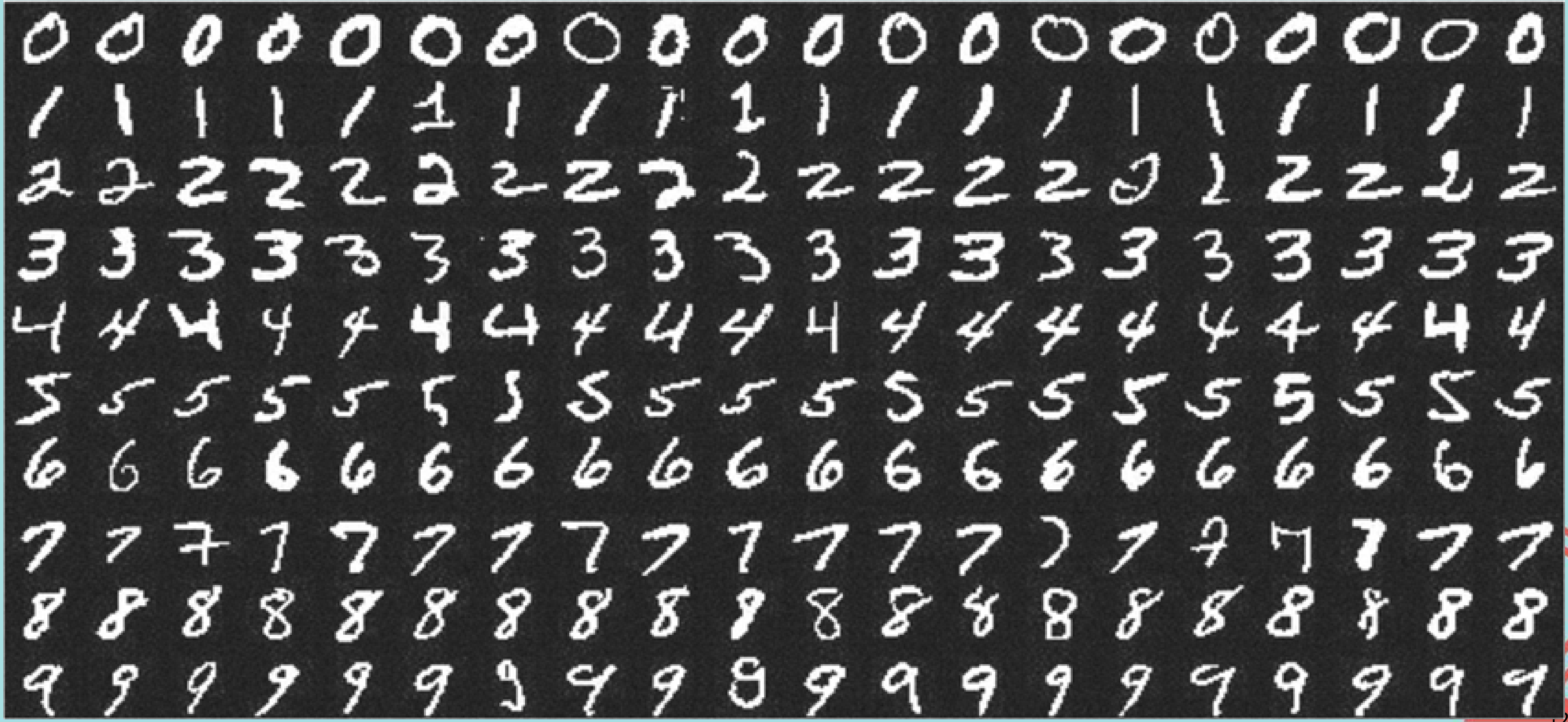
- Alcohol
- Malic acid
- Ash
- Alcalinity of ash
- Magnesium
- Total phenols
- Flavanoids
- Nonflavanoid phenols
- Proanthocyanins
- Color intensity
- Hue
- OD280/OD315 of diluted wines
- Proline

- **class:**
  - class_0
  - class_1
  - class_2

Alcohol:	11.0	14.8	13.0	0.8
Malic Acid:	0.74	5.80	2.34	1.12
Ash:	1.36	3.23	2.36	0.27
Alcalinity of Ash:	10.6	30.0	19.5	3.3
Magnesium:	70.0	162.0	99.7	14.3
Total Phenols:	0.98	3.88	2.29	0.63
Flavanoids:	0.34	5.08	2.03	1.00
Nonflavanoid Phenols:	0.13	0.66	0.36	0.12
Proanthocyanins:	0.41	3.58	1.59	0.57
Colour Intensity:	1.3	13.0	5.1	2.3
Hue:	0.48	1.71	0.96	0.23
OD280/OD315 of diluted wines:	1.27	4.00	2.61	0.71
Proline:	278	1680	746	315

Missing Attribute Values:	
	None
Class Distribution:	
	class_0 (59), class_1 (71), class_2 (48)
Creator:	R.A. Fisher
Donor:	Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
Date:	July, 1988

MNIST handwritten digit database



Breast cancer wisconsin (diagnostic) dataset

Number of Instances:

569

Number of Attributes:

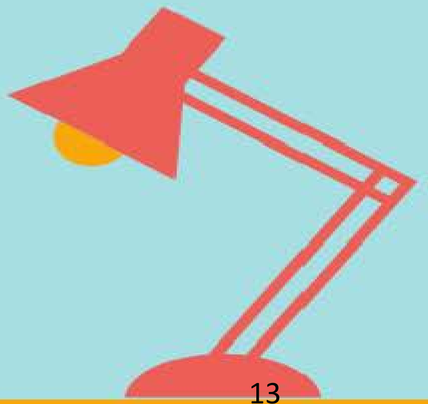
30 numeric, predictive attributes and the class

Attribute Information:

- radius (mean of distances from center to points on the perimeter)
- texture (standard deviation of gray-scale values)
- perimeter
- area
- smoothness (local variation in radius lengths)
- compactness (perimeter^2 / area - 1.0)
- concavity (severity of concave portions of the contour)
- concave points (number of concave portions of the contour)
- symmetry
- fractal dimension (“coastline approximation” - 1)

The mean, standard error, and “worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.

- **class:**
    - WDBC-Malignant
    - WDBC-Benign



Iris data

- from sklearn import datasets
- iris = datasets.load_iris()

- 1. sepal length in cm
  2. sepal width in cm
  3. petal length in cm
  4. petal width in cm
  5. class:
  -- Iris Setosa
  -- Iris Versicolour
  -- Iris Virginica

Classes	3
Samples per class	50
Samples total	150
Dimensionality	4
Features	real, positive

Training set/Test set

```
from sklearn.model_selection import train_test_split
from sklearn import datasets
iris = datasets.load_iris()
#print(iris.keys)
X = iris.data[:,0:4]
y = iris.target
X_train, X_test, y_train, y_test =
                    train_test_split(X, y, test_size=0.333)
```

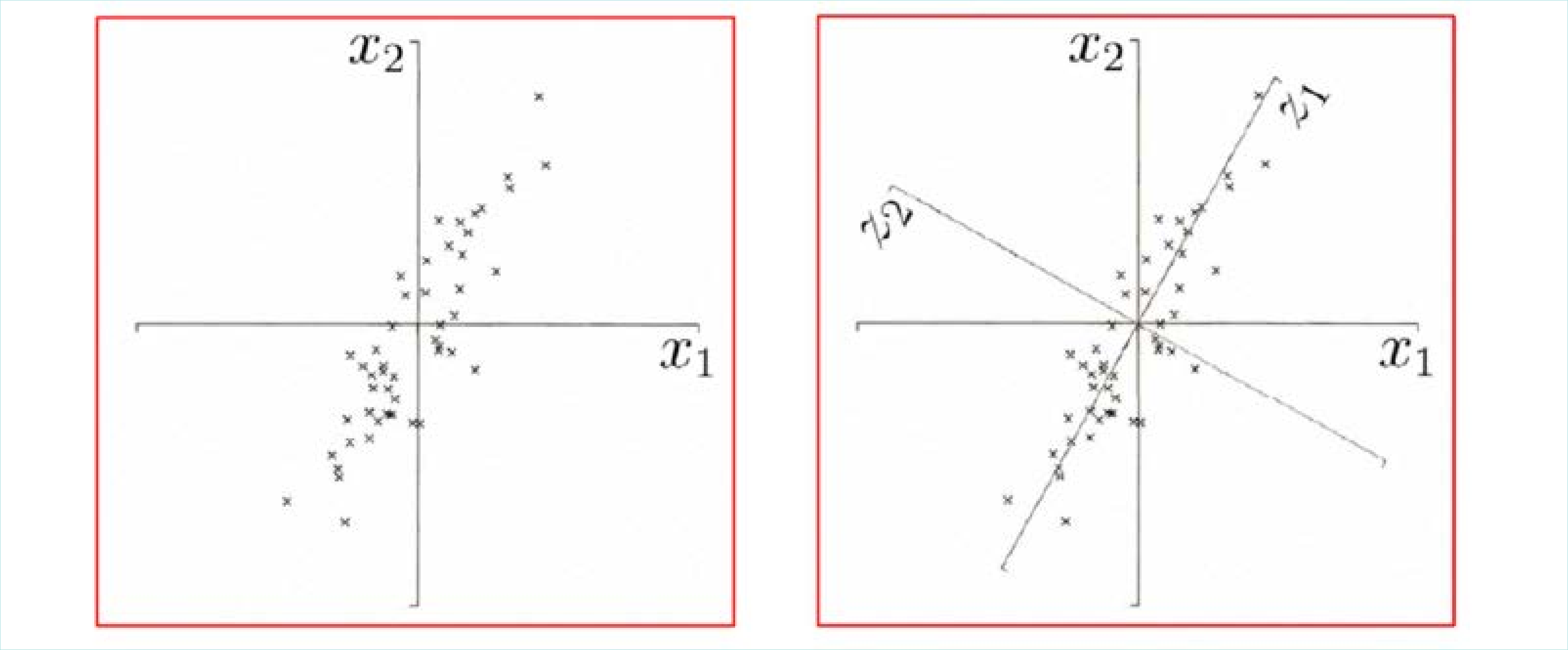
Principal component analysis (PCA)

- Widely used method for unsupervised, linear dimensionality reduction

- GOAL: account for variance of data in as few dimensions as possible (using linear projection)

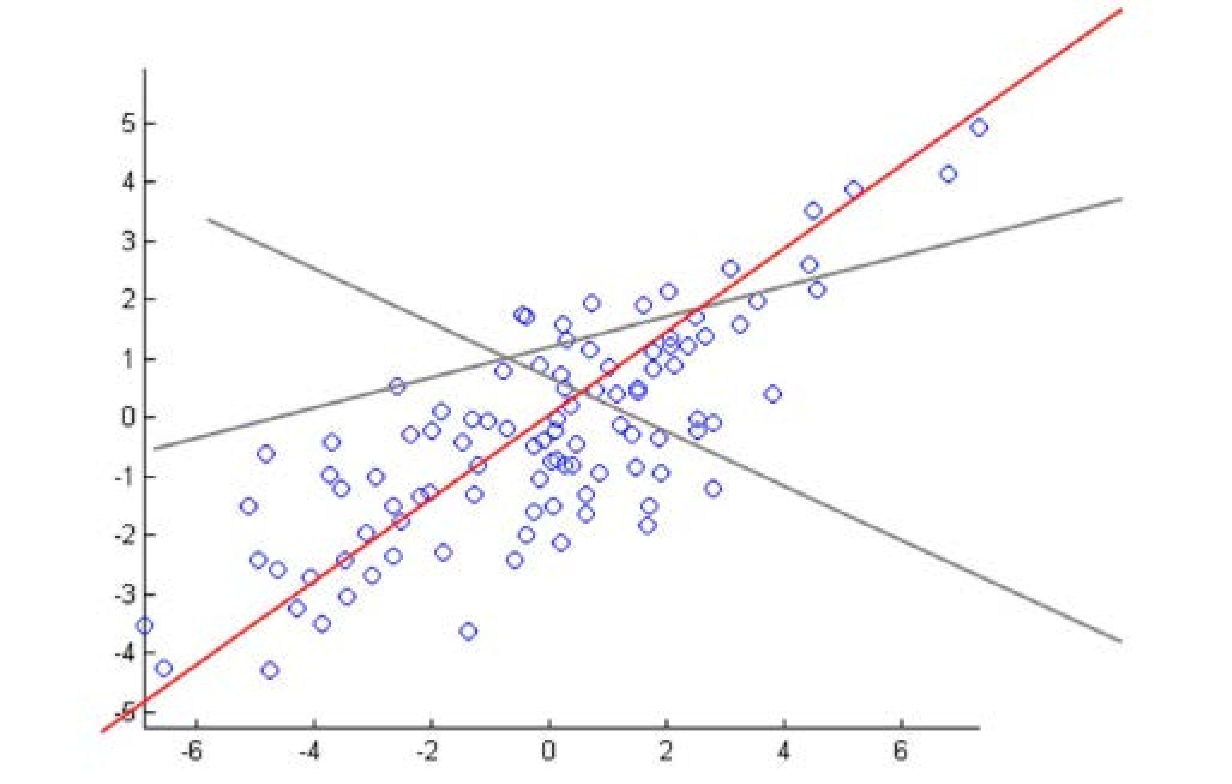
Geometric picture of principal components (PCs)

- First PC is the projection direction that maximizes the variance of the projected data
- Second PC is the projection direction that is orthogonal to the first PC and maximizes variance of the projected data



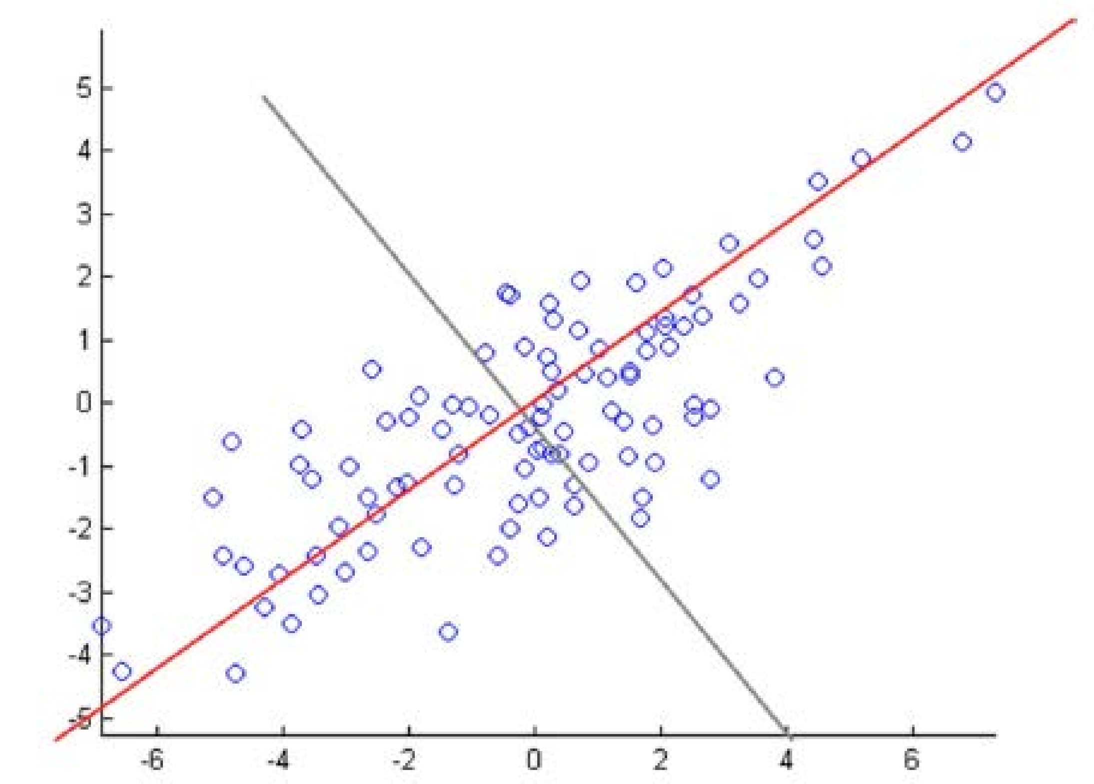
PCA: conceptual algorithm

- Find a line, such that when the data is projected onto that line, it has the maximum variance.



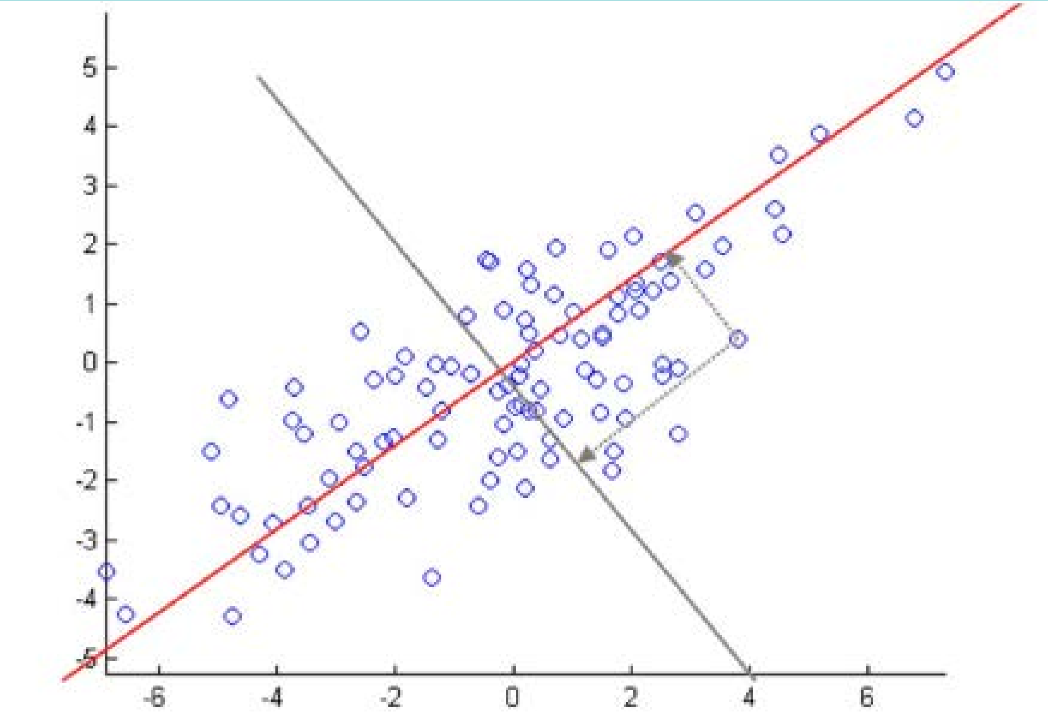
PCA: conceptual algorithm

- Find a second line, orthogonal to the first, that has maximum projected variance.



PCA: conceptual algorithm

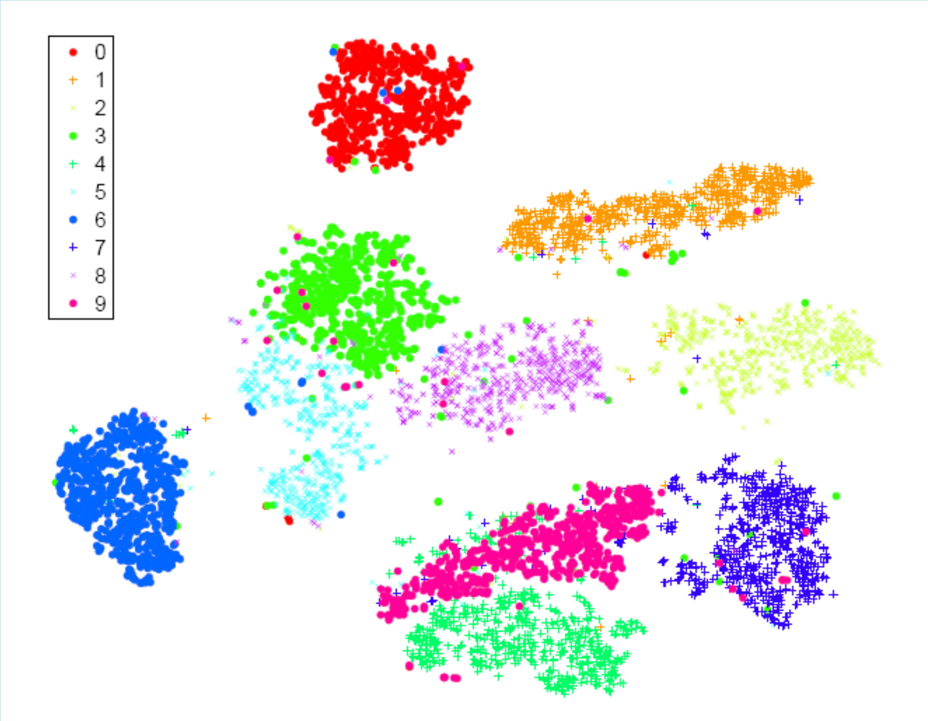
- Repeat until have *k* orthogonal lines
- The projected position of a point on these lines gives the coordinates in the *k*-dimensional reduced space.



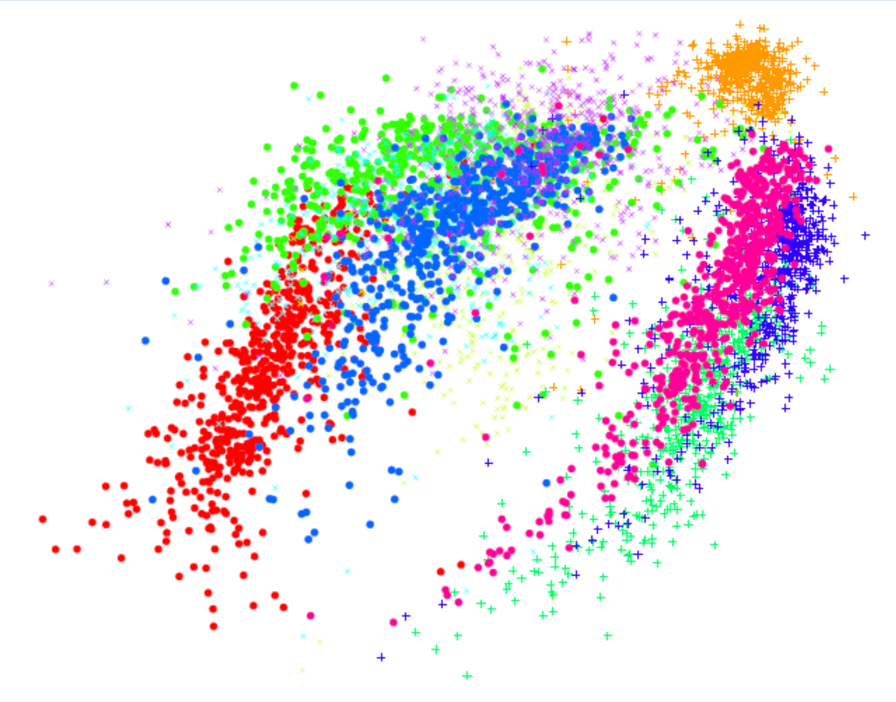
T-Stochastic neighbor embedding (t-SNE)

- Visualizes high-dimensional data in a 2- or 3-dimensional map.
- Better than existing techniques at creating a single map that reveals structure at many different scales.
- Particularly good for high-dimensional data that lie on several different, but related, low-dimensional manifolds.
  - Example: images of objects from multiple classes seen from multiple viewpoints.

Visualization of classes in MNIST data



t-SNE



ISOMAP

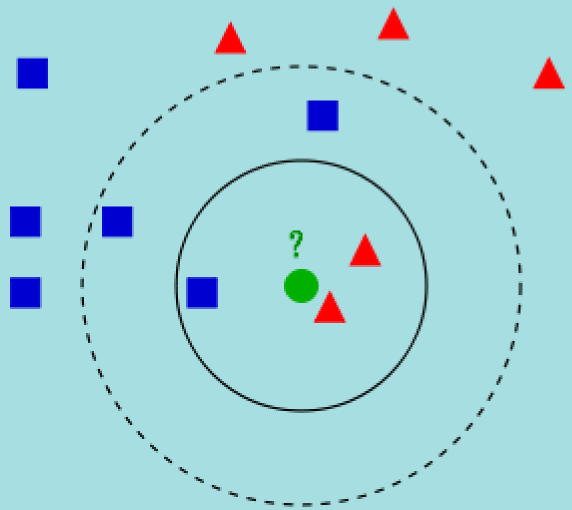
Classification algorithms

- KNN: K-nearest neighbors,
- SVM: Support vector machine
- RF: Random forest
- DT: Decision tree
- NN: Neural network (MLP)

http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

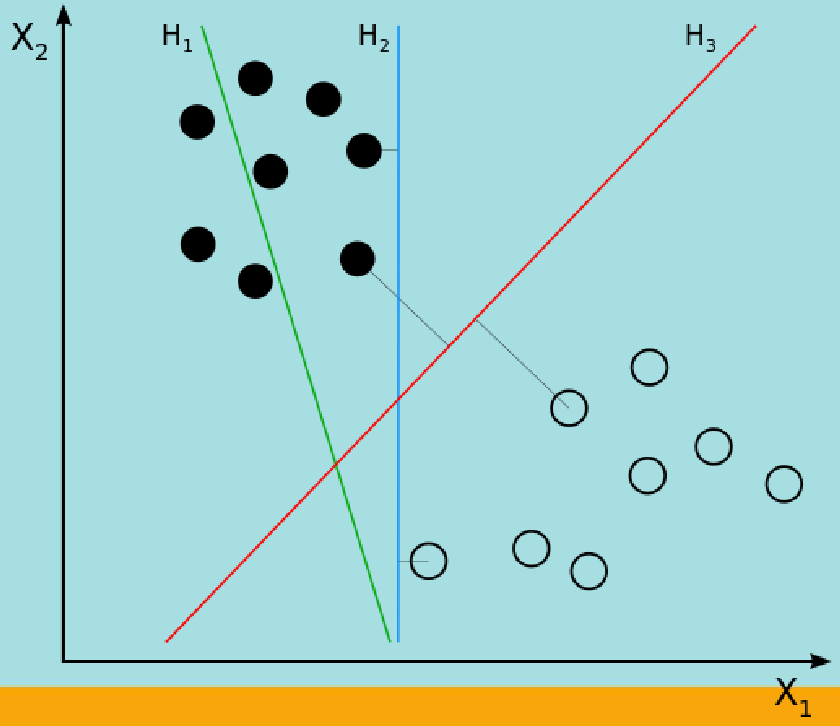
KNN: K-nearest neighbors

- In *k-NN classification*, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors.



SVM: Support vector machine

- A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space

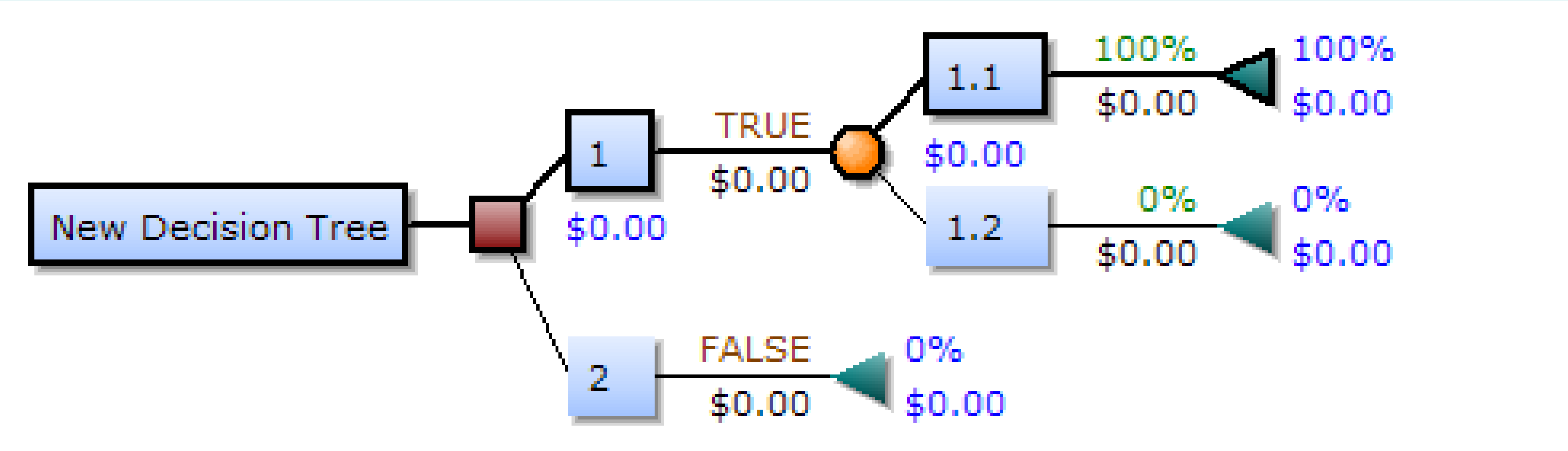


RF: Random forest

- Random forests are an ensemble learning method that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

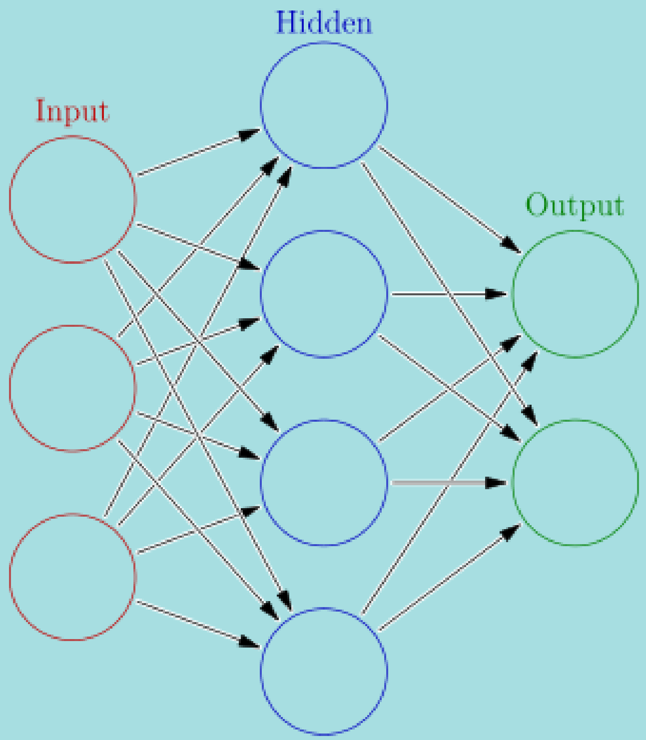
DT: Decision tree

- A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences.



NN: Neural network (MLP)

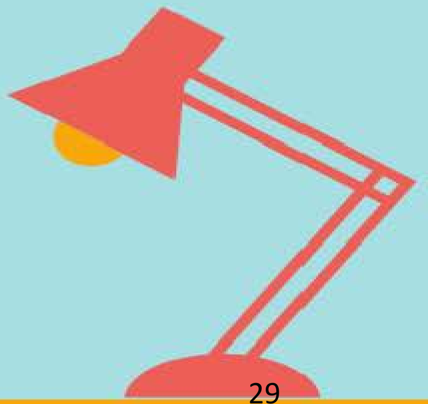
- A multilayer perceptron (MLP) is a class of feedforward artificial neural network.



OneHotEncoding

ID	Gender
1	Male
2	Female
3	Not Specified
4	Not Specified
5	Female

ID	Male	Female	Not Specified
1	1	0	0
2	0	1	0
3	0	0	1
4	0	0	1
5	0	1	0



Thanks! Q&A