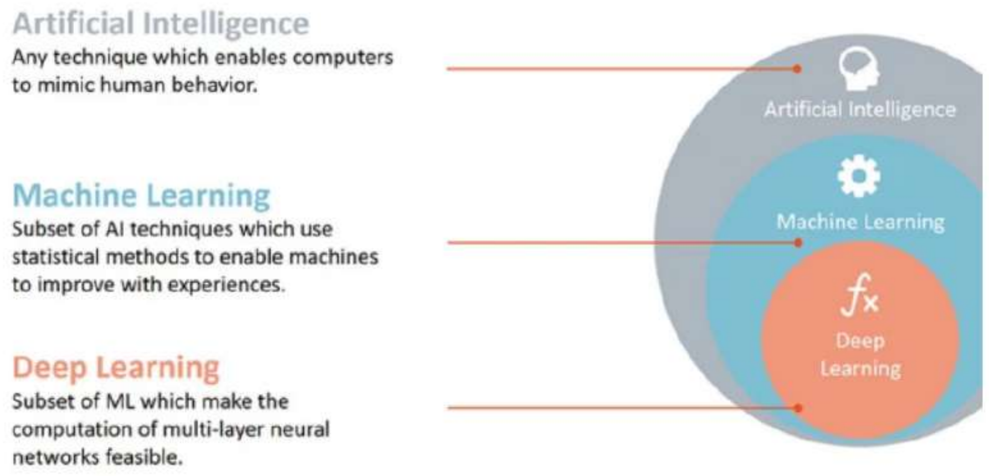
機器學習簡介

郭耀仁

需要哪些前置作業

- Gmail 帳號（為了使用 Google Colab）
- 會操作 Jupyter Notebook 或 Google Colab
- Python 程式設計
- 瞭解 numpy 與矩陣運算
- 瞭解 pandas 的 Series 與 DataFrame
- 瞭解 matplotlib.pyplot

機器學習的定位


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.
Artificial Intelligence
Machine Learning
Deep Learning

Source: [rapidminer]

定義機器學習

Arthur Samuel

> *The field of study that gives computers the ability to learn without being explicitly programmed.*

定義機器學習（2）

Tom Mitchell

> *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.*

定義機器學習（3）

林軒田

我們從小是怎麼樣辨認一棵樹的，是爸爸媽媽告訴我們一百條規則來定義嗎？其實不是的，很大一部分是透過我們自己的觀察很多的樹與不是樹之後，得到並且內化了辨認一棵樹的技巧，機器學習想要做的就是一樣的事情。

機器學習的種類

- 監督式學習（Supervised Learning）：具有目標變數的 Labeled data
  - 迴歸問題：目標變數是連續型（數值），像是股價、氣溫、匯率、房價...等
  - 分類問題：目標變數是離散型（類別），像是垃圾郵件/非垃圾郵件、熱狗/非熱狗...等
- 非監督式學習（Unsupervised Learning）：沒有目標變數的 Unlabeled data
  - 聚類問題：客戶分群
  - 降維問題：主成份分析

認識 Scikit-Learn

- 用來實作資料探勘與機器學習的 Python 套件
- 建構於 NumPy，SciPy 與 Matplotlib 套件之上
- 有六大功能模組：
    - 預處理
    - 降維
    - 迴歸
    - 分群
    - 分類
    - 模型評估

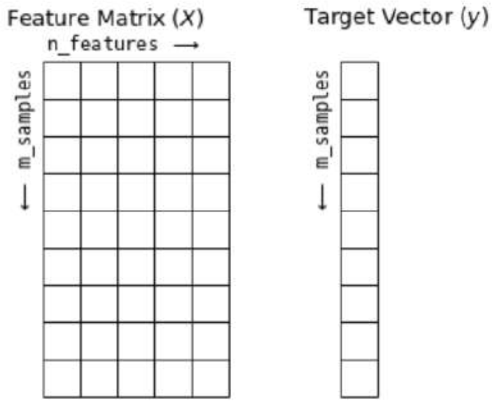
認識 Scikit-Learn（2）

- 其他的 Scikits
- Scikit-Learn 機器學習地圖

資料如何被 Scikit-Learn 解讀？

- 二維的陣列 `ndarray`
- 二維的陣列外觀為 (m_samples, n_features)
  - Target vector(y): (m, 1)
  - Feature matrix(X): (m, n-1)

In [2]:
```python
# Figure from the Python Data Science Handbook
plt.show()
```



範例資料：Getting Started with Kaggle

試著擷取出這些資料的 Target vector 與 Feature matrix

House Prices: Advanced Regression Techniques

https://www.kaggle.com/c/house-prices-advanced-regression-techniques

```
In [4]: # Target Vector: SalePrice

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

Titanic: Machine Learning from Disaster

https://www.kaggle.com/c/titanic

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

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

Digit Recognizer

https://www.kaggle.com/c/digit-recognizer

```
In [6]: # Target Vector: label

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

Scikit-Learn 的 Estimator

每個 Scikit-Learn 的演算法都是一個 Estimator 類別

- 在初始化的時候設定參數

In [9]:
```python
from sklearn.linear_model import LinearRegression

reg = LinearRegression()
print(reg)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

初始化之後將資料輸入進行 fit

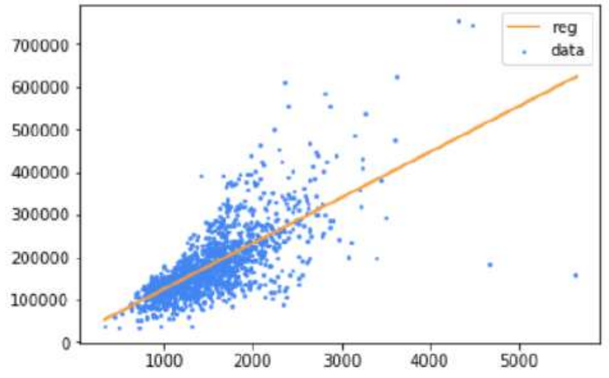
- Estimator 的屬性命名都以底線結尾

In [10]:
```python
import pandas as pd

data_url = "https://storage.googleapis.com/kaggle_datasets/House-Prices-Advanced-Regression-Techniq
ues/train.csv"
labeled = pd.read_csv(data_url)
X = labeled["GrLivArea"].values.reshape(-1, 1)
y = labeled["SalePrice"].values.reshape(-1, 1)
reg.fit(X, y)
print(reg.intercept_)
print(reg.coef_)
```

```
[ 18569.02585649]
[[ 107.13035897]]
```

```
In [12]:  import numpy as np
          import matplotlib.pyplot as plt

          xfit = np.linspace(X.min() - 10, X.max() + 10, 100).reshape(-1, 1)
          yfit = reg.predict(xfit)
          plt.scatter(X, y, label='data', s=3, color="#4286f4")
          plt.plot(xfit, yfit, color="#f4a041", linewidth=2, label='reg')
          plt.legend()

Out[12]:  <matplotlib.legend.Legend at 0x1a1eeed240>
```

In [13]:
```python
plt.show()
```



非演算法也用 Estimator 類別去操作

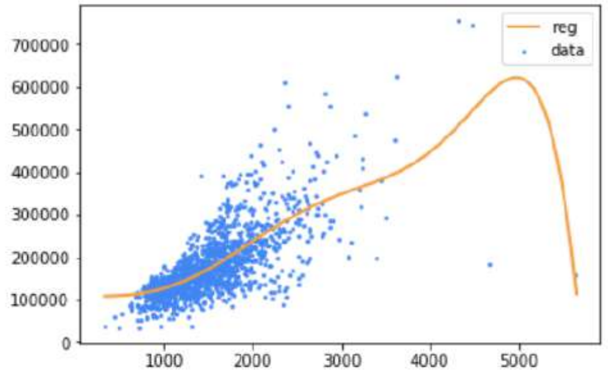
- 同樣在初始化的時候設定參數
- 如果是預處理，在初始化後將資料輸入做 `fit_transform`

```
In [24]: from sklearn.preprocessing import PolynomialFeatures

         poly = PolynomialFeatures(degree=7)
         X_poly = poly.fit_transform(X)
         reg.fit(X_poly, y)
         print(reg.intercept_)
         print(reg.coef_)
```

```
[ 108657.84974682]
[[  0.00000000e+00   2.44911864e-12   3.01889423e-14   4.35404746e-11
    3.78254367e-08  -2.37317287e-11   5.19276846e-15  -3.85321774e-19]]
```

```
In [25]: xfit = np.linspace(X.min() - 10, X.max() + 10, 100).reshape(-1, 1)
         xfit_poly = poly.fit_transform(xfit)
         yfit = reg.predict(xfit_poly)
         plt.scatter(X, y, label='data', s=3, color="#4286f4")
         plt.plot(xfit, yfit, color="#f4a041", linewidth=2, label='reg')
         plt.legend()

Out[25]: <matplotlib.legend.Legend at 0x1a1ff86940>
```

In [26]:
```
plt.show()
```



使用 NumPy 做矩陣運算

二維的 ndarray

- 向量：.reshape(-1, 1)
- 矩陣：.reshape(m, n)
- 轉置：.T屬性或np.transpose()
- 相乘：.dot()或np.dot()
- 反矩陣：np.linalg.inv()

運算練習

- 練習計算 $\boldsymbol{u}^T\boldsymbol{v}$

$$\boldsymbol{u} = \begin{bmatrix} 4 \\ -4 \\ -3 \end{bmatrix}$$

$$\boldsymbol{v} = \begin{bmatrix} 4 \\ 2 \\ 4 \end{bmatrix}$$

運算練習

- 練習計算 AB 與 BA

$$A = \begin{bmatrix} 1 & 2 \\ 4 & 5 \end{bmatrix}$$

$$B = \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix}$$