

# Introduction to Scikit-Learn: Machine Learning with Python

Machine Learning Intro

郭耀仁

## About Scikit-Learn

Scikit-Learn (<http://scikit-learn.org/stable/>) is a Python package designed to give access to **well-known** machine learning algorithms within Python code, through a **clean, well-thought-out API**. It has been built by hundreds of contributors from around the world, and is used across industry and academia.

Scikit-Learn is built upon Python's [NumPy \(Numerical Python\)](http://numpy.org) and [SciPy \(Scientific Python\)](http://scipy.org) libraries, which enable efficient in-core numerical and scientific computation within Python.

**What is Machine Learning?**

Machine Learning is about building programs with **tunable parameters** (typically an array of floating point values) that are adjusted automatically so as to improve their behavior by **adapting to previously seen data**.

Tom Mitchell (<http://www.cs.cmu.edu/~tom/>).

*A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ .*

Machine Learning can be considered a subfield of **Artificial Intelligence** since those algorithms can be seen as building blocks to make computers learn to behave more intelligently by somehow **generalizing** rather than just storing and retrieving data items like a database system would do.



## Artificial Intelligence

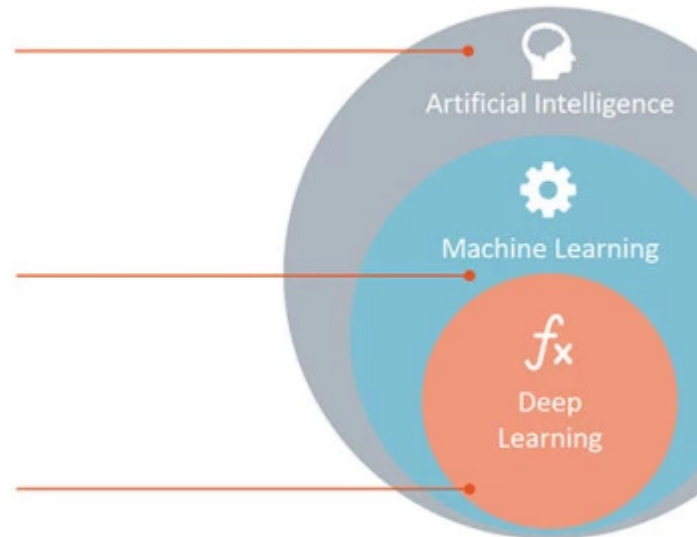
Any technique which enables computers to mimic human behavior.

## Machine Learning

Subset of AI techniques which use statistical methods to enable machines to improve with experiences.

## Deep Learning

Subset of ML which make the computation of multi-layer neural networks feasible.



Source: [rapidminer](https://rapidminer.com/artificial-intelligence-machine-learning-deep-learning/) (<https://rapidminer.com/artificial-intelligence-machine-learning-deep-learning/>).

# Machine Learning is

- Constructing/using algorithms that learn from data
- Performing better with more information
- Learning from previous solutions as experience

# Formulating Machine Learning

- input  $\rightarrow$  Function( $f$ )  $\rightarrow$  output
- input  $\rightarrow$  Estimated Function( $\hat{f}$ )  $\rightarrow$  predicted output

# Machine Learning Topics

- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
  - Dimensionality Reduction
  - Clustering

# Machine Learning Topics in Depth

- Supervised Learning: Finding function  $\hat{f}$  which can be used to assign a class or value to unseen observations given a set of labeled observations.
- Unsupervised Learning: Finding groups observation that are similar or using less variables to achieve similar predictability.

## Measuring performance

- Supervised Learning: Comparing real labels with predicted labels, great performance means predictions should be similar to real labels
- Unsupervised Learning: No real labels to compare

## **Representation of Data in Scikit-learn**

## **Machine learning is about**

- Creating models from data
- How data can be represented in order to be understood by the computer?



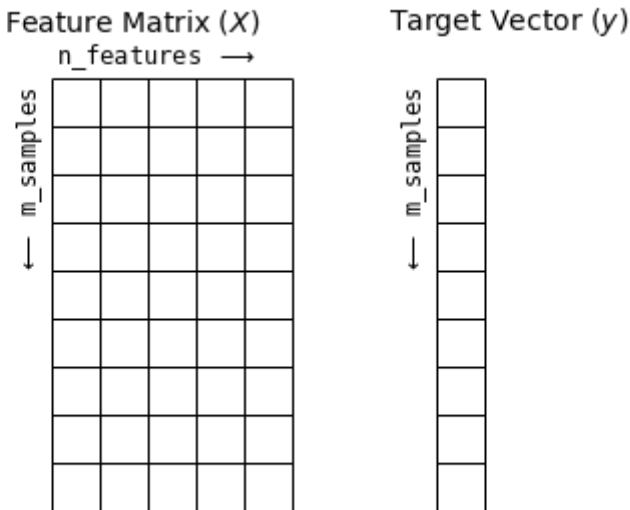
## Two-dimensional array or matrix

Most machine learning algorithms implemented in scikit-learn expect data to be stored in a **two-dimensional array or matrix**. The size of the array is expected to be `[m_samples, n_features]`

## m x n in general

- **m\_samples:** The number of samples: each sample is an item to process (e.g. classify). A sample can be a document, a picture, a sound, a video, an astronomical object, a row in database or CSV file, or whatever you can describe with a fixed set of quantitative traits.
- **n\_features:** The number of features or distinct traits that can be used to describe each item in a quantitative manner. Features are generally real-valued, but may be boolean or discrete-valued in some cases.

```
plt.show()
```



## Simple Examples: Getting Started with Kaggle

We're going to take a look at the data hosted by Kaggle. Try extracting feature matrix and target vector from the following datasets:

- House Prices: Advanced Regression Techniques
- Titanic: Machine Learning from Disaster
- Digit Recognizer

# Installing Kaggle Module

```
In [2]: !pip install kaggle
```

```
Requirement already satisfied: kaggle in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (1.5.3)  
Requirement already satisfied: requests in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (2.21.0)  
Requirement already satisfied: certifi in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (2018.11.29)  
Requirement already satisfied: six>=1.10 in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (1.12.0)  
Requirement already satisfied: python-dateutil in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (2.7.5)  
Requirement already satisfied: urllib3<1.25,>=1.21.1 in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (1.24.1)  
Requirement already satisfied: tqdm in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (4.28.1)  
Requirement already satisfied: python-slugify in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from kaggle) (3.0.2)  
Requirement already satisfied: idna<2.9,>=2.5 in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from requests->kaggle) (2.8)  
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from requests->kaggle) (3.0.4)  
Requirement already satisfied: text-unidecode==1.2 in /Users/kuoyaojen/anaconda3/lib/python3.7/site-packages (from python-slugify->kaggle) (1.2)
```

## Creating New API Token

- Register / Log in Kaggle
- My Account
- Create New API Token

## Using Kaggle API on Google Colab

```
!mkdir /root/.kaggle
import json
token = {"username": "YOUR-USERNAME", "key": "YOUR-KEY"}
with open('/root/.kaggle/kaggle.json', 'w') as file:
    json.dump(token, file)
!chmod 600 /root/.kaggle/kaggle.json
```

```
In [3]: !kaggle datasets list
```

ref	size	lastUpdated	downloadCount	title
ronitf/heart-disease-uci	3KB	2018-06-25 11:33:56	27773	Heart Disease UCI
karangadiya/fifa19 player dataset	2MB	2018-12-21 03:52:59	21716	FIFA 19 complete
iarunava/cell-images-for-detecting-malaria es Dataset	337MB	2018-12-05 05:40:21	4379	Malaria Cell Imag
bigquery/crypto-ethereum-classic Blockchain	70GB	2019-03-20 23:21:25	0	Ethereum Classic
jutrera/stanford-car-dataset-by-classes-folder set by classes folder	2GB	2018-07-02 07:35:45	2769	Stanford Car Data
vjchoudhary7/customer-segmentation-tutorial-in-python mentation Data	2KB	2018-08-11 07:23:02	6845	Mall Customer Seg
russellyates88/suicide-rates-overview-1985-to-2016 rview 1985 to 2016	396KB	2018-12-01 19:18:25	20324	Suicide Rates Ove
rmisra/news-headlines-dataset-for-sarcasm-detection taset For Sarcasm Detection	2MB	2018-06-09 22:14:56	2556	News Headlines Da
laval8/google-play-store-apps Apps	2MB	2019-02-03 13:55:47	48807	Google Play Store
mohansacharya/graduate-admissions ns	9KB	2018-12-28 10:07:14	18344	Graduate Admissio
jessicali9530/stanford-dogs-dataset aset	735MB	2019-02-13 05:45:25	2480	Stanford Dogs Dat
anokas/kuzushiji	318MB	2018-12-17 01:19:31	798	Kuzushiji-MNIST
cityofLA/los-angeles-parking-citations ng Citations	257MB	2019-04-13 22:17:45	3032	Los Angeles Parki
noriuk/us-education-datasets-unification-project tatasets: Unification Project	85MB	2019-03-02 18:41:52	3308	U.S. Education Da
safegraph/visit-patterns-by-census-block-group r Insights For Neighborhoods	66MB	2018-12-19 21:31:50	1303	Consumer & Visito



safegraph/census-block-group-american-community-survey-data	Census Block Grou
p American Community Survey Data	2GB 2018-12-22 00:29:56 707
jessicali9530/celeba-dataset	CelebFaces Attrib
utes (CelebA) Dataset	1GB 2018-06-01 20:08:48 6412
pavansanagapati/urban-sound-classification	Urban Sound Class
ification	6GB 2018-06-16 13:44:36 2271
fivethirtyeight/fivethirtyeight-comic-characters-dataset	FiveThirtyEight C
omic Characters Dataset	577KB 2019-03-26 15:01:15 2297
xvivancos/barcelona-data-sets	Barcelona data se
ts	1MB 2018-12-13 14:16:53 3669

```
In [4]: !kaggle datasets list -s MNIST
```

ref ated	downloadCount	title	size	lastUpd
-----	-----	-----	-----	-----
daavoo/3d-mnist		3D MNIST	154MB	2016-11
-09 18:53:12	2471			
datamunge/sign-language-mnist		Sign Language MNIST	31MB	2017-10
-20 15:09:18	5690			
kmader/skin-cancer-mnist-ham10000		Skin Cancer MNIST: HAM10000	3GB	2018-09
-20 20:36:13	7400			
zalando-research/fashionmnist		Fashion MNIST	69MB	2017-12
-07 00:54:20	28597			
kmader/colorectal-histology-mnist		Colorectal Histology MNIST	991MB	2018-09
-19 14:20:49	927			
vikramtiwari/mnist-numpy		mnist.npz	11MB	2018-06
-29 01:59:44	445			
oddrationale/mnist-in-csv		MNIST in CSV	15MB	2018-05
-19 02:24:20	4936			
pablotab/mnistpklgz		mnist.pkl.gz	15MB	2017-11
-20 15:02:57	298			
miningjerry/mnist-for-tf		mnist for tf	15MB	2017-05
-16 09:50:29	63			
joewkim/mnist-data		mnist data	11MB	2017-11
-07 19:39:08	66			
backalla/words-mnist		Words MNIST	47MB	2018-06
-06 09:34:28	114			
mustafaali96/mnist		mnist.	11MB	2018-05
-10 22:18:24	44			
crawford/emnist		EMNIST (Extended MNIST)	1GB	2017-12
-20 17:42:58	5887			
ashishguptajiit/handwritten-az		Handwritten A-Z	92MB	2018-01
-26 15:44:12	527			
kevinv/mnist1		mnist1	15MB	2017-11
-30 16:55:58	4			

jwjohnson314/notmnist		notMNIST	226MB	2018-02
-14 19:52:14	607			
anokas/kuzushiji		Kuzushiji-MNIST	318MB	2018-12
-17 01:19:31	798			
farhanhubble/multimnistm2nist		Multidigit MNIST(M2NIST)	17MB	2018-07
-16 12:05:34	161			
lianglirong/mnistnpz		mnist.npz	11MB	2018-09
-16 12:44:43	6			
zsx242030/mnistplk		mnist.plk	11MB	2018-10
-06 03:59:01	0			

# House Prices: Advanced Regression Techniques

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques>  
(<https://www.kaggle.com/c/house-prices-advanced-regression-techniques>)

```
In [5]: # Target Vector: SalePrice

train_url = "https://storage.googleapis.com/kaggle_datasets/House-Prices-Advanced-Regression-Techniques/train.csv"
```

```
In [6]: # Or using Kaggle-API
!kaggle competitions download -c house-prices-advanced-regression-techniques --for
ce
```

```
Downloading sample_submission.csv to /Users/kuoyaojen/python-sklearn-cht
0%|                                     | 0.00/31.2k [00:00<?, ?B/
s]
```

```
100%|████████████████████████████████████████████████████████████████████████████████| 31.2k/
31.2k [00:00<00:00, 639kB/s]
```

```
Downloading test.csv to /Users/kuoyaojen/python-sklearn-cht
100%|████████████████████████████████████████████████████████████████████████████████| 441
k/441k [00:00<00:00, 1.16MB/s]
```

```
Downloading train.csv to /Users/kuoyaojen/python-sklearn-cht
100%|████████████████████████████████████████████████████████████████████████████████| 450
k/450k [00:00<00:00, 2.46MB/s]
```

```
Downloading data_description.txt to /Users/kuoyaojen/python-sklearn-cht
0%|                                     | 0.00/13.1k [00:00<?, ?B/
s]
100%|████████████████████████████████████████████████████████████████████████████████| 13.1k/1
3.1k [00:00<00:00, 6.88MB/s]
```

# Titanic: Machine Learning from Disaster

<https://www.kaggle.com/c/titanic> (<https://www.kaggle.com/c/titanic>).

```
In [7]: # Target Vector: Survived
```

```
train_url = "https://storage.googleapis.com/kaggle_datasets/Titanic-Machine-Learning-from-Disaster/train.csv"
```

```
In [8]: # Or using Kaggle-API
```

```
!kaggle competitions download -c titanic --force
```

Downloading train.csv to /Users/kuoyaojen/python-sklearn-cht

[illegible]

Downloading test.csv to /Users/kuoyaojen/python-sklearn-cht

```
0% |                               | 0.00/28.0k [00:00<?, ?B/
```

```
100% |██████████████████████████████████████████████████████████████████████████| 28.0k/2  
8.0k [00:00<00:00, 10.9MB/s]
```

Downloading gender\_submission.csv to /Users/kuoyaojen/python-sklearn-cht

```
0% | _____ | 0.00/3.18k [00:00<?, ?B/
s]
```

```
100% |██████████████████████████████████████████████████████████████████████████| 3.18k/3.  
18k [00:00<00:00, 2.58MB/s]
```



# Digit Recognizer

<https://www.kaggle.com/c/digit-recognizer> (<https://www.kaggle.com/c/digit-recognizer>).

```
In [9]: # Target Vector: label
```

```
train_url = "https://storage.googleapis.com/kaggle_datasets/Digit-Recognizer/train.csv"
```

```
In [10]: # Or using Kaggle-API
```

```
!kaggle competitions download -c digit-recognizer --force
```

Downloading train.csv to /Users/kuoyaojen/python-sklearn-cht

[illegible][illegible]

Downloading test.csv to /Users/kuoyaojen/python-sklearn-cht

```
100% |██████████████████████████████████████████████████████████████████████████| 48.8M/4  
8.8M [00:13<00:00, 4.17MB/s]
```

Downloading sample submission.csv to /Users/kuoyaojen/python-sklearn-cht

```
0% | | 0.00/235k [00:00<?, ?B/
s]
```

[illegible]

# **The Scikit-learn Estimator Object**

Every algorithm is exposed in scikit-learn via an "Estimator" object.

```
In [11]: import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt
```

Estimator parameters: All the parameters of an estimator can be set when it is instantiated, and have suitable default values:

```
In [12]: from sklearn.linear_model import LinearRegression
```

```
model = LinearRegression()  
print(model)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,  
                  normalize=False)
```

Estimated Model parameters: When data is fit with an estimator, parameters are estimated from the data at hand. All the estimated parameters are attributes of the estimator object ending by an underscore:

```
In [13]: train_url = "https://storage.googleapis.com/kaggle_datasets/House-Prices-Advanced-Regression-Techniques/train.csv"
train_df = pd.read_csv(train_url)
X_train = train_df["GrLivArea"].values
y_train = train_df["SalePrice"].values
reg = LinearRegression()
reg.fit(X_train, y_train)
print(reg.intercept_)
print(reg.coef_)
```

```
[18569.02585649]
```

```
[[107.13035897]]
```

```
In [ ]: xfit = np.linspace(X_train.min() - 10, X_train.max() + 10, 100).reshape(-1, 1)
yfit = reg.predict(xfit)
plt.scatter(train_df["GrLivArea"], train_df["SalePrice"], label='train', s=3, color="#4286f4")
plt.plot(xfit, yfit, color="#f4a041", linewidth=2, label='thetas')
plt.legend()
```

```
In [14]: plt.show()
```

