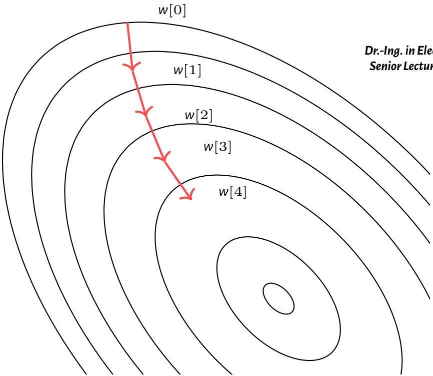
An Introduction To Machine Learning[1]

Abdelbacet Mhamdi
abdelbacet.mhamdi@bizerte.r-iset.tn

Dr.-Ing. in Electrical Engineering
Senior Lecturer at ISET Bizerte



"Computers are able to see, hear and learn. Welcome to the future."

Dave Waters

"This is nothing. In a few years, that bot will move so fast you'll need a strobe light to see it. Sweet dreams..."

Elon Musk

"Machine intelligence is the last invention that humanity will ever need to make."

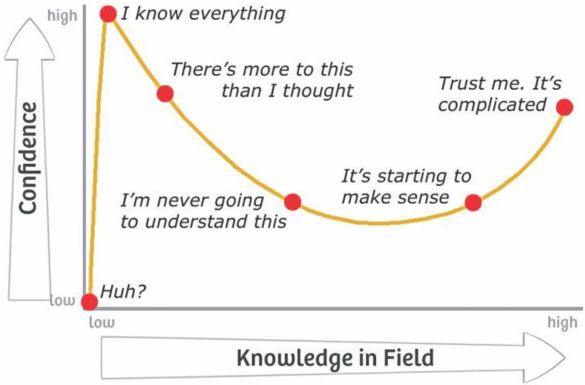
Nick Bostrom

[1]Available @ https://github.com/a-mhamdi/mlpy/

Disclaimer

This document features some materials gathered from multiple online sources. Please note no copyright infringement is intended, and I do not own nor claim to own any of the original materials. They are used for educational purposes only. I have included links solely as a convenience to the reader. Some links within these slides may lead to other websites, including those operated and maintained by third parties. The presence of such a link does not imply a responsibility for the linked site or an endorsement of the linked site, its operator, or its contents.

Dunning-Kruger Effect

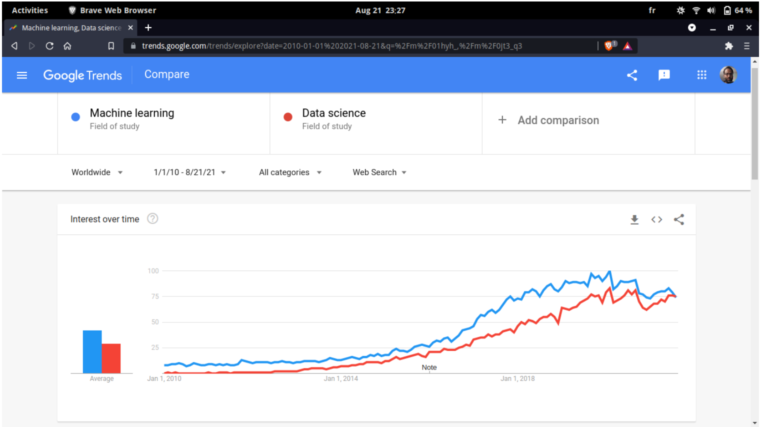


Kruger, J. and Dunning, D. (1999) *Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments.* J Pers Soc Psychol. 77(6) pp. 1121–1134.

Roadmap

Next...

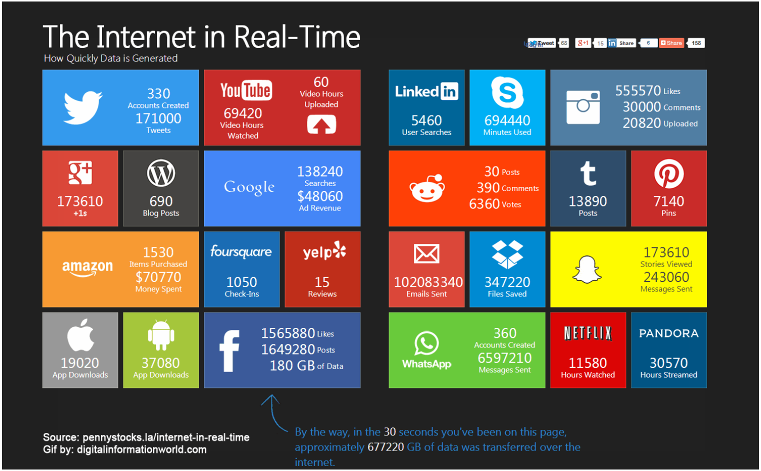
Trends



"Numbers represent search interest relative to the highest point on the chart for the given region and time.

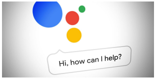
- A value of 100 is the peak popularity for the term;
- A value of 50 means that the term is half as popular;
- A score of 0 means there was not enough data for this term."

Global Data Traffic



Update on the internet in real time is available ▸ here.

Top Uses



Literature Review (1/3)

"The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience."

Mitchell, T. (1997) *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill.

Literature Review (2/3)

"Machine learning (ML) is a scientific discipline that concerns developing learning capabilities in computer systems. Machine learning is one of central areas of Artificial Intelligence (AI). It is an interdisciplinary area that combines results from statistics, logic, robotics, computer science, computational intelligence, pattern recognition, data mining, cognitive science, and more."

Wojtusiak, J. (2012) Machine learning. In *Encyclopedia of the Sciences of Learning*, pages 2082–2083. Springer US.

Literature Review (3/3)

> "Machine learning is an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment. They are considered the working horse in the new era of the so-called big data. Techniques based on machine learning have been applied successfully in diverse fields ranging from pattern recognition, computer vision, spacecraft engineering, finance, entertainment, and computational biology to biomedical and medical applications. [...] The ability of machine learning algorithms to learn from current context and generalize into unseen tasks would allow improvements in both the safety and efficacy of radiotherapy practice leading to better outcomes."

El Naqa, I. and Murphy, M. J. (2015) *What Is Machine Learning?*, pages 3–11. Springer International Publishing.

Debrief

Arthur Samuel (1959)

Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell (1998)

Well-posed Learning Problem: A computer is said to learn from experience $\mathcal{E}$ with respect to some task $\mathcal{T}$ and some performance measure $\mathcal{P}$, if its performance on $\mathcal{T}$, as measured by $\mathcal{P}$, improves with experience $\mathcal{E}$.

Debrief

Arthur Samuel (1959)

Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell (1998)

Well-posed Learning Problem: A computer is said to learn from experience $\mathcal{E}$ with respect to some task $\mathcal{T}$ and some performance measure $\mathcal{P}$, if its performance on $\mathcal{T}$, as measured by $\mathcal{P}$, improves with experience $\mathcal{E}$.

Task #1

Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task $\mathcal{T}$ in this setting?

1. Classifying emails as spam or not spam;
2. Watching you label emails as spam or not spam;
3. The number (or fraction) of emails correctly classified as spam/not spam;
4. None of the above-this not a machine learning problem.

Debrief

Arthur Samuel (1959)

Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell (1998)

Well-posed Learning Problem: A computer is said to learn from experience $\mathcal{E}$ with respect to some task $\mathcal{T}$ and some performance measure $\mathcal{P}$, if its performance on $\mathcal{T}$, as measured by $\mathcal{P}$, improves with experience $\mathcal{E}$.

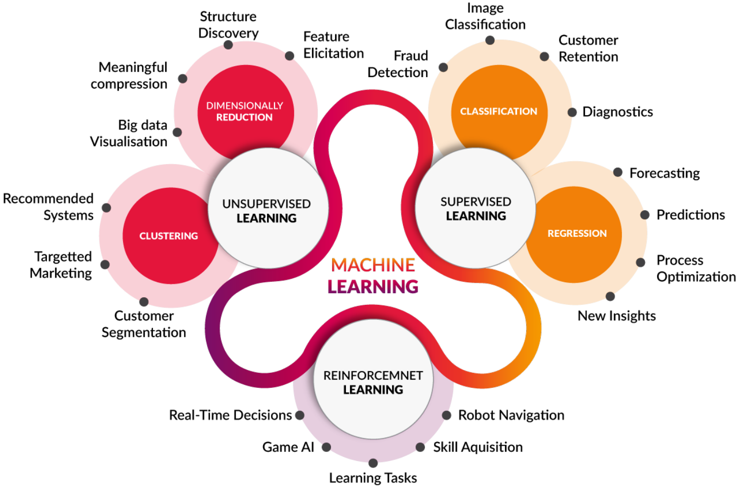
Task #1

Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task $\mathcal{T}$ in this setting?

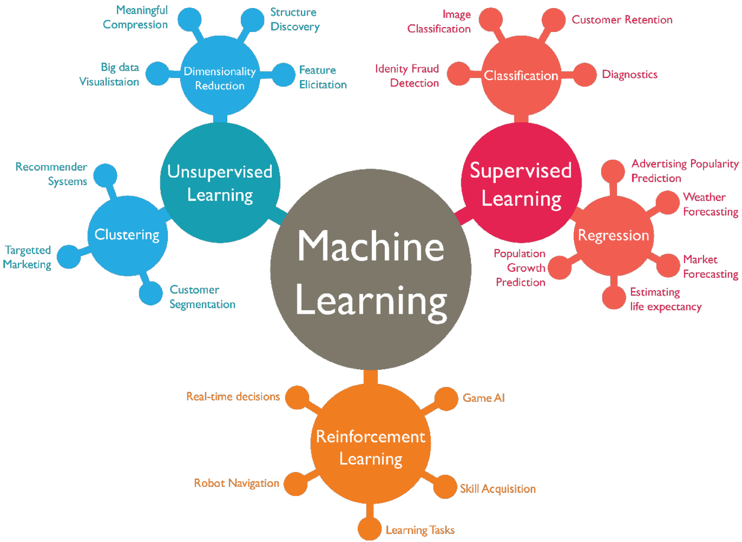
1. Classifying emails as spam or not spam;
2. Watching you label emails as spam or not spam;
3. The number (or fraction) of emails correctly classified as spam/not spam;
4. None of the above-this not a machine learning problem.

Overall Methodology

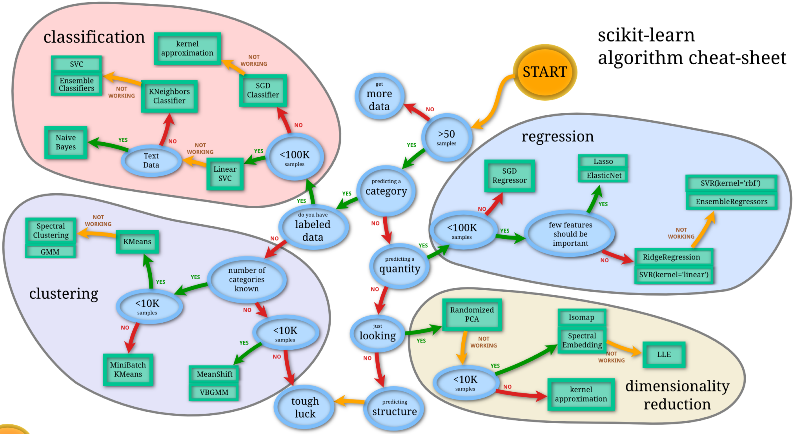
1. Define the problem;
2. Gather dataset;
3. Choose measure of success;
4. Decide evaluation protocol;
5. Prepare the data;
6. Develop a model;
7. Iterate models.



https://www.cognub.com/index.php/cognitive-platform/



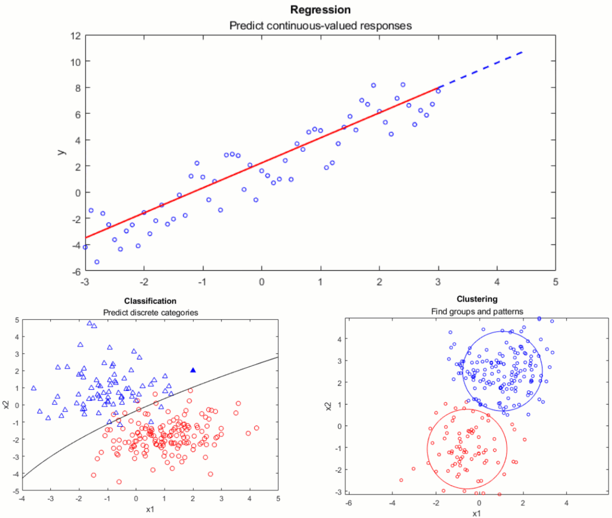
https://vitalflux.com/great-mind-maps-for-learning-machine-learning/



Back

https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

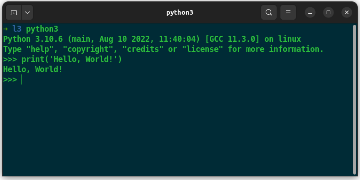
Regression | Classification | Clustering



https://github.com/MathWorks-Teaching-Resources/Machine-Learning-for-Regression

Programming Language

python.org/



Development Environments

- ▲ $ docker compose up
- ▼ $ docker compose down

Required Packages

Valid only for...

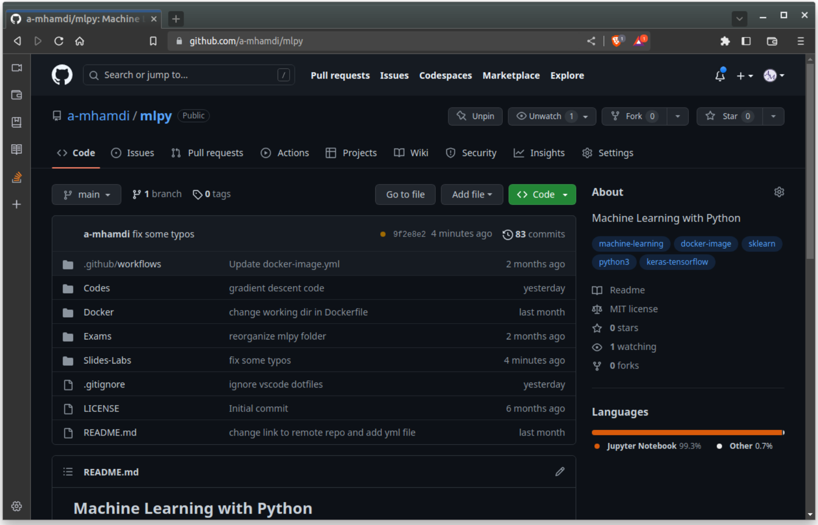
- A full list is available @ https://pypi.org/

Numpy *Matplotlib* *Pandas* *Scikit – learn* *Keras*

$ pip install virtualenv
$ virtualenv - -version
$ virtualenv «virtualenv_name»
$ source «virtualenv_name»/bin/activate # ACTIVATE
$ deactivate # DEACTIVATE

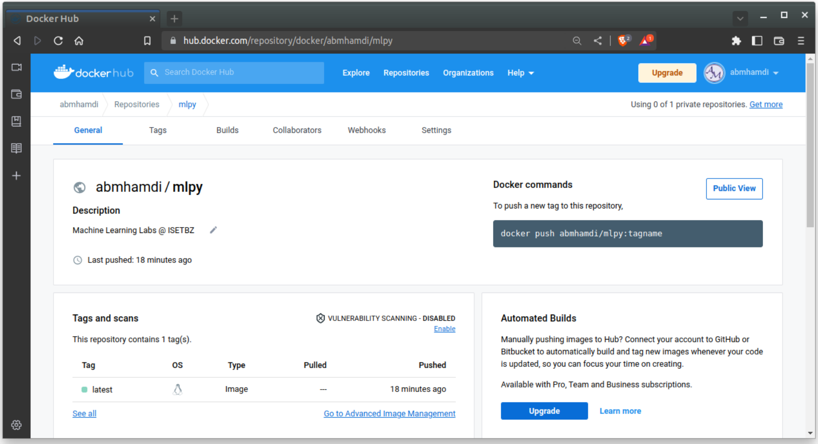
> pip install virtualenv
> virtualenv - -version
> virtualenv «virtualenv_name»
> «virtualenv_name»\Scripts\activate %= ACTIVATE =%
> deactivate %= DEACTIVATE =%

Source Control Management (SCM)



https://github.com/a-mhamdi/mlpy

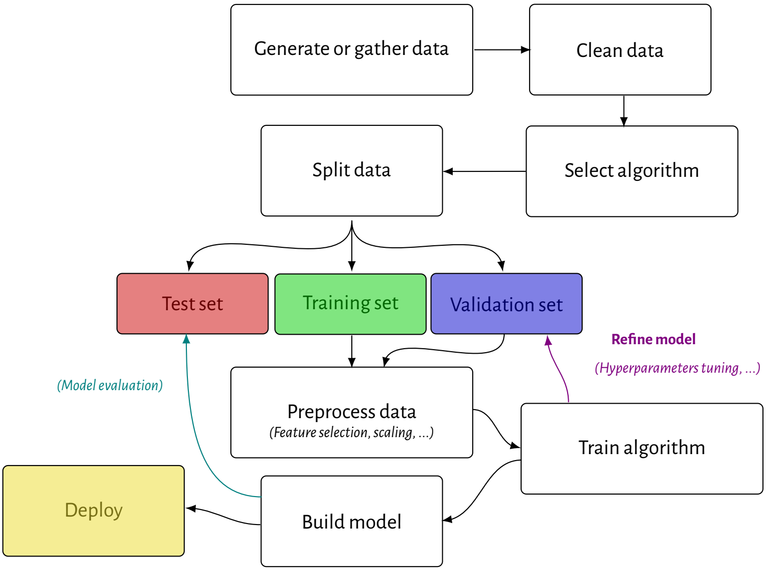
Continuous Integration (CI)



https://hub.docker.com/r/abmhamdi/mlpy

Next...

Workflow in Machine Learning



Data Preprocessing

How?

Cleaning Identifying and correcting or removing inaccuracies and inconsistencies in the data.

Transformation Converting data from one format or structure to another.

Normalization Scaling the data so that it fits within a specific range. This is often done to make the data more amenable to certain operations or algorithms.

Data Preprocessing

Why?

- Raw data is often messy and may need to be cleaned and formatted before it can be used for machine learning.
  (This may involve removing missing or invalid data, handling outliers, and encoding categorical variables.)
- Normalizing the data can help to scale the features so that they are on the same scale.
  (This can be important for algorithms that use distance measures, as features on different scales can dominate the distance measure.)
- Preprocessing techniques such as feature selection and feature extraction can help to reduce the dimensionality of the data.
  (This may improve the performance of the model and reduce the risk of overfitting.)
- Preprocessing techniques such as feature selection can help to identify the most important features in the data.
  (This can make the model more interpretable and easier to understand.)

Data Preprocessing

Feature Scaling

Normalization *(MinMaxScaler)*	**Standardization** *(StandardScaler)*
$X \triangleq \dfrac{X - X.\min()}{X.\max() - X.\min()}$	$X \triangleq \dfrac{X - \mu}{\sigma}$
▲ No assumption on data distribution	▲ More recommended when following normal distribution

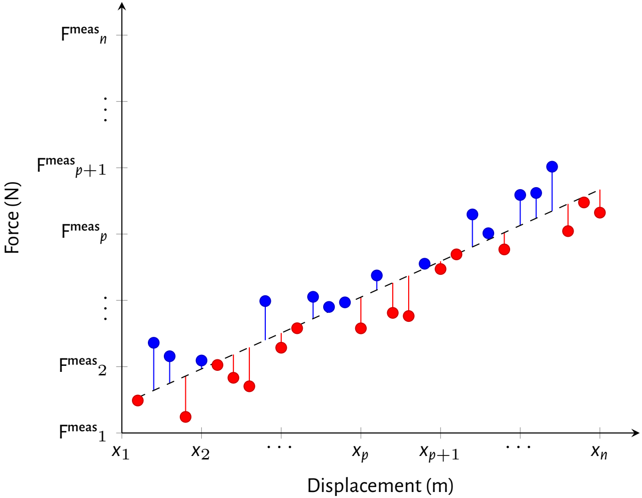
Data Preprocessing Template

[ ]:
```
from sklearn.preprocessing import StandardScaler
```

[ ]:
```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
⟶ *Codes* ⟶ *Python* ⟶ *data-preprocessing.ipynb*

This kind of supervised learning deals with labelled data. A subset of this data is used later to predict in continuous form. Regression problems involve tasks where the outputs form generally a set of real numbers. They oftenly follow linear formats.



Consider the example of a spring. Our main goal is to determine the stiffness k of this spring, given some experimental data. The mathematical model *(Hooke's law)*:

$$F = kx \tag{1}$$

Restoring force is proportional to displacement.

Table: Measurements of couple $(x_i,\ \mathrm{F^{meas}}_i)$

x_i	x_0	$\cdots$	x_p	$\cdots$	x_{n-1}
$\mathrm{F^{meas}}_i$	$\mathrm{F^{meas}}_0$	$\cdots$	$\mathrm{F^{meas}}_p$	$\cdots$	$\mathrm{F^{meas}}_{n-1}$

$$\begin{aligned} \mathrm{F^{meas}}_i &= \mathrm{F}_i + \varepsilon_i \\ &= kx_i + \varepsilon_i, \end{aligned} \tag{2}$$

where F_i denotes the unknown real value of the force applied to the spring. In order to estimate the stiffness value k, we can consider the quadratic criterion:

$$\begin{aligned} \mathcal{J} &= \sum_{i=0}^{n-1} \varepsilon_i^2 \\ &= \sum_{i=0}^{n-1} \left(\mathrm{F^{meas}}_i - kx_i\right)^2 \end{aligned}$$

$$\frac{\partial \mathcal{J}}{\partial k} = 0 \tag{3}$$

$$2\sum_{i=0}^{n-1}\left(\mathrm{F^{meas}}_i - kx_i\right)\sum_{i=0}^{n-1}\frac{\partial\left(\mathrm{F^{meas}}_i - kx_i\right)}{\partial k} = 0$$

$$\sum_{i=0}^{n-1}\left(\mathrm{F^{meas}}_i - kx_i\right)\sum_{i=0}^{n-1}x_i = 0$$

$$\sum_{i=0}^{n-1}\mathrm{F^{meas}}_i\, x_i = k\sum_{i=0}^{n-1}x_i^2 \iff \boxed{\hat{k} = \frac{\sum_{i=0}^{n-1}\mathrm{F^{meas}}_i\, x_i}{\sum_{i=0}^{n-1}x_i^2}}$$

Simple Linear Regression

CODE SNIPPET

Training the Simple Linear Regression model on the Training set

[ ]:
```
from sklearn.linear_model import LinearRegression
```

[ ]:
```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

[ ]:
```
LinearRegression()
```

Predicting the Test set results

[ ]:
```
y_pred = lr.predict(X_test)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
⟶ *Codes* ⟶ *Python* ⟶ *simple-linear-regression.ipynb*

Multiple Linear Regression

CODE SNIPPET

Training the multiple linear regression model on the training set

[ ]:
```
from sklearn.linear_model import LinearRegression
```

[ ]:
```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

[ ]:
```
LinearRegression()
```

Making predictions using the X test set and comparison

[ ]:
```
y_pred = lr.predict(X_test)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
$\longrightarrow$ *Codes* $\longrightarrow$ *Python* $\longrightarrow$ *multiple-linear-regression.ipynb*

Consider the following multivariable equation:

$$y \quad = \quad \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_m x_m \tag{4}$$

For a particular single measurement, eq. (4) can be updated as

$$y_k = \theta_0 x_{(0,k)} + \theta_1 x_{(1,k)} + \cdots + \theta_m x_{(m,k)} + \varepsilon_k \tag{5}$$

We denote hereafter by θ the vector $\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}$. The function y_k becomes:

$$y_k \quad = \quad \underbrace{\left[x_{(0,k)}, x_{(1,k)}, \cdots, x_{(m,k)}\right]}_{x_k^T} \theta + \varepsilon_k \tag{6}$$

We assume that we have n measurements for y. Then we can transform the previous equation into

$$y \quad = \quad X\theta + \varepsilon, \tag{7}$$

where $y^T = [y_0, y_1, \cdots, y_{n-1}]$, $X = \begin{bmatrix} x_0^T \\ x_1^T \\ \vdots \\ x_{n-1}^T \end{bmatrix}$ and $\varepsilon^T = [\varepsilon_0, \varepsilon_1, \cdots, \varepsilon_{n-1}]$.

An estimate of $\hat{\theta}$ is given by

$$\boxed{\hat{\theta} = \left(X^T X\right)^{-1} X^T y}$$

$X^T X$ **is not invertible (singular/degenerate)**

▼ Redundant Features

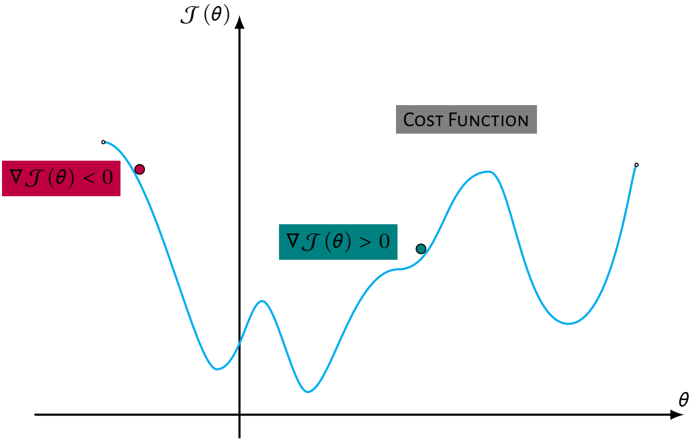
Some features are linearly dependent, i.e, $\exists$ some $x_p \propto$ some x_l.

▼ Too many features

Fewer observations compared to the number of features, i.e, $m > n$.

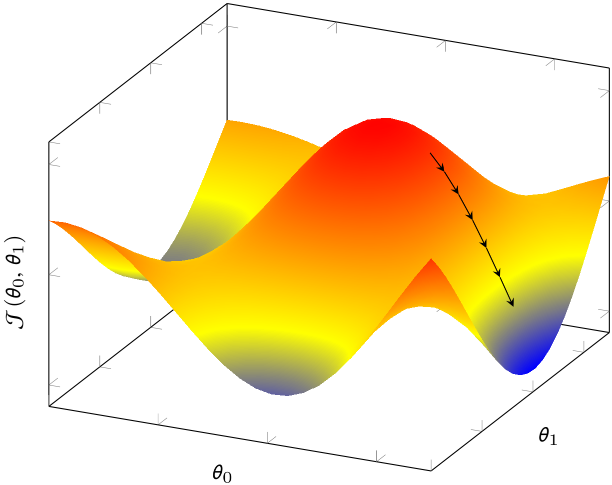
▲ Delete some features
▲ Add extra observations

Gradient Descent (1/3)



$$\underbrace{\theta}_{\text{New value}} \triangleq \underbrace{\theta}_{\text{Old value}} - \overbrace{\alpha}^{\text{Weighting factor}} \underbrace{\nabla \mathcal{J}(\theta)}_{\text{Gradient term is steepest ascent}}$$

Gradient Descent (2/3)



① Start with some random values of θ_0 and θ_1

② Keep changing θ_0 and θ_1 to reduce $\mathcal{J}(\theta_0, \theta_1)$ until we hopefully end up at minimum

Gradient Descent (3/3)

The linear regression is given by:

$$y_k = \underbrace{\underbrace{\left[x_{(0,k)}, x_{(1,k)}, \cdots, x_{(m,k)}\right]}_{x_k^T} \theta}_{h_\theta(x_k)} + \varepsilon_k \tag{8}$$

$$\boxed{\theta_0 \triangleq \theta_0 + \alpha \frac{1}{n} \sum_{k=0}^{n-1} \left(y_k - h_\theta\left(x_k\right)\right) x_{(0,k)}}$$

$$\boxed{\theta_1 \triangleq \theta_1 + \alpha \frac{1}{n} \sum_{k=0}^{n-1} \left(y_k - h_\theta\left(x_k\right)\right) x_{(1,k)}}$$

$$\vdots$$

$$\boxed{\theta_m \triangleq \theta_m + \alpha \frac{1}{n} \sum_{k=0}^{n-1} \left(y_k - h_\theta\left(x_k\right)\right) x_{(m,k)}}$$

The hyperparameter α is the learning rate.

Polynomial Regression

CODE SNIPPET

[ ]:
```
from sklearn.preprocessing import PolynomialFeatures
```

[ ]:
```
poly_reg = PolynomialFeatures(degree=4)
X_poly = poly_reg.fit_transform(X)
print(X_poly[:5])
lr_2 = LinearRegression()
lr_2.fit(X_poly, y)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
→ *Codes* → *Python* → *polynomial-regression.ipynb*

Task #2

The yield y of a chemical process is a random variable whose value is considered to be a linear function of the temperature x. The following data of corresponding values of x and y is found:

Temperature in °C (x)	0	25	50	75	100
Yield in grams (y)	14	38	54	76	95

The linear regression model $y = \theta_0 + \theta_1 x$ is used. Determine the values of $\theta_0,\ \theta_1$ using normal equation.

$$y = \begin{bmatrix} 14 \\ 38 \\ 54 \\ 76 \\ 95 \end{bmatrix} \quad \text{and} \quad X = \begin{bmatrix} 1 & 0 \\ 1 & 25 \\ 1 & 50 \\ 1 & 75 \\ 1 & 100 \end{bmatrix} \implies X^T X = \begin{bmatrix} 5 & 250 \\ 250 & 18750 \end{bmatrix}$$

$$\hat{\theta} = \begin{bmatrix} \hat{\theta}_0 \\ \hat{\theta}_1 \end{bmatrix} = \begin{bmatrix} 15.4 \\ 0.8 \end{bmatrix}$$

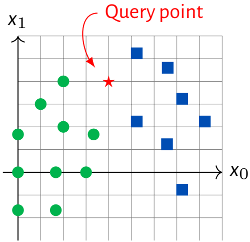
Code implementation

python

```
import numpy as np
import matplotlib.pyplot as plt

X = np.array([[1, 0], [1, 25], [1, 50], [1, 75], [1, 100]])
y = np.array([14, 38, 54, 76, 95])

# NORMAL EQUATION
theta_ne = np.linalg.inv(X.T @ X) @ X.T @ y

# GRADIENT DESCENT
theta_gd = np.zeros(shape=(2, 1001))
theta_gd[:, 0] = np.array([10, .5])
cost = []
for k in range(1000):
    eps = y-(X @ theta_gd[:, k])
    cost.append(1/10*(eps @ eps))
    theta_gd[:, k+1] = theta_gd[:, k] + .003/5*(eps @ X)

plt.plot(theta_gd[0, :], label=r'$\hat{\theta}_0$')
plt.plot(theta_gd[1, :], label=r'$\hat{\theta}_1$')
plt.legend(); plt.grid(); plt.show()

plt.plot(cost); plt.grid(); plt.show()
```

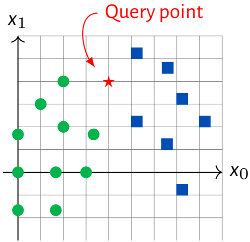
Introduction

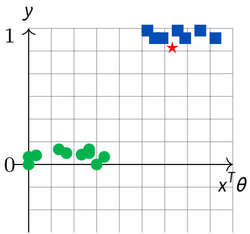
Classification is a type of supervised machine learning algorithm. A model is trained on a set of *labeled data*, where each data point is associated with a known class or category. The goal of the algorithm is to learn the relationship between the *input features* x and the corresponding *output classes* y, so that it can accurately predict the class of new, unseen query points.



Introduction

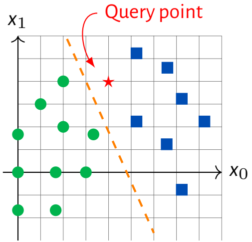
Classification is a type of supervised machine learning algorithm. A model is trained on a set of *labeled data*, where each data point is associated with a known class or category. The goal of the algorithm is to learn the relationship between the *input features* x and the corresponding *output classes* y, so that it can accurately predict the class of new, unseen query points.

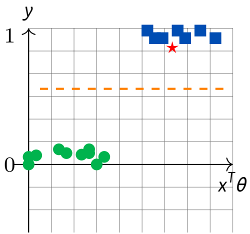




Introduction

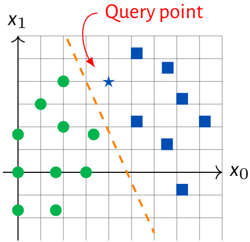
Classification is a type of supervised machine learning algorithm. A model is trained on a set of *labeled data*, where each data point is associated with a known class or category. The goal of the algorithm is to learn the relationship between the *input features* x and the corresponding *output classes* y, so that it can accurately predict the class of new, unseen query points.

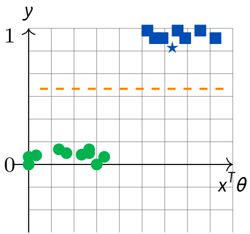




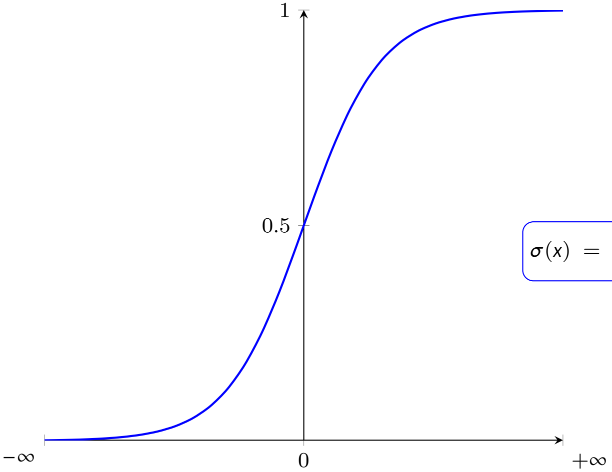
Introduction

Classification is a type of supervised machine learning algorithm. A model is trained on a set of *labeled data*, where each data point is associated with a known class or category. The goal of the algorithm is to learn the relationship between the *input features* x and the corresponding *output classes* y, so that it can accurately predict the class of new, unseen query points.





Logistic or S-shaped function σ



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

- σ squashes range of distance from $]-\infty, +\infty[$ to $[0, 1]$
- σ is differentiable and easy to compute: $\boxed{\dot{\sigma} = \sigma \times (1 - \sigma)}$

Decision boundary

$$y = \sigma\left(\theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_m x_m\right)$$

$$y = \frac{1}{1+\mathrm{e}^{-x^T\theta}}$$

Hypothesis

Considering $h_\theta(x) = \dfrac{1}{1+\mathrm{e}^{-x^T\theta}}$ yields $h_\theta(x_k) = \dfrac{1}{1+\mathrm{e}^{-x_k^T\theta}}$

GRADIENT DESCENT

$$\vdots$$

$$\boxed{\theta_m \triangleq \theta_m + \alpha \frac{1}{n}\sum_{k=0}^{n-1}\left(y_k - h_\theta(x_k)\right) x_{(m,k)}}$$

Logistic Regression

CODE SNIPPET

Training the logistic regressor

[ ]:
```
from sklearn.linear_model import LogisticRegression
```

[ ]:
```
clf = LogisticRegression(random_state=123)
clf.fit(X_train, y_train)
```

[ ]:
```
LogisticRegression(random_state=123)
```

Predicting the test set results

[ ]:
```
y_pred = clf.predict(X_test)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
$\longrightarrow$ *Codes* $\longrightarrow$ *Python* $\longrightarrow$ *logistic-regression.ipynb*

Confusion Matrix

		Actual	
		Positive	Negative
Predicted	Positive	**TP**	**FP**
	Negative	**FN**	**TN**

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

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

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

$$f1 - \text{score} = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

Confusion Matrix

		Actual	
		Positive	Negative
Predicted	Positive	**TP**	**FP**
	Negative	**FN**	**TN**

$$\begin{aligned} \mathit{Accuracy} &= \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \\ \mathit{Precision} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \\ \mathit{Recall} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \\ f1 - \mathrm{score} &= \frac{2}{\frac{1}{\mathit{Precision}} + \frac{1}{\mathit{Recall}}} \end{aligned}$$

183	141
13	663

$$\begin{aligned} \mathit{Accuracy} &= 0.846 \\ \mathit{Precision} &= 0.565 \\ \mathit{Recall} &= 0.934 \\ f1 - \mathrm{score} &= 0.704 \end{aligned}$$

320	20
43	538

$$\begin{aligned} \mathit{Accuracy} &= 0.932 \\ \mathit{Precision} &= 0.941 \\ \mathit{Recall} &= 0.882 \\ f1 - \mathrm{score} &= 0.910 \end{aligned}$$

Evaluation metrics

Accuracy **Precision** **Recall** **$f1$-score**

Accuracy denotes the ratio of how much we got right over all cases:

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

Precision designates how much positives do we get right over all positive predictions:

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

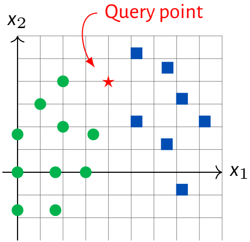
Recall is the ratio of how much positives we got right over all actual positive cases:

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

$f1 -$ score denotes the Harmonic Mean of *Precision* & *Recall*:

$$f1 - \text{score} = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

k-Nearest Neighbors (1/5)



Evelyn Fix and Joseph Hodges, 1951 | Thomas Cover, 1966

Algorithm Summary Construction

1: **procedure** How does k-NN work? (Finding Nearest Neighbors)
Input: A query point;
Output: Assign a class label to that point.
2: Define how many neighbors will be checked to classify the specific query point;
3: Compute the distance $d(x;\ y)$ of the query point to other data points;
4: Count the number of the data points in each category;
5: Assign the query point to the class with most frequent neighbors.
6: **end procedure**

k-Nearest Neighbors (2/5)

Minkowski distance
$$d(x;\, y) = \left(\sum_{i=0}^{n-1} |y_i - x_i|^p\right)^{1/p}$$

Manhattan distance (p=1)
$$d(x;\, y) = \sum_{i=0}^{n-1} |y_i - x_i|$$

Euclidean distance (p=2)
$$d(x;\, y) = \sqrt{\sum_{i=0}^{n-1} (y_i - x_i)^2}$$

Task #3

Let be the following coordinate points:

$$A(1,\, 6);\ B(2,\, 6);\ C(3,\, 1);\ D(4,\, 2);\ E(6,\, 0);\ F(7,\, 5);\ G(7,\, 3);\ H(10,\, 3);\ I(-4,\, -1)$$

Using the Euclidean distance, what are the two closest neighbors of point $P(5,\, 5)$?

$d(A;\, P) = \sqrt{17} \approx 4.12$ $\quad d(B;\, P) = \sqrt{10} \approx 3.16$ $\quad d(C;\, P) = \sqrt{20} \approx 4.47$

$d(D;\, P) = \sqrt{10} \approx 3.16$ $\quad d(E;\, P) = \sqrt{26} \approx 5.1$ $\quad d(F;\, P) = \sqrt{4} = 2$

$d(G;\, P) = \sqrt{8} \approx 2.83$ $\quad d(H;\, P) = \sqrt{29} \approx 5.38$ $\quad d(I;\, P) = \sqrt{117} \approx 10.82$

k-Nearest Neighbors (3/5)

```
from math import sqrt
def dds(a, b): # `a` and `b` are coordinates of some point
    d_squared = (a-5)**2+(b-5)**2
    return (d_squared, sqrt(d_squared))

dds(1, 6) # Point `A`
dds(2, 6) # Point `B`
```

k-Nearest Neighbors (4/5)

Task #4[a]

[a]From Prof. Winston's book

We try to predict the color of a fruit according to its width (w) and height (h). The following training data is available:

Fruit	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
w	2	5	2	6	1	4	2	6
h	6	6	5	5	2	2	1	1
Color	Red	Yellow	Orange	Purple	Red	Blue	Violet	Green

The goal here is to study the influence of neighbors on the color property of a fruit. Let U be the new fruit of width $w = 1$ and height $h = 4$

1. What is its color if we consider 1 neighbor?
2. What is its color if we consider 3 neighbors?
3. Rather than majority voting, we would like to consider the vote of neighbors weighted by the distance. Each neighbor votes according to a weight inversely proportional to the square of its distance: $\frac{1}{d^2}$. We take 3 neighbors, what is the color of U? Compare your results to those in question 2.

k-Nearest Neighbors (5/5)

$d(U;\ F_1) = \sqrt{5} \approx 2.24$ $d(U;\ F_2) = \sqrt{20} \approx 4.47$ $d(U;\ F_3) = \sqrt{2} \approx 1.41$

$d(U;\ F_4) = \sqrt{26} \approx 5.1$ $d(U;\ F_5) = \sqrt{4} = 2$ $d(U;\ F_6) = \sqrt{13} \approx 3.6$

$d(U;\ F_7) = \sqrt{10} \approx 3.16$ $d(U;\ F_8) = \sqrt{34} \approx 5.83$

1. Color of U is Orange because $d(U;\ F_3)$ is the smallest.
2. Color of U is Red: F_1 and F_5 *(+2 to Red class)*, F_3 *(+1 to Orange class)*
3. Color of U is Orange

$$\boxed{\mathcal{S}(\text{Red}) = \frac{1}{d^2(U;\ F_1)} + \frac{1}{d^2(U;\ F_5)} = 0.45} \quad \boxed{\mathcal{S}(\text{Orange}) = \frac{1}{d^2(U;\ F_3)} = 0.5}$$

```python
from math import sqrt
def dds(w, h): # `w` and `h` are width and height of some fruit
    d_squared = (w-1)**2+(h-4)**2
    return (d_squared, sqrt(d_squared))

dds(2, 6) # Fruit `F_1`
dds(5, 6) # Fruit `F_2`
```

k-NN

CODE SNIPPET

Importing the classifier

```
[ ]: from sklearn.neighbors import KNeighborsClassifier
```

Training the K-NN model on the training set

```
[ ]: clf = KNeighborsClassifier(n_neighbors, metric, p)
     clf.fit(X_train, y_train)
```

Predicting the test set results

```
[ ]: y_pred = clf.predict(X_test)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
⟶ *Codes* ⟶ *Python* ⟶ *k-nearest-neighbors.ipynb*

Rule of Thumb to Choose k

k is even if the number of classes is odd

k is odd if the number of classes is even

k is an important hyperparameter that can affect the performance of the model.

1. Larger values of k will result in a smoother decision boundary, which can lead to a more generalized model.
2. Smaller values of k will result in a more complex decision boundary, which can lead to a model that is more prone to overfitting.
3. The optimal value of K may depend on the specific dataset and the characteristics of the data.

Outroduction

Method		Pros		Cons
Logistic Regression	▲	Probabilistic	▼	Almost linearly separable data
K-NN	▲ ▲ ▲	Simple Fast Efficient	▼ ▼	Number of neighbors k Detecting outliers[2]

[2]Points that differ significantly from the rest of the data points.

Next...

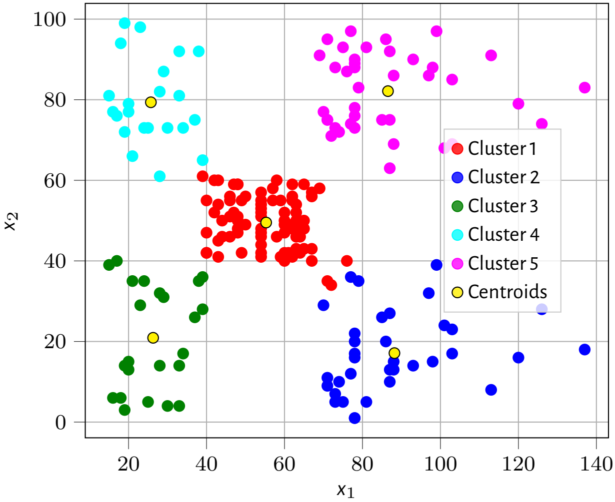
K-Means Clustering (1/3)

The algorithm *K*-**Means** allows to display regularities or patterns in unlabeled data.

- The term 'means' refers to averaging the data when computing each centroid;
- A centroid is the arithmetic mean of all the data points belonging to a particular cluster.

This technique identifies a certain number of centroids within a data set. The algorithm then allocates every data point to the nearest cluster as it attempts to keep the clusters as small as possible. At the same time, *K*-Means attempts to keep the other clusters as different as possible.

K-Means Clustering (2/3)



K-Means Clustering (3/3)

Algorithm Summary Construction

procedure HOW DOES *K*-MEANS WORK? (Discovering similarities)
Input: Unlabeled data sets;
Output: Grouping into clusters.
    Define how many clusters will be used to group the data sets;
    Initialize all the coordinates of the k cluster centers
    repeat
        Assign each point to its nearest cluster;
        Update the centroids coordinates;
    until No changes to the centers of the clusters
    Assign new cases to one of the clusters
end procedure

Task #5[a]

[a] From 'Machine Learning' course on 'Coursera'

Of the following examples, which would you address using an unsupervised learning algorithms? *(Check all that apply.)*

1. Given email labeled as spam/not spam, learn a spam filter
2. Given a set of news articles found on the web, group them into set of articles about the same story
3. Given a database of customer data, automatically discover market segments and group customers into different market segments
4. Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not.

Task #5[a]

[a] From 'Machine Learning' course on 'Coursera'

Of the following examples, which would you address using an unsupervised learning algorithms? *(Check all that apply.)*

1. Given email labeled as spam/not spam, learn a spam filter
2. Given a set of news articles found on the web, group them into set of articles about the same story
3. Given a database of customer data, automatically discover market segments and group customers into different market segments
4. Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not.

Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10); \beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) \ = \ |x_m - x_n| + |y_m - y_n|$$

Task #6[a]

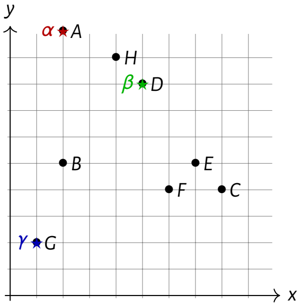
[a]Credit: Shokoufeh Mirzaei, PhD

Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) \ = \ |x_m - x_n| + |y_m - y_n|$$



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

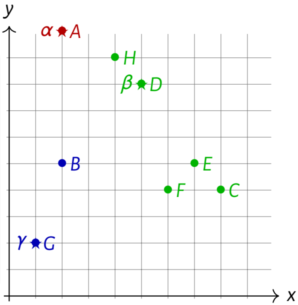
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) = |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(2,\ 10)$	$\beta(5,\ 8)$	$\gamma(1,\ 2)$	#
$A(2,\ 10)$	0	5	9	1
$B(2,\ 5)$	5	6	4	3
$C(8,\ 4)$	12	7	9	2
$D(5,\ 8)$	5	0	10	2
$E(7,\ 5)$	10	5	9	2
$F(6,\ 4)$	10	5	7	2
$G(1,\ 2)$	9	10	0	3
$H(4,\ 9)$	3	2	10	2



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

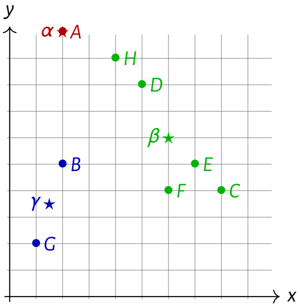
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N)\ =\ |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(2,\ 10)$	$\beta(5,\ 8)$	$\gamma(1,\ 2)$	#
$A(2,\ 10)$	0	5	9	1
$B(2,\ 5)$	5	6	4	3
$C(8,\ 4)$	12	7	9	2
$D(5,\ 8)$	5	0	10	2
$E(7,\ 5)$	10	5	9	2
$F(6,\ 4)$	10	5	7	2
$G(1,\ 2)$	9	10	0	3
$H(4,\ 9)$	3	2	10	2
	$\alpha(2,\ 10)$	$\beta(6,\ 6)$	$\gamma(1.5,\ 3.5)$	



Task #6[a]

[a] Credit: Shokoufeh Mirzaei, PhD

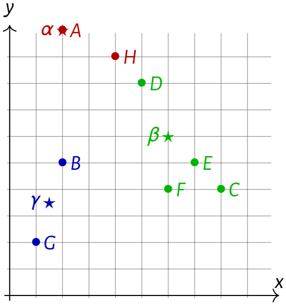
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) \ = \ |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(2,\ 10)$	$\beta(6,\ 6)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	0	8	7	1
$B(2,\ 5)$	5	5	2	3
$C(8,\ 4)$	12	4	7	2
$D(5,\ 8)$	5	3	8	2
$E(7,\ 5)$	10	2	7	2
$F(6,\ 4)$	10	2	5	2
$G(1,\ 2)$	9	9	2	3
$H(4,\ 9)$	3	5	8	1



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

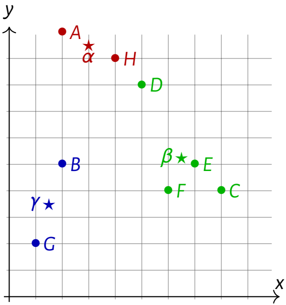
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5);\ C(8,\ 4);\ D(5,\ 8);\ E(7,\ 5);\ F(6,\ 4);\ G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) = |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(2,\ 10)$	$\beta(6,\ 6)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	0	8	7	1
$B(2,\ 5)$	5	5	2	3
$C(8,\ 4)$	12	4	7	2
$D(5,\ 8)$	5	3	8	2
$E(7,\ 5)$	10	2	7	2
$F(6,\ 4)$	10	2	5	2
$G(1,\ 2)$	9	9	2	3
$H(4,\ 9)$	3	5	8	1
$\boxed{\alpha(3,\ 9.5)}$		$\boxed{\beta(6.5,\ 5.25)}$	$\boxed{\gamma(1.5,\ 3.5)}$	



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

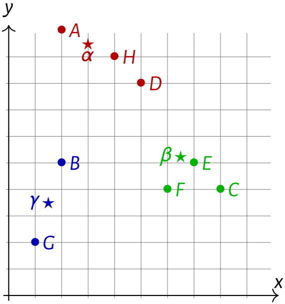
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) \ = \ |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(3,\ 9.5)$	$\beta(6.5,\ 5.25)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	1.5	9.25	7	1
$B(2,\ 5)$	5.5	4.75	2	3
$C(8,\ 4)$	10.5	2.75	7	2
$D(5,\ 8)$	3.5	4.25	8	1
$E(7,\ 5)$	8.5	0.75	7	2
$F(6,\ 4)$	8.5	1.75	5	2
$G(1,\ 2)$	9.5	8.75	2	3
$H(4,\ 9)$	1.5	6.25	8	1



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

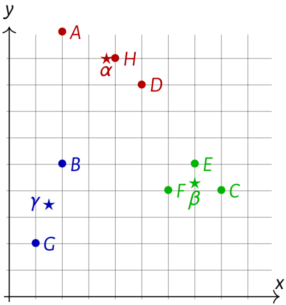
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10); \beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) \ = \ |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(3,\ 9.5)$	$\beta(6.5,\ 5.25)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	1.5	9.25	7	1
$B(2,\ 5)$	5.5	4.75	2	3
$C(8,\ 4)$	10.5	2.75	7	2
$D(5,\ 8)$	3.5	4.25	8	1
$E(7,\ 5)$	8.5	0.75	7	2
$F(6,\ 4)$	8.5	1.75	5	2
$G(1,\ 2)$	9.5	8.75	2	3
$H(4,\ 9)$	1.5	6.25	8	1
	$\alpha(3.67,\ 9)$	$\beta(7,\ 4.3)$	$\gamma(1.5,\ 3.5)$	



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

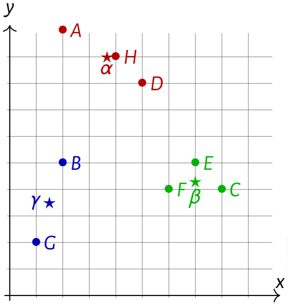
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5); C(8,\ 4); D(5,\ 8); E(7,\ 5); F(6,\ 4); G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10); \beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) = |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(3.67,\ 9)$	$\beta(7,\ 4.3)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	2.67	10.7	7	1
$B(2,\ 5)$	5.67	5.7	2	3
$C(8,\ 4)$	9.33	1.3	7	2
$D(5,\ 8)$	2.33	5.7	8	1
$E(7,\ 5)$	7.33	0.7	7	2
$F(6,\ 4)$	7.33	1.3	5	2
$G(1,\ 2)$	9.67	8.3	2	3
$H(4,\ 9)$	0.33	7.7	8	1



Task #6[a]

[a]Credit: Shokoufeh Mirzaei, PhD

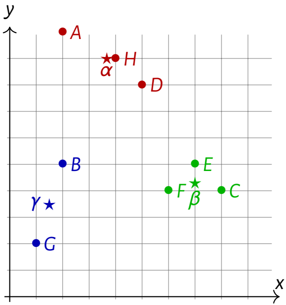
Use *K*-Means algorithm to cluster the following eight points intro three clusters:

$$A(2,\ 10);\ B(2,\ 5);\ C(8,\ 4);\ D(5,\ 8);\ E(7,\ 5);\ F(6,\ 4);\ G(1,\ 2) \text{ and } H(4,\ 9).$$

- Initial cluster centers are: $\alpha(2,\ 10)$; $\beta(5,\ 8)$ and $\gamma(1,\ 2)$
- The distance between two points: $M(x_m,\ y_m)$ and $N(x_n,\ y_n)$ is defined as

$$d(M;\ N) = |x_m - x_n| + |y_m - y_n|$$

Point	$\alpha(3.67,\ 9)$	$\beta(7,\ 4.3)$	$\gamma(1.5,\ 3.5)$	#
$A(2,\ 10)$	2.67	10.7	7	1
$B(2,\ 5)$	5.67	5.7	2	3
$C(8,\ 4)$	9.33	1.3	7	2
$D(5,\ 8)$	2.33	5.7	8	1
$E(7,\ 5)$	7.33	0.7	7	2
$F(6,\ 4)$	7.33	1.3	5	2
$G(1,\ 2)$	9.67	8.3	2	3
$H(4,\ 9)$	0.33	7.7	8	1
	$\alpha(3.67,\ 9)$	$\beta(7,\ 4.3)$	$\gamma(1.5,\ 3.5)$	



K-Means

CODE SNIPPET

Import `KMeans` **class**

```
[ ]: from sklearn.cluster import KMeans
```

Training the K-Means model on the dataset

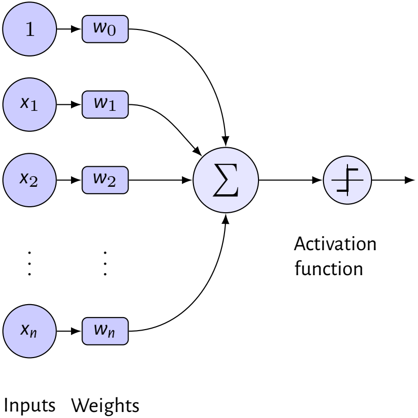
```
[ ]: kmeans = KMeans(n_clusters, init, random_state)
     y_pred = kmeans.fit_predict(X)
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
⟶ *Codes* ⟶ *Python* ⟶ *k-means-clustering.ipynb*

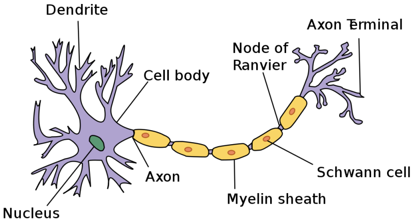
Next...

Fundamental unit of a neural network (1/3)

Artificial neuron



Biological neuron

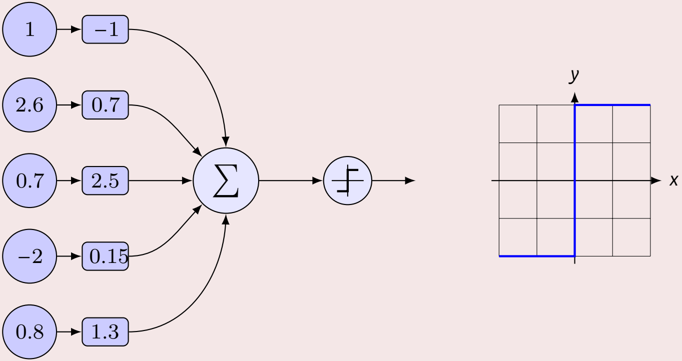


https://id.wikipedia.org/wiki/Sel_saraf

Fundamental unit of a neural network (2/3)

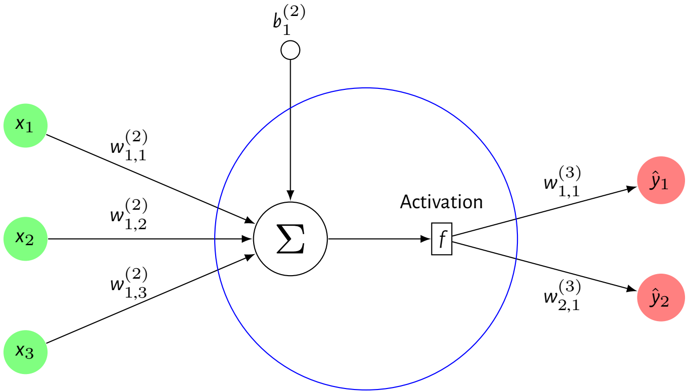
Task #7

Compute the output of the following neuron.

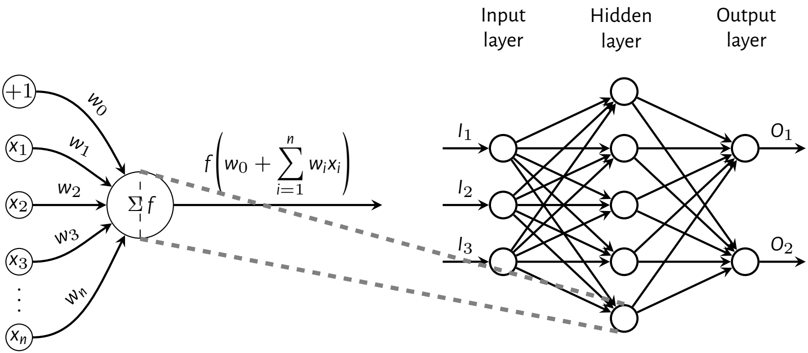


$$y = \text{sign}(1 \times -1 + 2.6 \times 0.7 + 0.7 \times 2.5 - 2 \times 0.15 + 0.8 \times 1.3) = 1$$

Fundamental unit of a neural network (3/3)



Multilayer Perceptron (MLP)

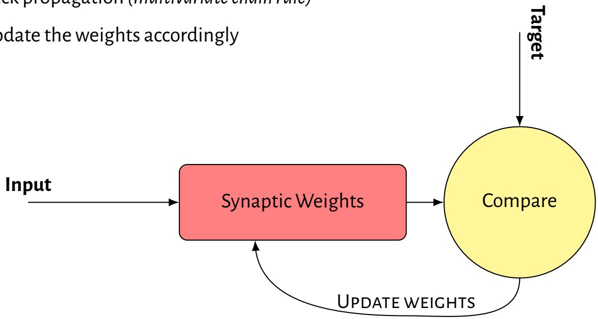


Task #8

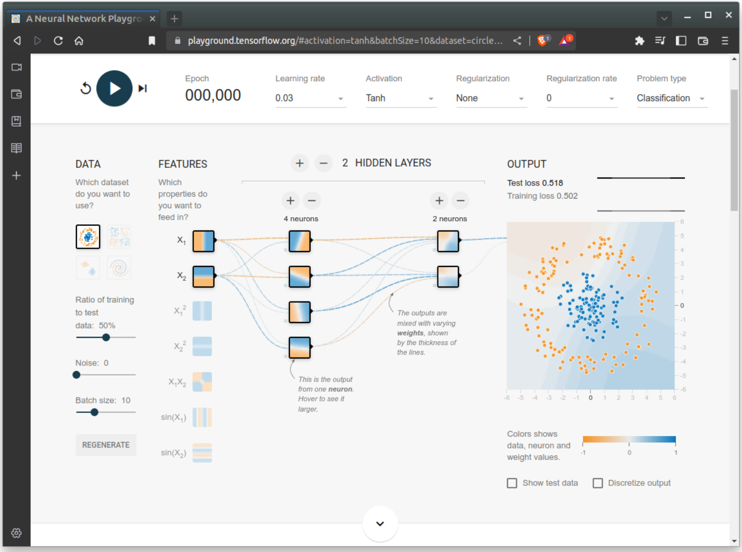
For the above structure, determine how many parameters are to be adjusted.

$$\#\,\text{params} \;=\; 5 \times 3 + 5 + 2 \times 5 + 2 \;=\; 32$$

- ✓ Design a structure
- ✓ Specify a loss function to minimize
- ✓ Optimize using gradient descent
  1. Feedforward propagation *(matrix multiplication and point-wise activation)*
  2. Back propagation *(multivariate chain rule)*
  3. Update the weights accordingly



Tinker with a neural network



https://playground.tensorflow.org/

ANN

CODE SNIPPET

```
[ ]: from keras.models import Sequential
     from keras.layers import Dense
```

```
[ ]: clf = Sequential()
     ndim = X_train.shape[1]
     clf.add(Dense(units=8, activation="relu", input_dim=ndim))
     clf.add(Dense(units=4, activation="relu"))
     clf.add(Dense(units=4, activation="relu"))
     clf.add(Dense(units=1, activation="sigmoid"))
```

```
[ ]: clf.compile(optimizer="adam", loss="binary_crossentropy",
             metrics=["accuracy"])
```

```
[ ]: clf.fit(X_train, y_train, batch_size=16, epochs=32);
```

The notebook is available at https://github.com/a-mhamdi/mlpy/
$\longrightarrow$ *Codes* $\longrightarrow$ *Python* $\longrightarrow$ *artificial-neural-network.ipynb*

List of available optimizers

Here is a list of some common optimizers for artificial neural networks:

$$\boxed{\Delta\hat{\mathcal{W}} \triangleq \mathcal{F}\left(\nabla \underbrace{\mathcal{J}\left(\hat{\mathcal{W}}\right)}_{\text{Loss Function}}\right)} \equiv \boxed{\hat{\mathcal{W}} \triangleq \hat{\mathcal{W}} + \mathcal{F}\left(\nabla\mathcal{J}\left(\hat{\mathcal{W}}\right)\right)} \quad \nabla\mathcal{J}\left(\hat{\mathcal{W}}\right) = \begin{bmatrix} \dfrac{\partial \mathcal{J}}{\partial \hat{w}_0} \\ \vdots \\ \dfrac{\partial \mathcal{J}}{\partial \hat{w}_n} \end{bmatrix}$$

SGD

SGD+MOMENTUM

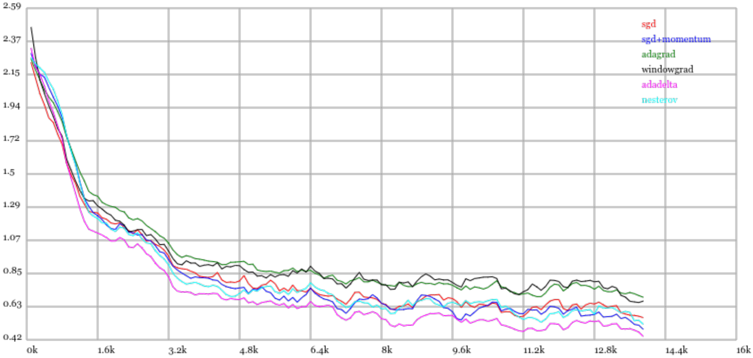
ADAGRAD

WINDOWGRAD

ADADELTA

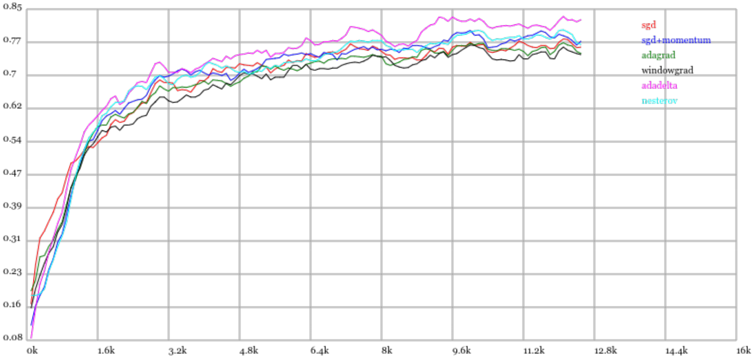
NESTEROV

Effect of optimizer on loss values



https://cs.stanford.edu/people/karpathy/convnetjs/demo/trainers.html

Effect of optimizer on testing accuracy values



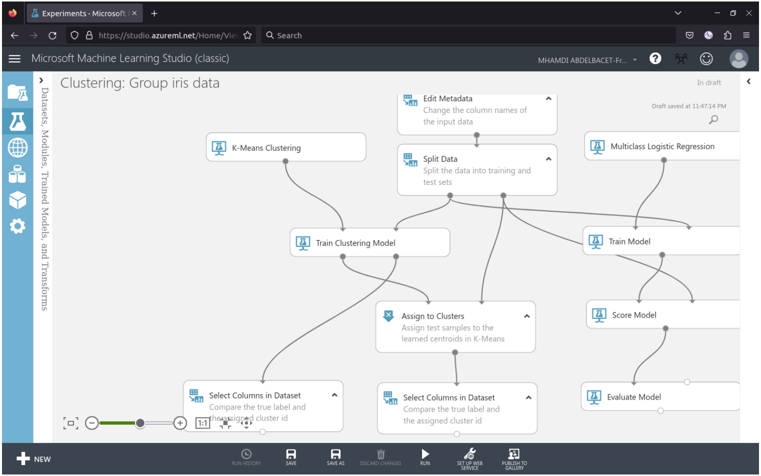
https://cs.stanford.edu/people/karpathy/convnetjs/demo/trainers.html

Next...

On the day of assignment, you will be informed about the **dataset to consider**, **specific features to keep**, and **name of machine learning model to build**. You will be asked to:

① conduct the experiment successfully *(pipeline, featurization, split, etc.)*;

② deploy a fully functional web service app that meets the given specifications.

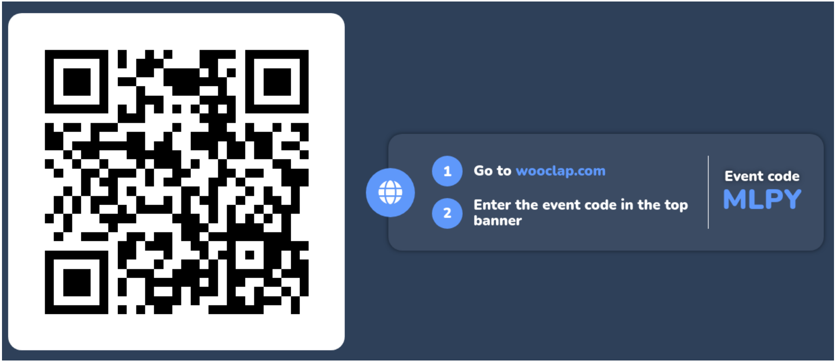


https://studio.azureml.net/

DEMO!

Next...

Knowledge Check



https://app.wooclap.com/MLPY

Link Bundle

https://karpathy.ai/ https://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

http://yann.lecun.com/ https://www.ibm.com/downloads/cas/GB8ZMQZ3

https://www.hackingnote.com/ https://stanford.edu/ shervine/teaching/

https://machinelearningmastery.com/

Further Reading (1/2)

[Bur19] A. Burkov. *The Hundred-Page Machine Learning Book*. Andriy Burkov, Jan. 1, 2019. 160 pp.

[Bur20] A. Burkov. *Machine Learning Engineering*. True Positive Inc., Sept. 8, 2020. 310 pp.

[DFO20] M. P. Deisenroth, A. A. Faisal, and C. S. Ong. *Mathematics for Machine Learning*. Cambridge University Pr., Apr. 1, 2020. 398 pp.

[ENM15] I. El Naqa and M. J. Murphy. “What Is Machine Learning?” In: *Machine Learning in Radiation Oncology: Theory and Applications*. Ed. by I. El Naqa, R. Li, and M. J. Murphy. Cham: Springer International Publishing, 2015, pp. 3–11. DOI: 10.1007/978-3-319-18305-3_1.

[Fla12] P. Flach. “References”. In: *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, Sept. 2012, pp. 367–382. DOI: 10.1017/CBO9780511973000.017.

[GBC16] I. Goodfellow, J. Bengio, and A. Courville. *Deep Learning*. MIT Press Ltd, Nov. 18, 2016. 800 pp.

[Gé19] A. Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O’Reilly Media, Oct. 15, 2019. 819 pp.

[HYU21] T. J. Hui (York University. *Machine Learning Fundamentals*. Cambridge University Press, Nov. 25, 2021. 420 pp.

[Jia22] H. Jiang. *Machine Learning Fundamentals*. Cambridge University Pr., Jan. 31, 2022.

[JPM21] L. M. John Paul Mueller. *Machine Learning For Dummies*. Wiley John + Sons, Apr. 8, 2021. 464 pp.

Further Reading (2/2)

[Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill, 1997.

[Pra18] M. L. de Prado. *Advances in Financial Machine Learning*. John Wiley & Sons Inc, May 4, 2018. 400 pp.

[SG16] A. C. M. Sarah Guido. *Introduction to Machine Learning with Python*. O'Reilly Media, July 31, 2016.

[Woj12] J. Wojtusiak. "Machine Learning". In: *Encyclopedia of the Sciences of Learning*. Springer US, 2012, pp. 2082–2083. DOI: 10.1007/978-1-4419-1428-6_1927.