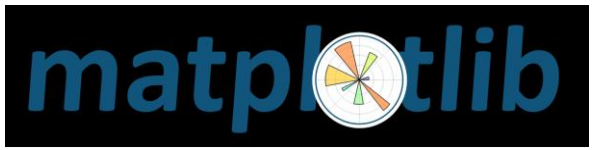
Machine Learning in Security

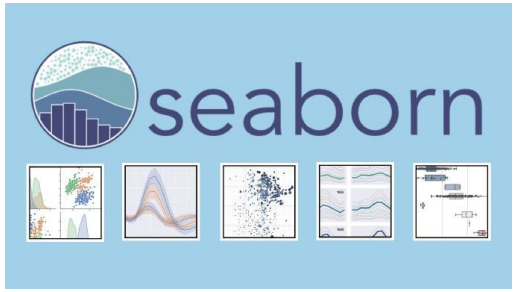
Linear Regression

파이썬 라이브러리 설치

- Jupyter Notebook을 사용하여 본 머신러닝 강의에 필요한 Python 라이브러리를 설치
- **라이브러리 설치는 “activate (가상머신명)”으로 활성화된 상태에서 진행**

```
pip install pandas==1.2.3
pip install numpy==1.20.1
pip install matplotlib==3.3.4
pip install scikit-learn==0.24.1
pip install ipython== 7.21.0
pip install pydotplus==2.0.2
pip install seaborn==0.11.1
```





파이썬 라이브러리 설명(Description)

pandas

- 관계형 또는 분류된 데이터로 쉽고 직관적으로 작업할 수 있도록 설계되었고, 주로 Excel, SQL과 같이 테이블 형식, 시계열 등과 같은 데이터에 적합한 데이터 분석 패키지

numpy

- 행렬이나 일반적으로 대규모 다차원 배열을 쉽게 처리 하는데 적합하며, 데이터 구조 외에도 수치 계산을 위해 사용하는 패키지

matplotlib

- 파이썬에서 수치 해석 및 프로그래밍 환경을 제공하는 공학용 소프트웨어인 매트랩과 유사한 그래프 표시를 가능케 하여 데이터 시각화를 하는데 적합한 패키지

seaborn

- matplotlib 기반으로 구축됐으며, 통계 그래프를 그리는데 적합한 데이터 시각화 라이브러리

파이썬 라이브러리 설명(Description)

scikit-learn

- 파이썬의 기계 학습을 위한 라이브러리, 파이썬의 인터페이스에서 분류, 회귀, 클러스터링 및 차원 감소를 포함한 기계 학습 및 통계 모델링을 지원하는 도구 제공

ipython

- Jupyter 노트북같은 대화 형 프런트 엔드에서 파이썬 코드를 사용하도록 설계되었고, Jupyter 노트북에 Image함수를 사용하여 이미지를 삽입하는데 사용

pydotplus

- pydot은 오픈 소스 그래프 시각화 소프트웨어인 Graphviz의 인터페이스며, 그래프 설명 언어인 DOT 언어로 구문 분석하고 덤프하는데 적합한 패키지
- pydotplus는 pydot을 개선한 버전

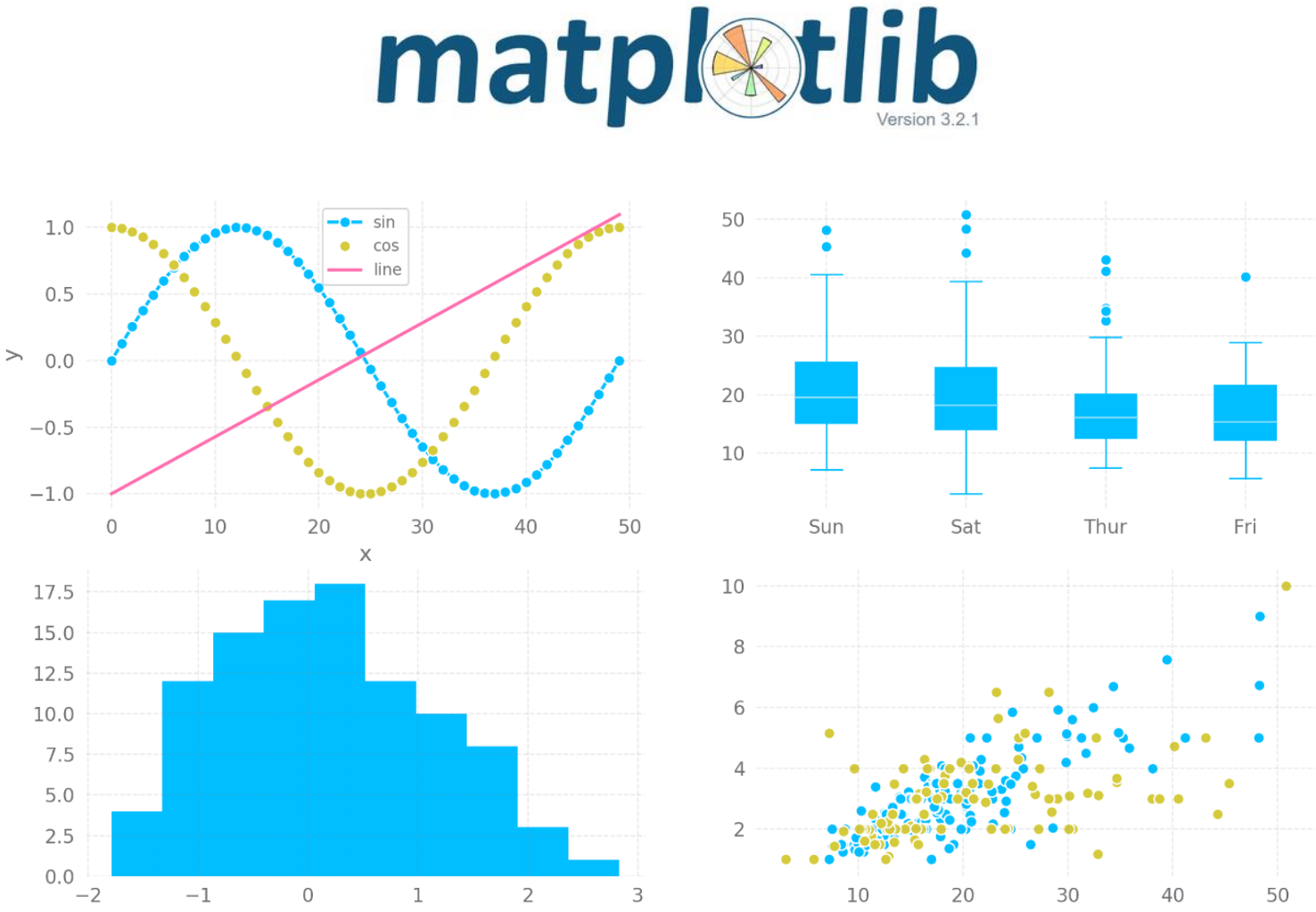
0. 주요 파이썬 라이브러리 실습

- **matplotlib:**
  - 데이터 시각화 및 2D 그래프 플롯

- **pyplot (모듈):**

2D로 보여지는 그래프의 시각화를 지원하는 다양한 함수들로 구성
  - 그래프 영역
  - 선, 레이블 표현

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html

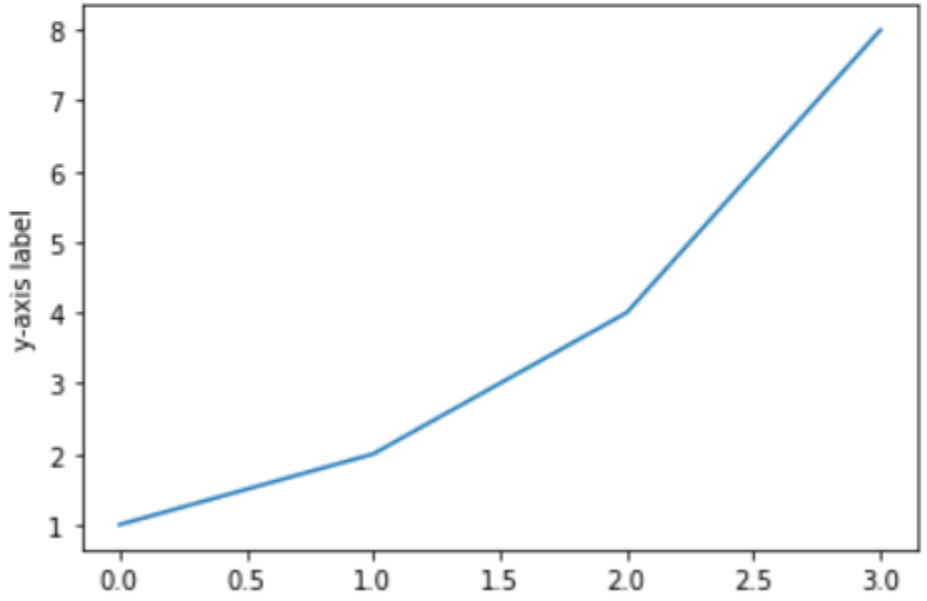


0. 주요 파이썬 라이브러리 실습

- **plot ():**
- 인자값: 임의의 개수 (예: 숫자 리스트, 색상 등)
- 색상/모양: blue / - (기본값) (예: 'go'는 green 색상에 원형(o) 마커를 사용)

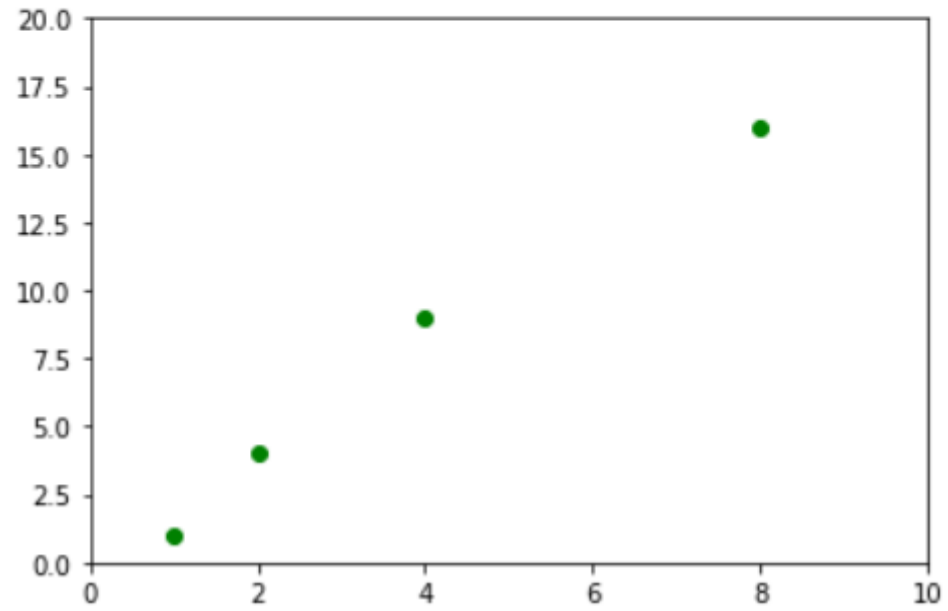
```
import matplotlib.pyplot as plt

plt.plot([1, 2, 4, 8])
plt.ylabel('y-axis label')
plt.show()
```




```
import matplotlib.pyplot as plt

plt.plot([1, 2, 4, 8], [1, 4, 9, 16], 'go')
plt.axis([0, 10, 0, 20])
plt.show()
```

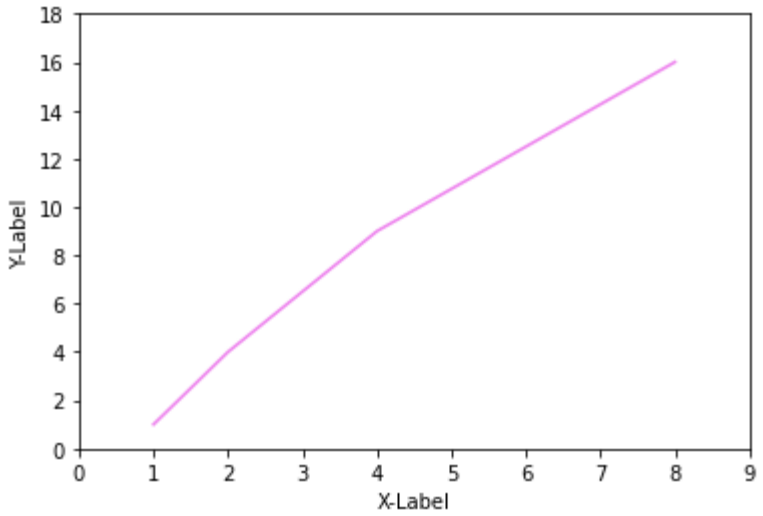


0. 주요 파이썬 라이브러리 실습

- **축의 범위 지정: axis()** 함수
  [xmin, xmax, ymin, ymax] 4개 값 모두 필요
- **색상**
  color 키워드 인자값 사용 (우측 그림 참조)

```
import matplotlib.pyplot as plt

plt.plot([1, 2, 4, 8], [1, 4, 9, 16], color='violet')
plt.xlabel('X-Label')
plt.ylabel('Y-Label')
plt.axis([0, 9, 0, 18])
plt.show()
```

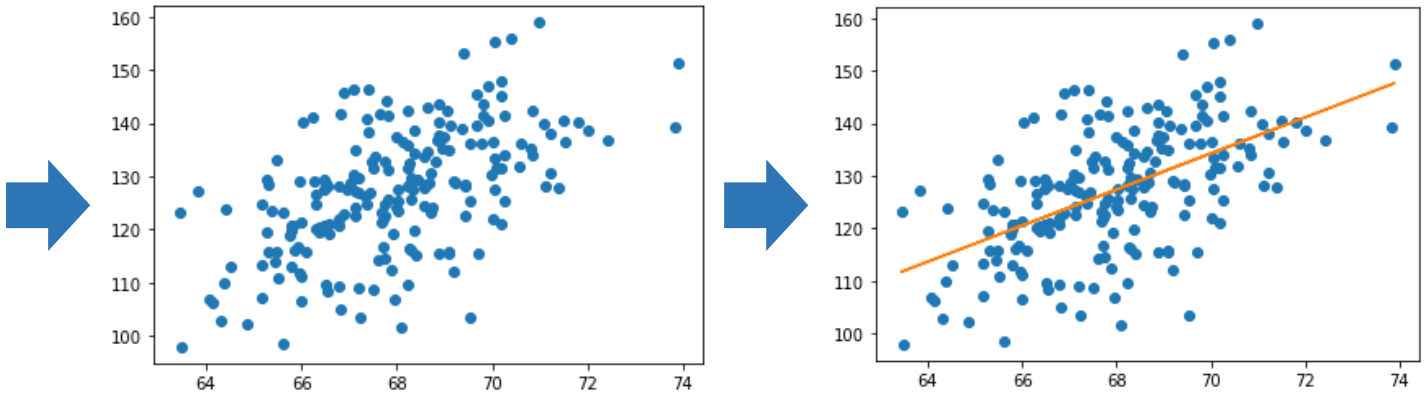


1. 학습 알고리즘(Linear Regression)

- **선형 회귀**는 종속 변수 y와 1개 이상의 독립 변수 x와의 선형 상관 관계를 모델링하는 기법이며, 데이터의 분포가 하나의 선 안에 표현될 수 있을 때 최적의 모델을 찾는 것을 선형 모델이라 한다.
- 우리는 머신 러닝을 통해 실제 데이터를 바탕으로 모델을 생성하고, **새로운 입력 값(input)에 따른 결과(output)를 예측**하는 것이 목적이다. 이때 우리가 찾아낼 수 있는 가장 직관적이고 간단한 모델은 **선(line)**이다. 이처럼 데이터를 놓고 그걸 가장 잘 설명할 수 있는 선을 찾는 분석하는 방법을 **선형 회귀(Linear Regression)**라 한다.

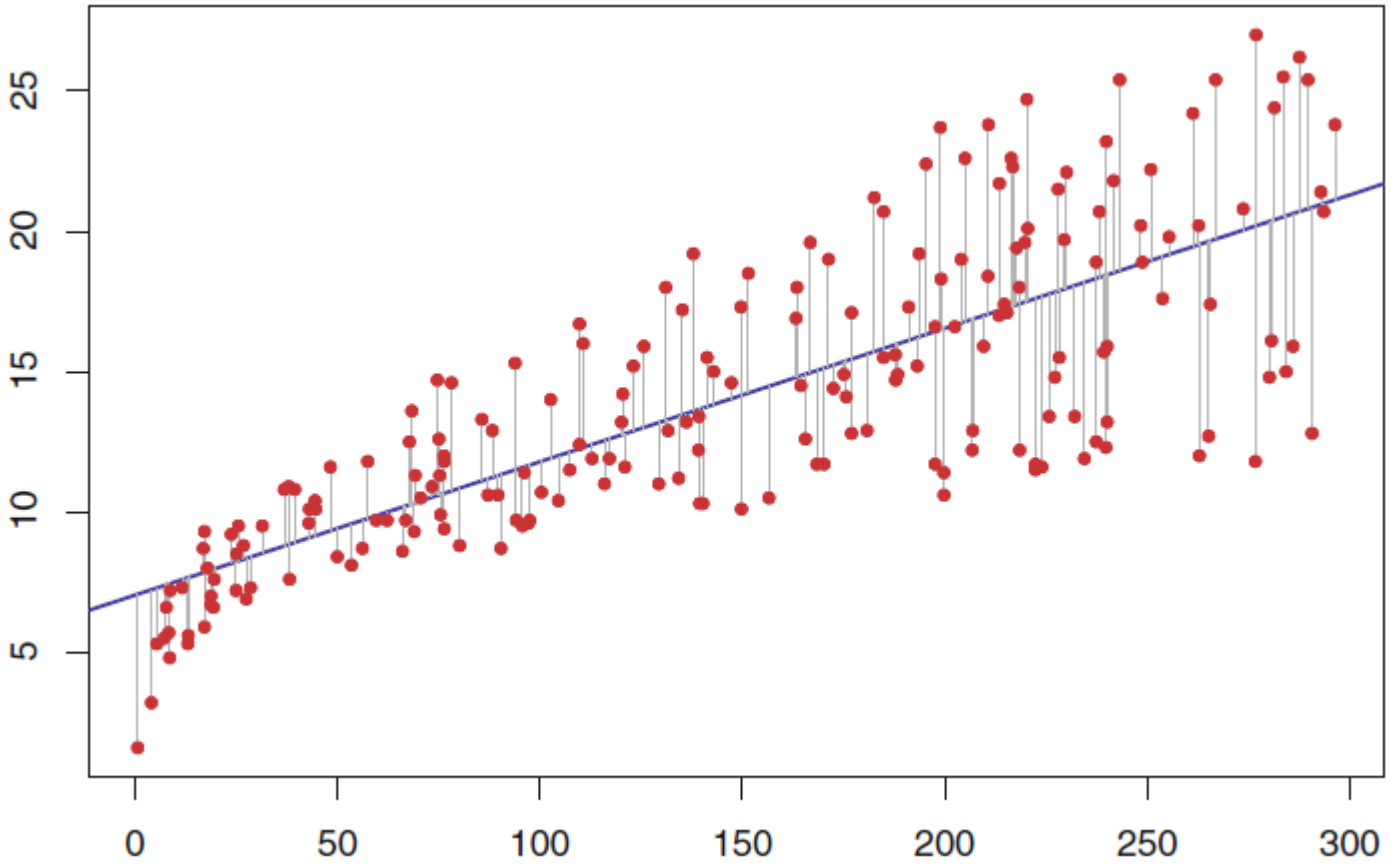
- 예를 들어, 키와 몸무게 데이터를 펼쳐 놓고 가장 잘 설명할 수 있는 선을 하나 잘 그리게 되면, 특정 사람의 키 데이터를 바탕으로 몸무게를 예측할 수 있다.

	Index	Height(inches)	Weight(pounds)
93	94	69.20	112.14
57	58	66.54	128.07
27	28	67.49	131.55



1. 학습 알고리즘(Linear Regression)

- residual: 각 점들과 그래프 간의 차이
- f(x)를 선형인 1차 함수라고 가정할 때, x, y 값은 주어지고, residual(잔차)의 최소값은 기울기 a와 절편 b에 의해 결정됨



$$\sum_{i=1}^{n} residual^2$$

$$\sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$$

$$f(x_i, \beta) = ax_i + b$$

2개의 점 A(1, 4)와 B(2, 3)에 대해
식: y = 2x + 1

A의 잔차: 4 – 3 = 1
B의 잔차: 3 – 5 = -2
Least Squared Method = 5

1. 학습 알고리즘(Linear Regression)

- 알고리즘

$Y = ax + b$

1. a와 b값을 랜덤으로 생성 (즉, 임의의 직선을 하나 그림)
2. x에 대한 실제 y값과 (1)에서 그린 직선에 x값을 대입한 결과 값을 비교 (residual)
   만약, 두 값의 차이가 크면 오류가 큰 것임
3. a와 b값을 조정 (오류 값이 감소하는 방향으로 직선의 기울기와 위치를 조정)
4. (2)단계로 돌아가 다시 오류를 계산하여 이전보다 오류 값이 감소 했는지를 확인
   - 오류 값이 줄었다면 (3)단계로 돌아가 a와 b값을 재조정
   - 오류 값이 늘었으면 학습 종료

1. 학습 알고리즘(Linear Regression) – 몸무게 예측

In [1]:
```
from sklearn.linear_model import LinearRegression
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
df = pd.read_csv("hw_200.csv")
df.sample(3)
```

Out [3]:

	Index	Height(inches)	Weight(pounds)
93	94	69.20	112.14
57	58	66.54	128.07
27	28	67.49	131.55

 # 데이터 정제를 위해 필요 없는 값인 Index 칼럼을 삭제

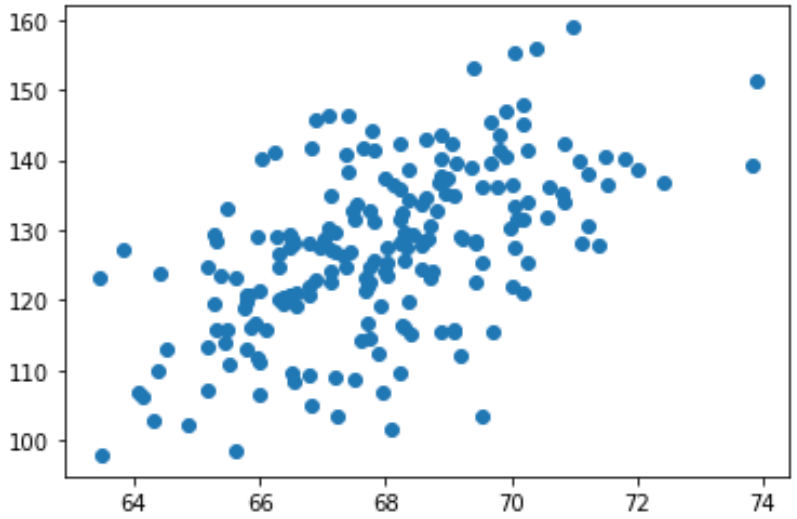
In [6]:
```
df = df.drop(["Index"], axis = 1)
df.sample(3)
```

Out [6]:

	Height(inches)	Weight(pounds)
157	67.50	108.79
108	69.44	128.42
101	67.09	130.35

1. 학습 알고리즘(Linear Regression) – 몸무게 예측

In [8]:
```
X = df["Height(inches)"]
y = df["Weight(pounds)"]
plt.plot(X, y, 'o')
plt.show()
```




```
X
```

```
0      65.78
1      71.52
2      69.40
3      68.22
4      67.79
       ...
195    65.80
196    66.11
197    68.24
198    68.02
199    71.39
Name: Height(inches), Length: 200, dtype: float64
```

In [9]:
```
line_fitter = LinearRegression()
line_fitter.fit(X.values.reshape(-1,1), y)
```

Out[9]:
```
LinearRegression()
```

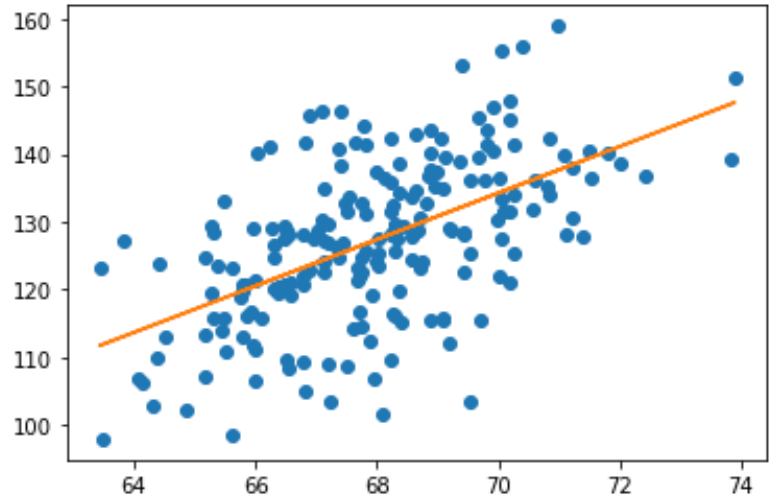
In [11]:
```
line_fitter.predict([[70]])
```

Out[11]:
```
array([134.2596226])
```

1. 학습 알고리즘(Linear Regression) – 몸무게 예측

- hw_200.csv 데이터를 바탕으로 키/몸무게에 대한 "선(line)"이 그려진 것을 확인할 수 있음

In [12]:
```
plt.plot(X, y, 'o')
plt.plot(X,line_fitter.predict(X.values.reshape(-1,1)))
plt.show()
```



응용 과제

- 위의 예제를 바탕으로 현재 가지고 있는 hw_200.csv 파일을 우리가 흔히 사용하는 cm와 kg으로 변환하고 Linear Regression 알고리즘을 적용하라.

1. 학습 알고리즘(Linear Regression) – 몸무게 예측

소스코드(본문)

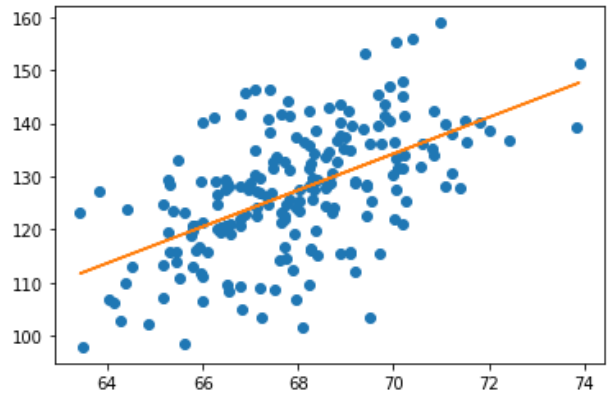
```python
from sklearn.linear_model import LinearRegression
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# 데이터 전처리
df = pd.read_csv("./linear regression/hw_200.csv")
df = df.drop(["Index"], axis = 1) # Index 삭제

# 학습을 위한 라벨링
X = df["Height(inches)"]
y = df["Weight(pounds)"]

# 학습
line_fitter = LinearRegression()
line_fitter.fit(X.values.reshape(-1,1), y)

# 예측 및 시각화
plt.plot(X, y, 'o')
plt.plot(X, line_fitter.predict(X.values.reshape(-1,1)))
plt.show()
print("predict = %.2f (pounds)" % line_fitter.predict([[70]])[0]) #테스트
```




```
predict = 134.26 (pounds)
```

소스코드(응용)

1. 학습 알고리즘(Linear Regression) – 연봉 예측

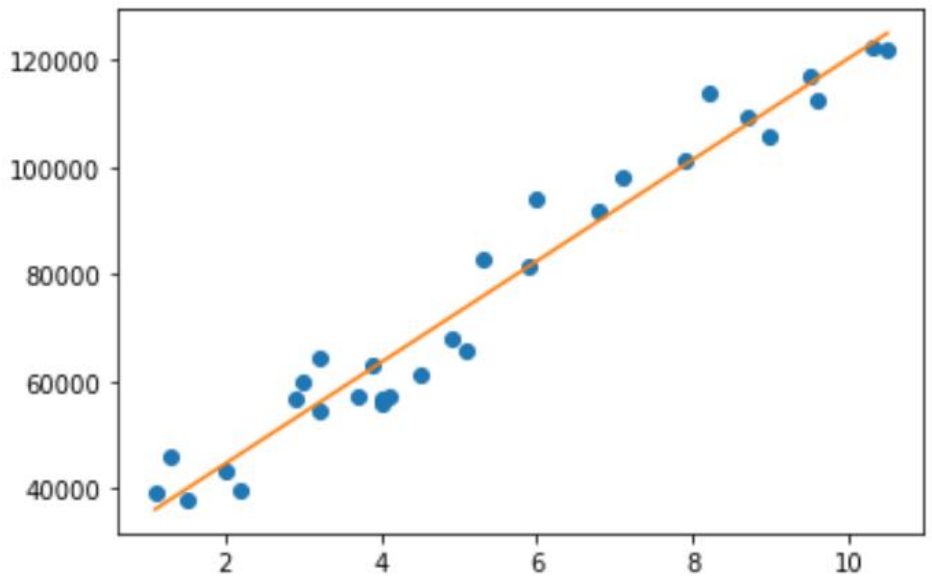
응용 과제

제공되는 salary_data.csv를 Linear Regression을 사용하여 분류하라

```
1 from sklearn.linear_model import LinearRegression
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
```

```
1 df = pd.read_csv("salary_data.csv")
2 df.sample(3)
```

	YearsExperience	Salary
19	6.0	93940
15	4.9	67938
13	4.1	57081



1. 학습 알고리즘(Linear Regression) – GPA 예측

응용 과제

제공되는 SAT.csv를 Linear Regression을 사용하여 분류하라

```
1 from sklearn.linear_model import LinearRegression
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
```

```
1 df = pd.read_csv("SAT.csv")
2 df.sample(3)
```

	SAT	GPA
46	1824	3.40
26	1787	3.28
63	2041	3.51