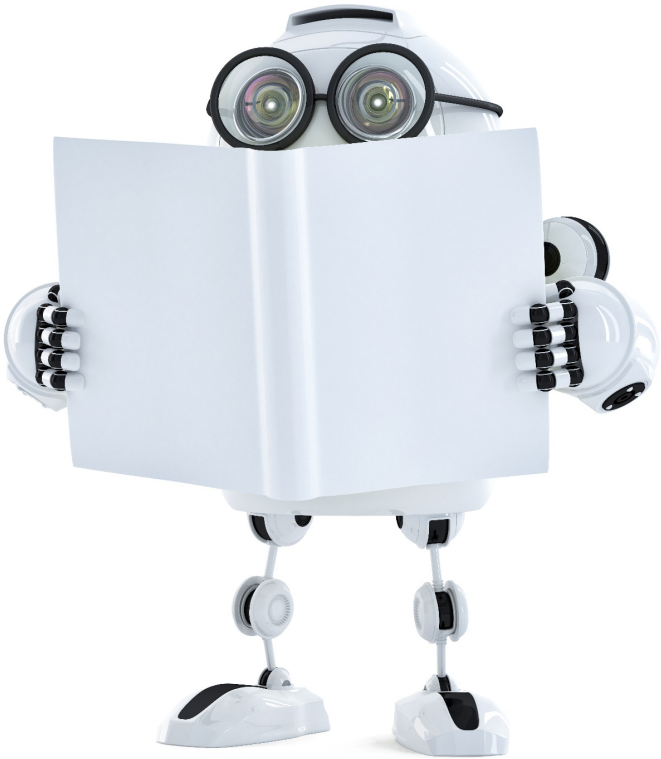
Representing a model

An understanding of data

Director of TEAMLAB
Sungchul Choi



데이터는 불렀다
그 다음은?

$$y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$$
$$+\beta_6 x_6 + \beta_7 x_7 + \cdots \beta_{13} x_{13} + \beta_0 \cdot 1$$

이 식을 표현하고 싶다

컴퓨터로 저 식을 다 표현할 수 있을까?

https://goo.gl/3PoYSx https://goo.gl/1DgF83 https://goo.gl/6uDPX4

있다
sympy

그러나 더 쉽게
표현하고 싶다.

비밀은 Vector+Matrix

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_{13} x_{13} + w_0 x_0$$

$$= \sum_{j=0}^{13} w_i x_i = \mathbf{w}^{\mathbf{T}} \mathbf{x}$$

벡터를 Array로 표현하기

$\mathbf{w}^T\mathbf{x}$

여러개의 Table.

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_{13} \end{bmatrix}$$

$$\boldsymbol{x}^{(1)} = \begin{bmatrix} 1 \\ 0.00632 \\ 18 \\ 2.31 \\ 0.538 \\ \vdots \\ 24 \end{bmatrix}$$

numpy
벡터연산 최적 라이브러리

Numpy

- Numerical Python
- 파이썬의 고성능 과학 계산용 기초 패키지
- Matrix와 Vector와 같은 Array 연산의 사실상의 표준
- 한글로 넘파이로 주로 통칭, 넘피/늄파이라고 부르기도 함
- 일반 List에 비해 빠르고, 메모리 효율적
- 반복문 없이 데이터 배열에 대한 처리를 지원함
- 선행대수와 관련된 다양한 기능을 제공함
- C, C++, 포트란등의 언어와 통합 가능

From 파이썬 라이브러리를 활용한 데이터 분석

Numpy 설치

```
activate ml_scratch # 가상환경실행
conda install numpy #  numpy 설치
jupyter notebook # 주피터 실행하기
```

Numpy 계산해보기

$$\mathbf{w}^T\mathbf{x} \qquad \boldsymbol{w} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \boldsymbol{x}^{(1)} = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

```
In [1]: import numpy as np # Numpy 라이브러리 호출

In [2]: weight_vector = np.array([[1], [1], [1]]) # Weight Vector
        x_vector = np.array([[3], [4], [5]])

In [3]: weight_vector.T.dot(x_vector) # 1 * 3 + 1 * 4 + 1 * 5 = 12
Out[3]: array([[12]])
```

그럼 이건???

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	CAT. MEDV
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24	0
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6	0
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7	1
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4	1
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2	1
0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7	0
0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9	0
0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1	0
0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311	15.2	386.63	29.93	16.5	0
0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9	0
0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15	0
0.11747	12.5	7.87	0	0.524	6.009	82.9	6.2267	5	311	15.2	396.9	13.27	18.9	0
0.09378	12.5	7.87	0	0.524	5.889	39	5.4509	5	311	15.2	390.5	15.71	21.7	0
0.62976	0	8.14	0	0.538	5.949	61.8	4.7075	4	307	21	396.9	8.26	20.4	0
0.63796	0	8.14	0	0.538	6.096	84.5	4.4619	4	307	21	380.02	10.26	18.2	0

Pandas + Numpy

$$\text{for each } i \sum_{j=0}^{13} w_j x_j^{(i)} = X \cdot \mathbf{w}$$

$x_0 = 1$

Data table

데이터 Sample

Weight number

Pands 데이터 로딩

In [4]:
```
import pandas as pd
```

In [5]:
```
data_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data' #Data URL
df_data = pd.read_csv(data_url, sep='\s+', header = None) #csv 타입 데이터 로드, separate는 빈공간으로 지정하고, Column은 없음
df_data.columns = ['CRIM','ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX', 'PTRATIO' ,'B', 'LSTAT', 'MEDV']
df_data.head()
```

Out[5]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2

In [6]:
```
df_data['weight_0'] = 1
df_data= df_data.drop("MEDV", axis=1)
df_data.head()
```

X_0 추가, Y 값 없애기

In [6]:
```
df_data['weight_0'] = 1 # weight 0 값 추가
df_data= df_data.drop("MEDV", axis=1) #Y값 제거
df_data.head()
```

Out[6]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	weight_0
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	1
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	1
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	1
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	1
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	1

Dataframe → numpy, 곱셈연산

```
In [9]: df_matrix = df_data.as_matrix() # Matrix Data로 변환하기
        weight_vector =  np.random.random_sample((14, 1))

In [10]: df_matrix.dot(weight_vector)

Out[10]: array([[ 433.52000019],
                [ 394.83907923],
                [ 390.44525364],
                [ 377.71929915],
                [ 380.42922667],
                [ 378.55996125],
                [ 442.05358228],
                [ 446.92169217],
                [ 442.16560852],
                [ 439.29722575],
                [ 444.48909317],
                [ 444.65609699],
                [ 436.55549191],
                [ 429.5979006 ],
                [ 421.84837601],
                [ 428.20934742],
                [ 420.27744009],
                [ 426.35315551],
                [ 360.92936403],
                [ 426.96198486],
```

$$\text{for each } i \sum_{j=0}^{13} w_j x_j^{(i)} = X \cdot \mathbf{w}$$

Human knowledge belongs to the world.