

dataset의 차원을 줄이는 방법

# Introduction

DIMENSIONALITY REDUCTION IN PYTHON



Jeroen Boeye

Machine Learning Engineer, Faktion

# Tidy data

차원을 언급할때  
dataset에 있는 열의 수를 의미함

	Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Ivysaur	Grass	60	62	63	60	1
Venusaur	Venusaur	Grass	80	82	83	80	1
Charmander	Charmander	Fire	39	52	43	65	1
Charmeleon	Charmeleon	Fire	58	64	58	80	1

# Tidy data

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

모든 열이 변수와 유사한지 확인

# Tidy data

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

각 행에는 각 변수에 대한 관측치가 있음

# Tidy data

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

dataset의 각 셀에 값이 생김

# The shape attribute

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

```
pokemon_df.shape
```

```
(5, 7)
```

(dataset에 10개 이상의 열이 있는 경우 data는 고차원으로 간주하는데 고차원에서 오는 복잡성 때문에 가장 중요한 패턴을 찾기가 어려운 수 있음.  
=> 이를 극복하기 위해 차원 축소 기법을 사용하여 열 수를 줄일 수 있음)

# When to use dimensionality reduction?

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

# When to use dimensionality reduction?

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

generation은 분산이 없기 때문에 유용하지 X  
-> 단순히 그 열을 drop함으로써 차원을 줄일 수 있음

# The describe method

`pokemon_df.describe()`

열에 대한 요약 통계 제공

	HP	Attack	Defense	Speed	Generation
<b>count</b>	5.0	5.0	5.0	5.0	5.0
<b>mean</b>	56.4	61.8	59.2	66.0	1.0
<b>std</b>	15.9	13.0	15.4	14.7	0.0
<b>min</b>	39.0	49.0	43.0	45.0	1.0
<b>25%</b>	45.0	52.0	49.0	60.0	1.0
<b>50%</b>	58.0	62.0	58.0	65.0	1.0
<b>75%</b>	60.0	64.0	63.0	80.0	1.0
<b>max</b>	80.0	82.0	83.0	80.0	1.0

# The describe method

```
pokemon_df.describe()
```

	HP	Attack	Defense	Speed	Generation
<b>count</b>	5.0	5.0	5.0	5.0	5.0
<b>mean</b>	56.4	61.8	59.2	66.0	1.0
<b>std</b>	15.9	13.0	15.4	14.7	0.0
<b>min</b>	39.0	49.0	43.0	45.0	1.0
<b>25%</b>	45.0	52.0	49.0	60.0	1.0
<b>50%</b>	58.0	62.0	58.0	65.0	1.0
<b>75%</b>	60.0	64.0	63.0	80.0	1.0
<b>max</b>	80.0	82.0	83.0	80.0	1.0

열의 표준 편차가 0이고 최대값과 최소값이 같다는 것을 알 수 있음

# The describe method

```
pokemon_df.describe(exclude='number')
```

	Name	Type
<b>count</b>	5	5
<b>unique</b>	5	2
	<b>top</b>	Charmander
	<b>freq</b>	3

describe()는 'exclude'인수에 'number'를 전달하여 반대로 수행  
그런 다음 숫자가 아닌 데이터에 적용된 요약 통계를 얻음

# The describe method

```
pokemon_df.describe(exclude='number')
```

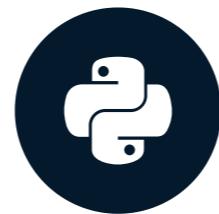
	Name	Type	
<b>count</b>	5	5	
<b>unique</b>	5	2	5개 중 2개의 고유한 유형의 포켓몬만 있음
<b>top</b>	Charmander	Grass	
<b>freq</b>	1	3	가장 많이 발생하는 유형은 유형 열에 3번 있기 때문에 잔디임

# **Let's practice!**

**DIMENSIONALITY REDUCTION IN PYTHON**

# Feature selection vs feature extraction

DIMENSIONALITY REDUCTION IN PYTHON



Jeroen Boeye

Machine Learning Engineer, Faktion

# Why reduce dimensionality?

## Your dataset will:

- be less complex
- require less disk space
- require less computation time
- have lower chance of model overfitting model은 차원이 더 적은 dataset에 과적합할 가능성이 적음

# Feature selection

income	age	favorite color
10000	18	Black
50000	47	Blue
20000	40	Blue
30000	29	Green
20000	22	Purple

차원을 줄이는 가장 간단한 방법은 더 큰 데이터 집합에서 사용자에게 중요한 feature 또는 columns만 선택하는 것

여기서 어려운 점은 어떤 feature가 중요한지 결정하는 것.

# Feature selection

ex) 대출 불이행 여부를 예측하고 싶다면,  
한 사람이 가장 좋아하는 색깔과 관련없기 때문에 dataframe에서 drop() 메서드를 사용하여 제거함

The diagram illustrates the process of feature selection. On the left, a DataFrame is shown with columns: income, age, and favorite color. The 'income' and 'age' columns are highlighted with a green border, while the 'favorite color' column is highlighted with a red border. An arrow points from this original DataFrame to a modified one on the right. The modified DataFrame only contains the 'income' and 'age' columns, with the same data as the original, demonstrating that the 'favorite color' column has been removed.

income	age	favorite color
10000	18	Black
50000	47	Blue
20000	40	Blue
30000	29	Green
20000	22	Purple

→

income	age
10000	18
50000	47
20000	40
30000	29
20000	22

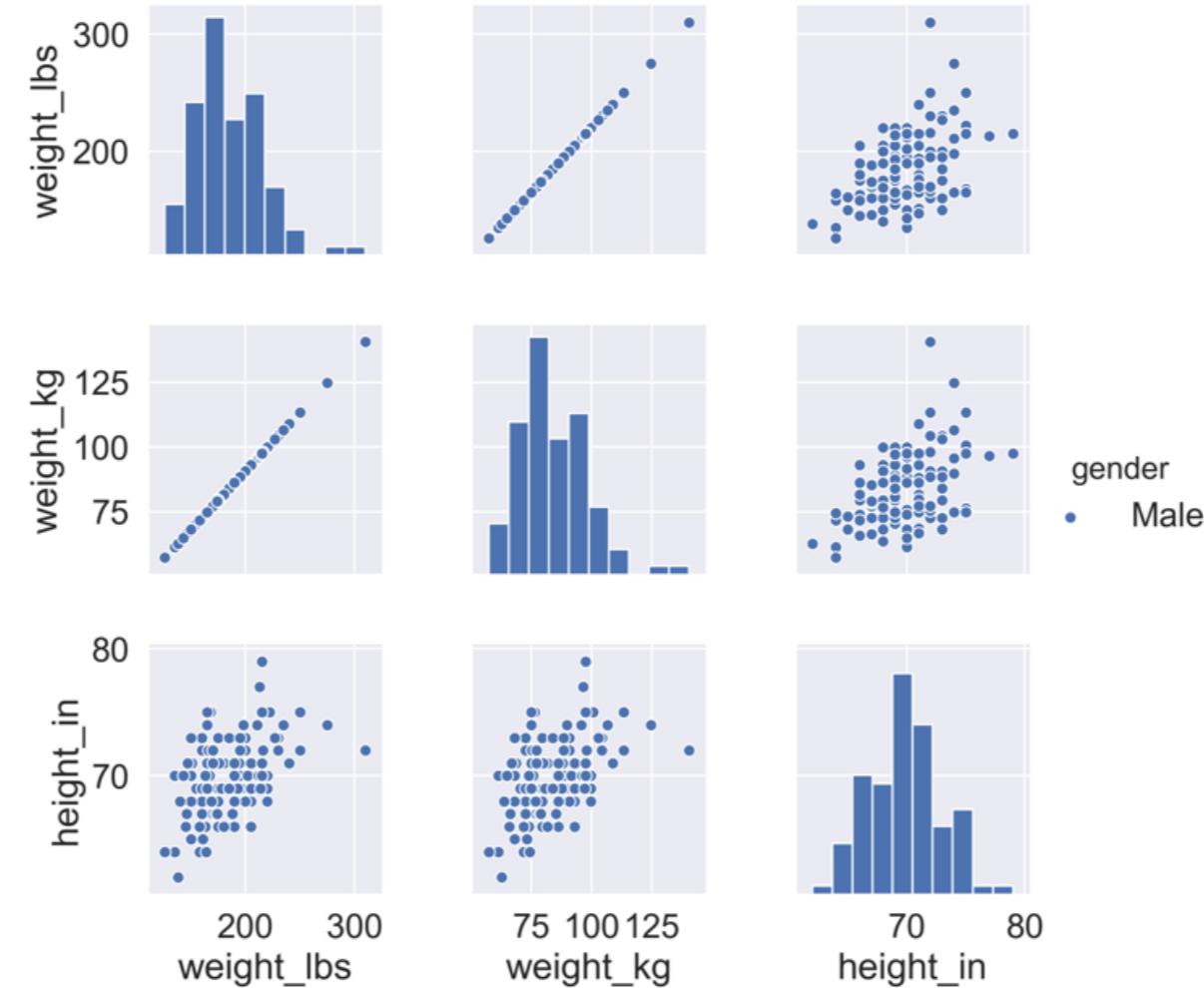
```
insurance_df.drop('favorite color', axis=1)
```

행 대신 열을 삭제하도록 지정

# Building a pairplot on ANSUR data

```
sns.pairplot(ansur_df, hue="gender", diag_kind='hist')
```

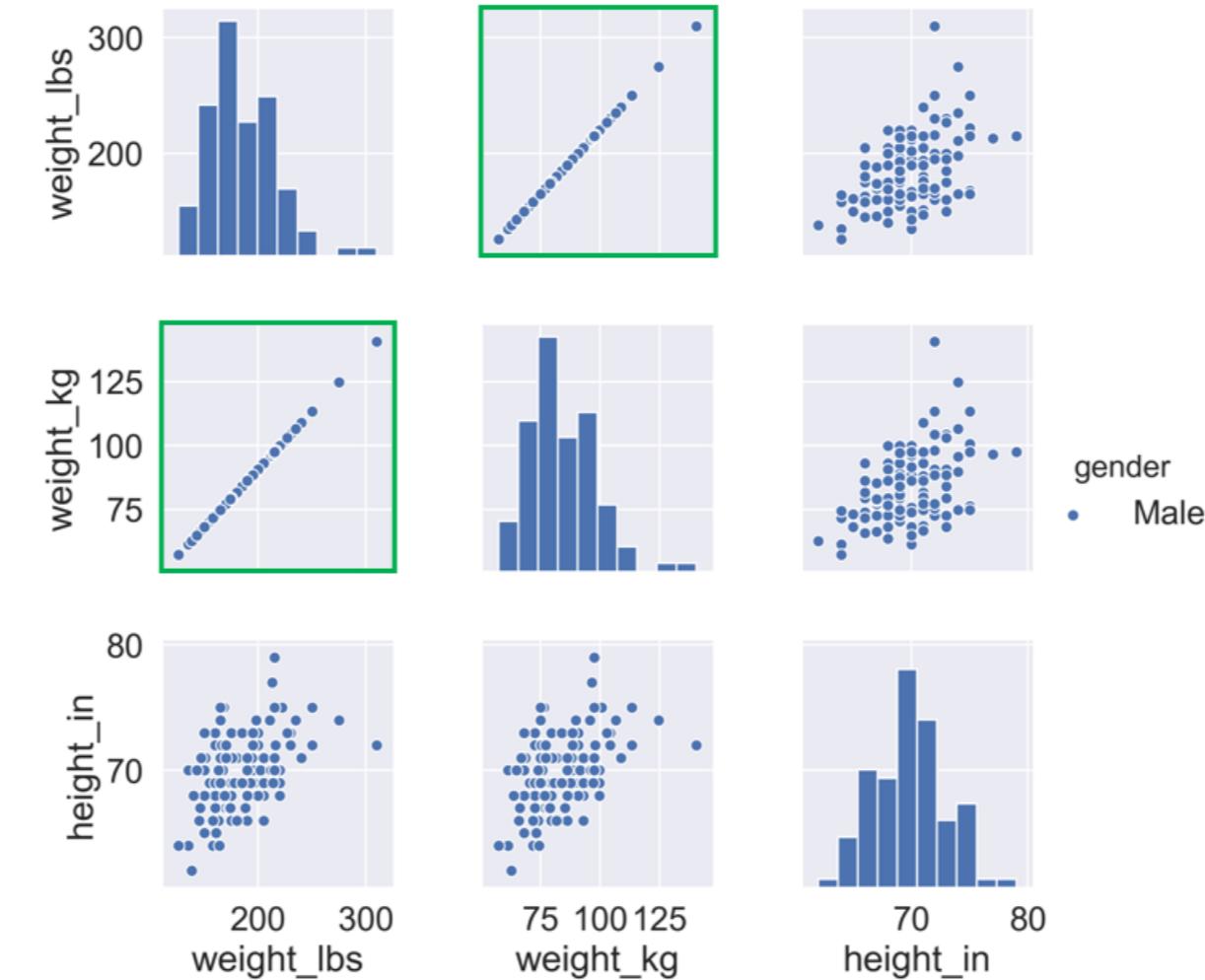
dataset의 각 숫자 feature를 대각선으로 산점도 및 각 feature의 분포의 뷰 형태로 하나씩 비교



# Building a pairplot on ANSUR data

```
sns.pairplot(ansur_df, hue="gender", diag_kind='hist')
```

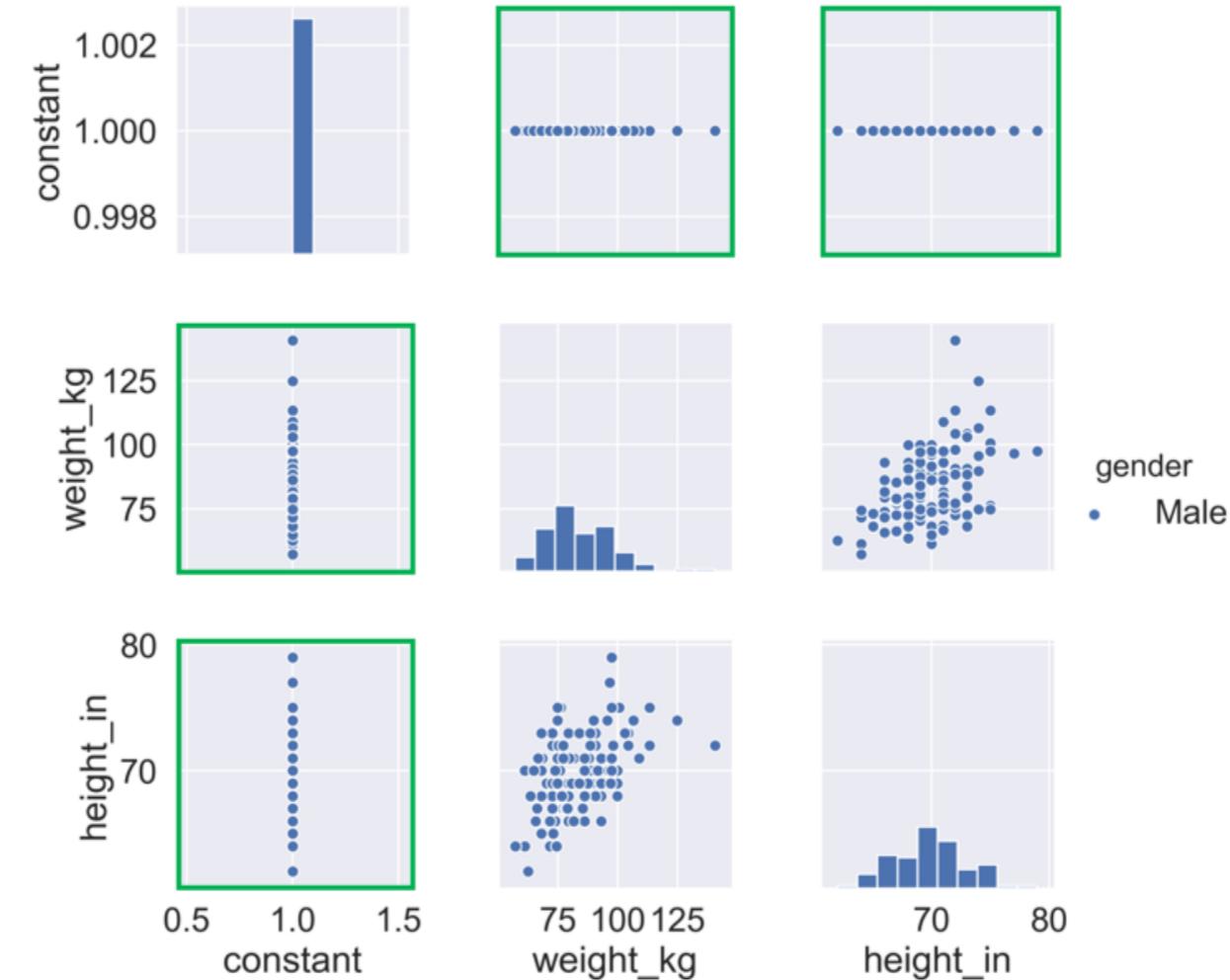
모든 점들이 대각선 상에 있기 때문에 파운드 단위의 무게가 킬로그램 단위의 무게와 완벽하게 상관되어 있다는 것을 발견할 수 있음  
→ 두 feature가 모두 동일한 정보를 가지고 있으므로 둘 중 하나를 삭제하는 것이 좋음



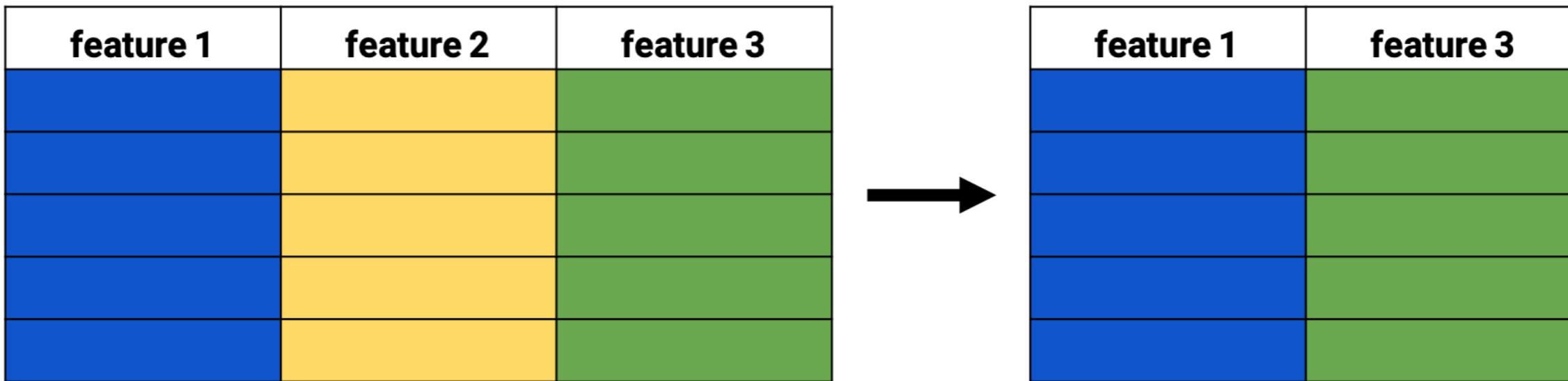
# Building a pairplot on ANSUR data

```
sns.pairplot(ansur_df, hue="gender", diag_kind='hist')
```

추가된 상수와 같은 dataset에 분산이 없는 수치적 feaurer가 있다면, 시각적으로 쉽게 발견할 수 있음.  
표본의 모든 점이 남성용이므로 gender 도 삭제

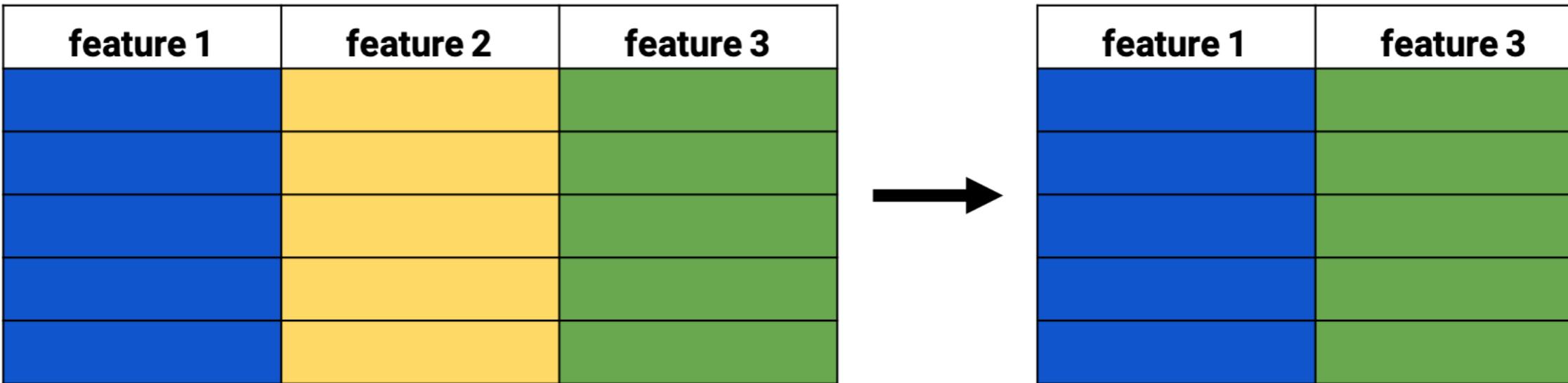


# Feature selection

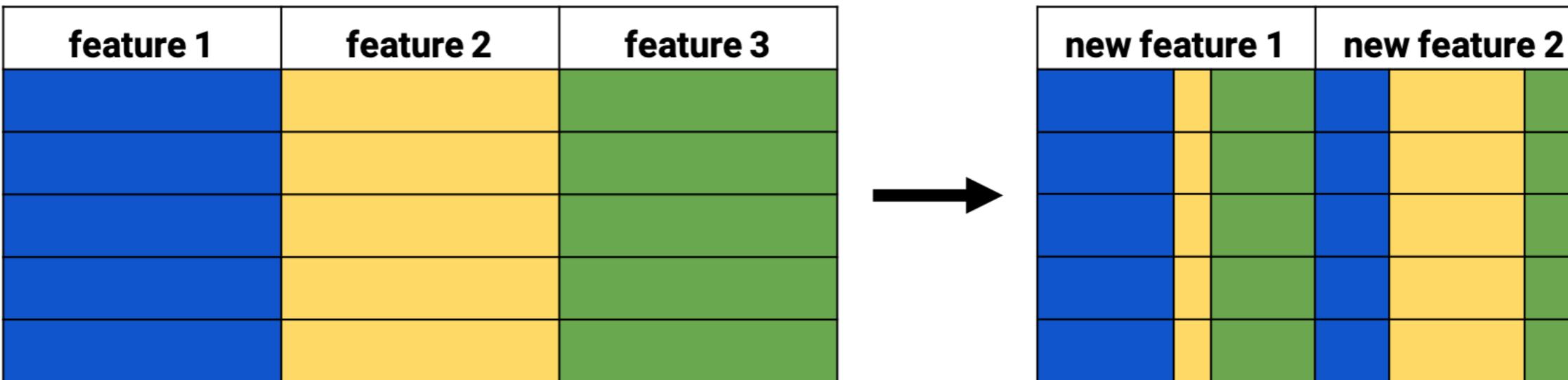


# Feature selection

feature selection, feature extraction 차원성을 줄이는 동일한 목표를 가지고 있음



# Feature extraction

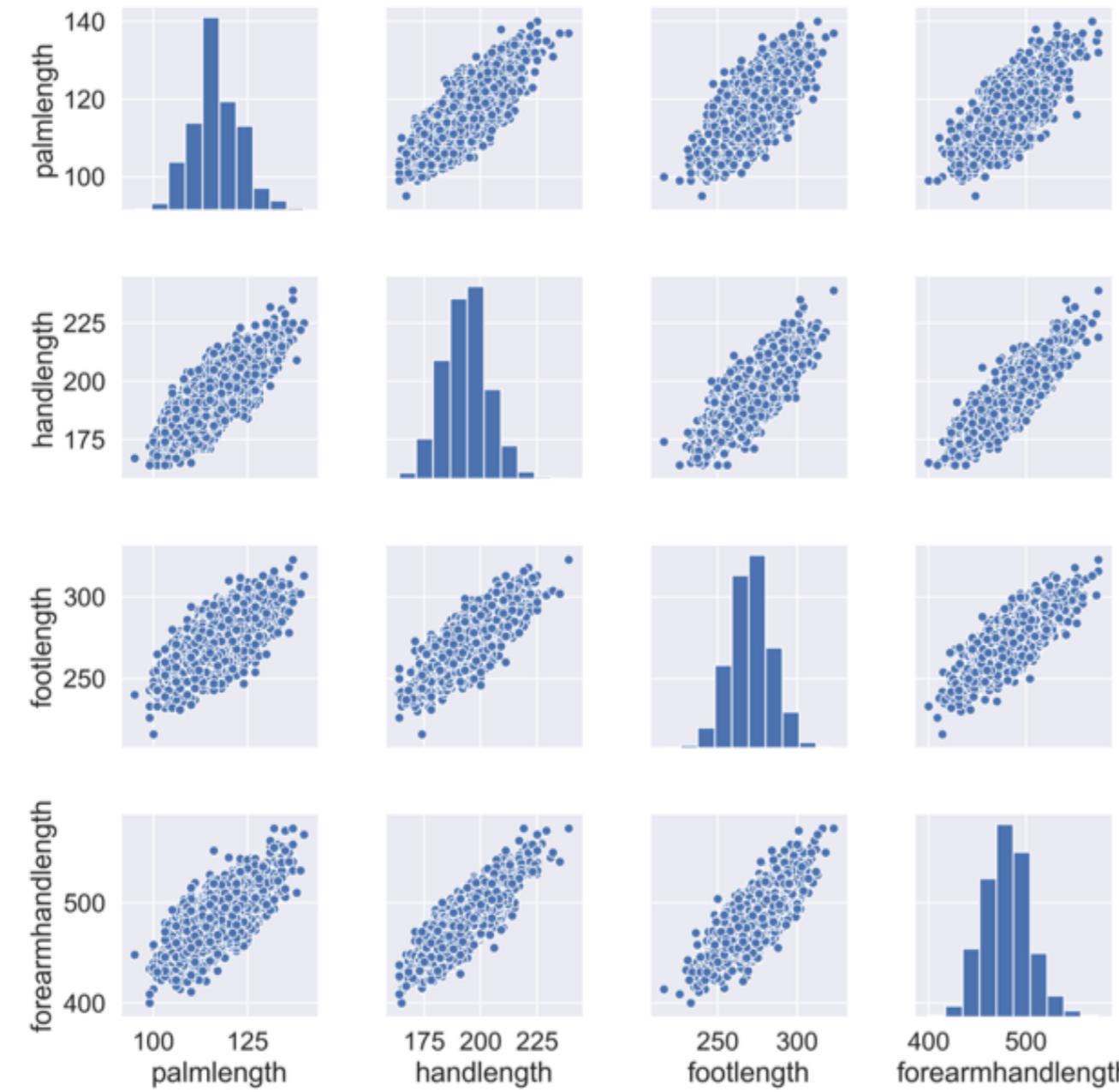


초기 데이터 세트에서 피쳐의 하위 집합을 선택하는 대신 원래 피쳐에서 새 피쳐를 계산하거나 추출합니다.

이러한 새로운 기능들은 중복되는 정보를 최대한 적게 가지고 있기 때문에 그 수가 더 적다.

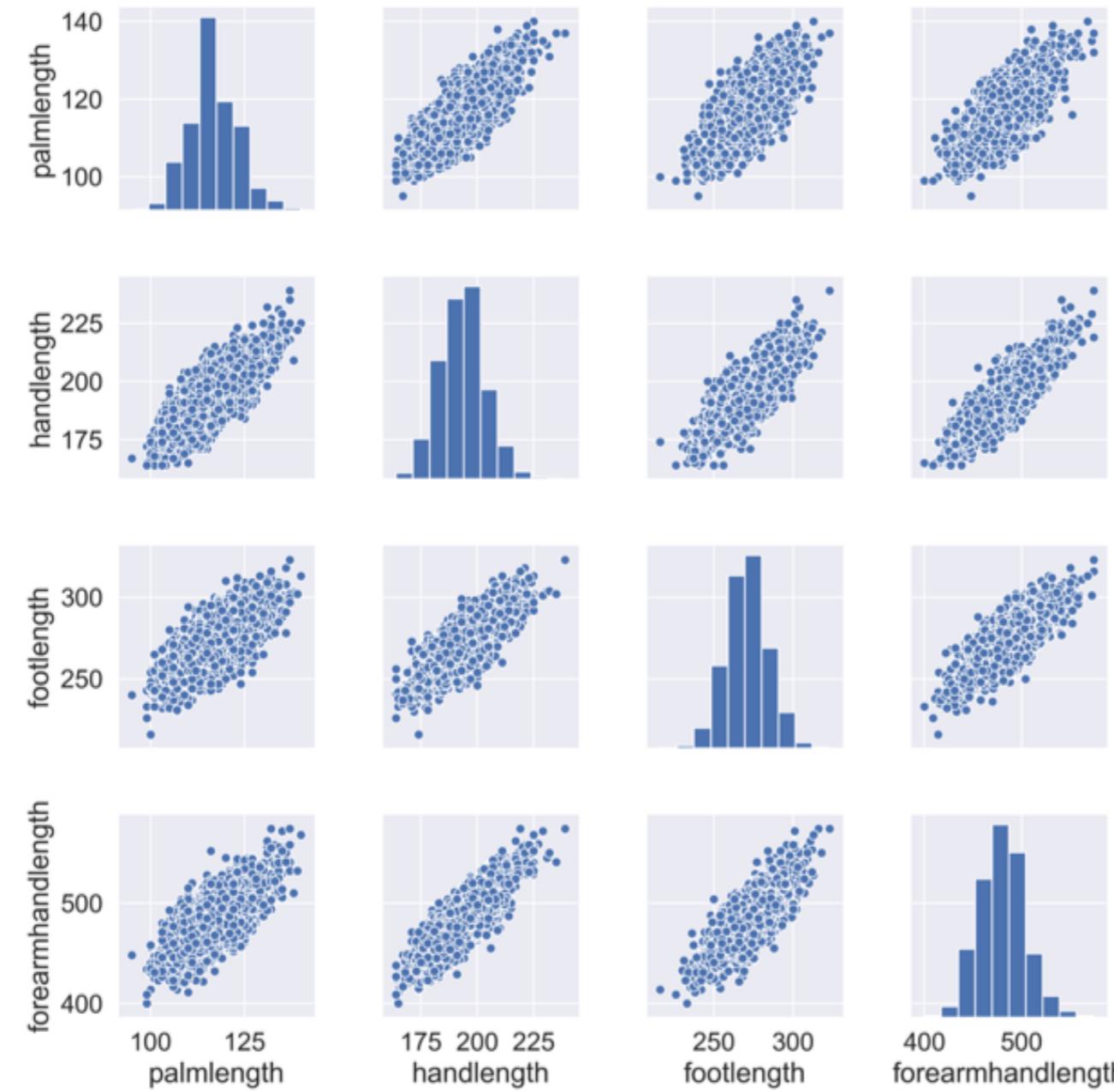
한 가지 단점은 새로 생성된 기능이 원래 기능보다 이해하기 어려운 경우가 많다는 것

# Feature extraction - Example

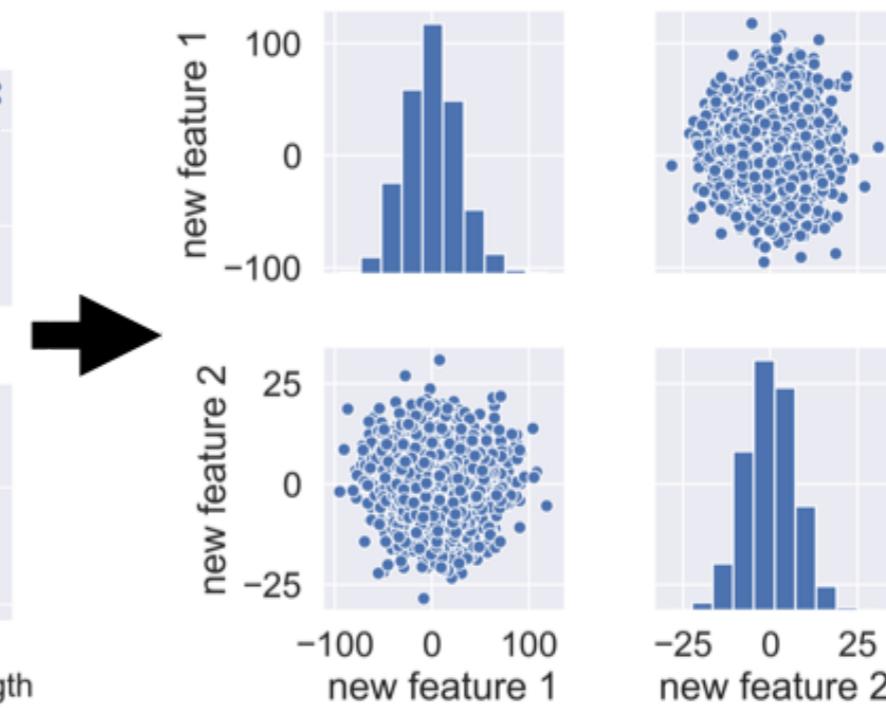


서로 feature간의 상관관계가 있는 것을 알수 있음.  
ex) 긴발, 긴 손, 긴 팔 등을 가지고 있다면  
이 사람이 키가 크다는 것을 예상할 수 있음

# Feature extraction - Example



PCA 또는 주성분 분석이라는 기술을 사용하여 ANSUR dataset sample의 차원수를 4에서 2로 줄이고 data 분산의 96%를 유지할 수 있음.



# **Let's practice!**

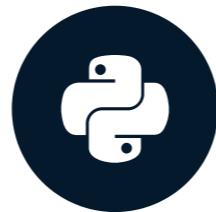
**DIMENSIONALITY REDUCTION IN PYTHON**

# t-SNE visualization of high-dimensional data

DIMENSIONALITY REDUCTION IN PYTHON

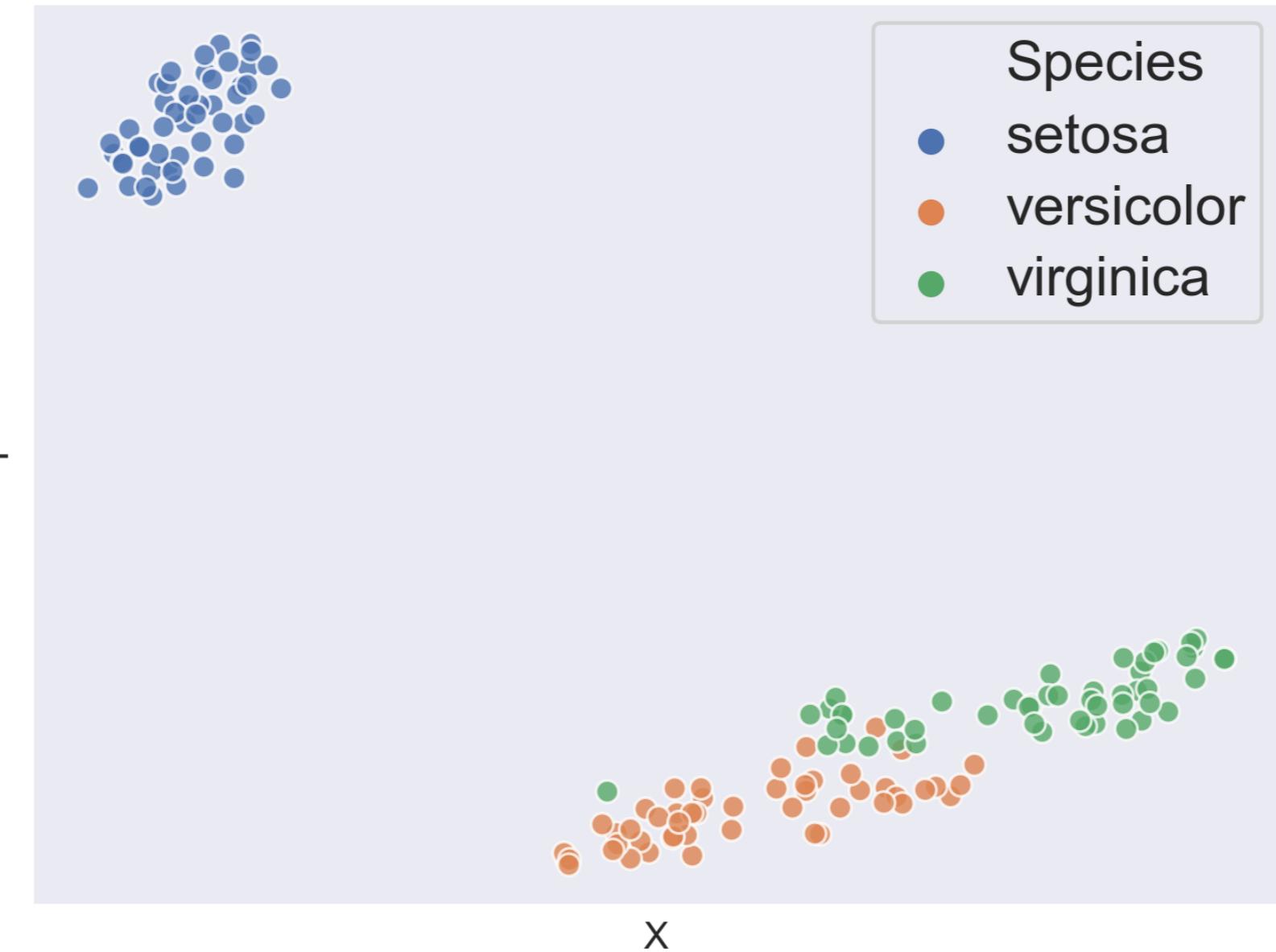
Jeroen Boeye

Machine Learning Engineer, Faktion

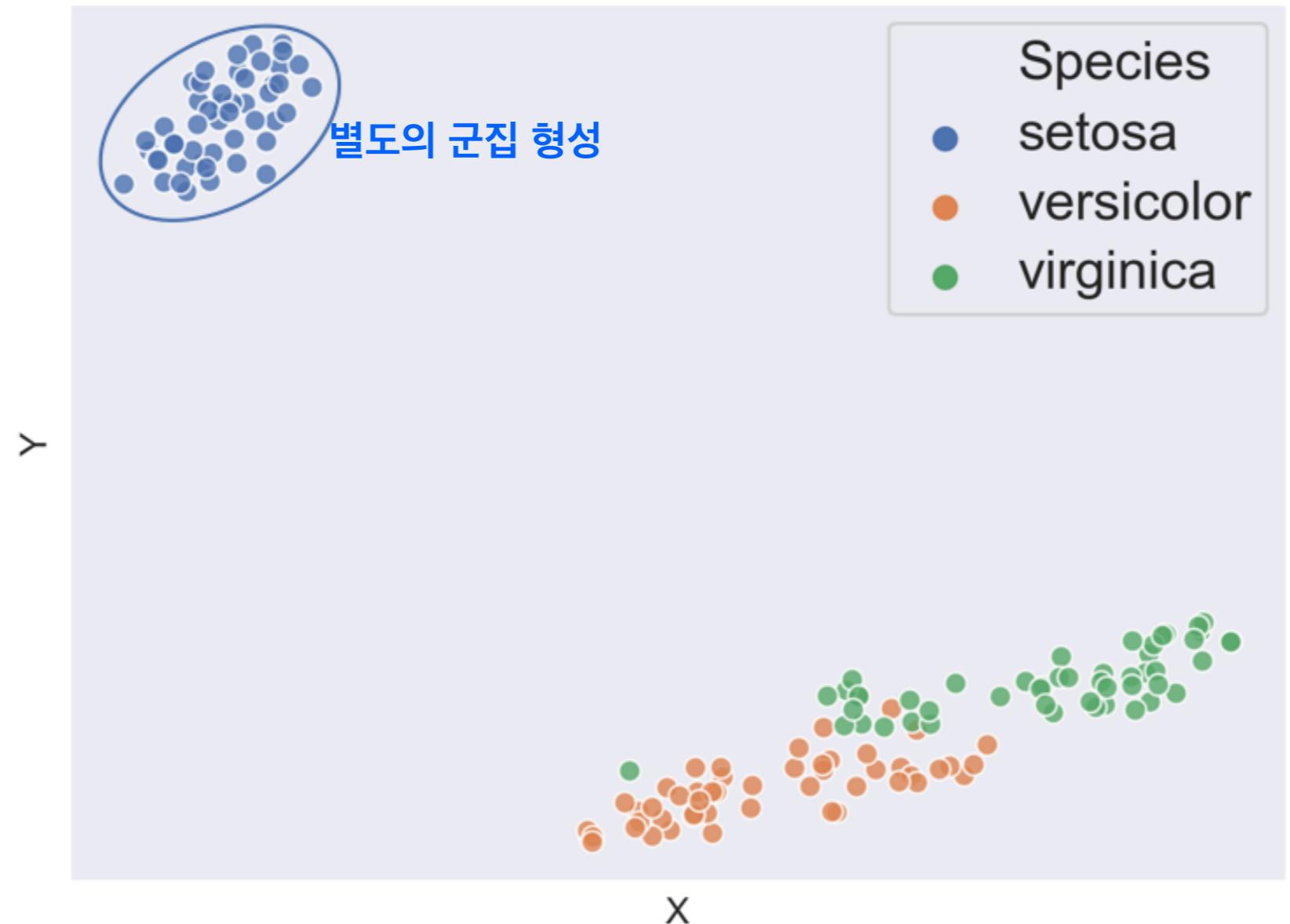


# t-SNE on IRIS dataset

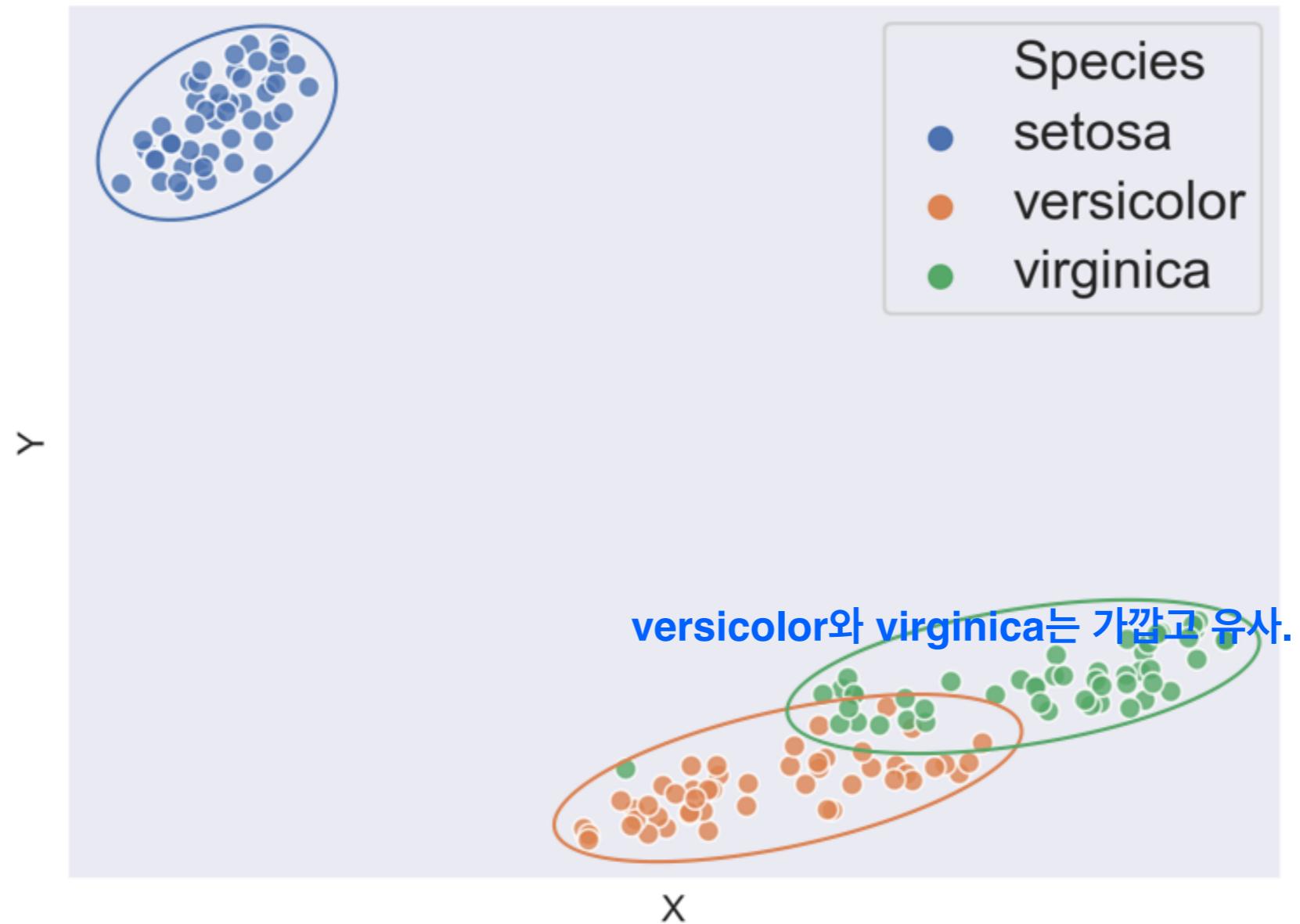
t-SNE는 고차원 공간에서 가장 다른  
관측치 사이의 거리를 2차원 공간에서  
최대화함



# t-SNE on IRIS dataset



# t-SNE on IRIS dataset



# t-SNE on female ANSUR dataset

```
df.shape
```

```
(1986, 99) 99개의 차원을 가지고 있음
```

```
non_numeric = ['BMI_class', 'Height_class',  
               'Gender', 'Component', 'Branch']
```

```
df_numeric = df.drop(non_numeric, axis=1)
```

t-SNE를 적용하기 전에 원하지 않는 열을 `drop()` 메서드를 사용하여  
숫자가 아닌 모든 열을 제거

```
df_numeric.shape
```

```
(1986, 94)
```

# Fitting t-SNE

```
from sklearn.manifold import TSNE  
m = TSNE(learning_rate=50)
```

높은 학습률은 알고리즘이 시도하는 구서에서 더 모험적인 결과를 초래하는 반면  
낮은 학습률은 알고리즘이 보수적인 결과를 초래함  
보통 학습률은 10~1000사이의 범위

```
tsne_features = m.fit_transform(df_numeric)
```

```
tsne_features[1:4,:]
```

```
array([[-37.962185,  15.066088],  
      [-21.873512,  26.334448],  
      [ 13.97476 ,  22.590828]], dtype=float32)
```

# Assigning t-SNE features to our dataset

```
tsne_features[1:4, :]
```

```
array([[-37.962185,  15.066088],  
      [-21.873512,  26.334448],  
      [ 13.97476 ,  22.590828]], dtype=float32)
```

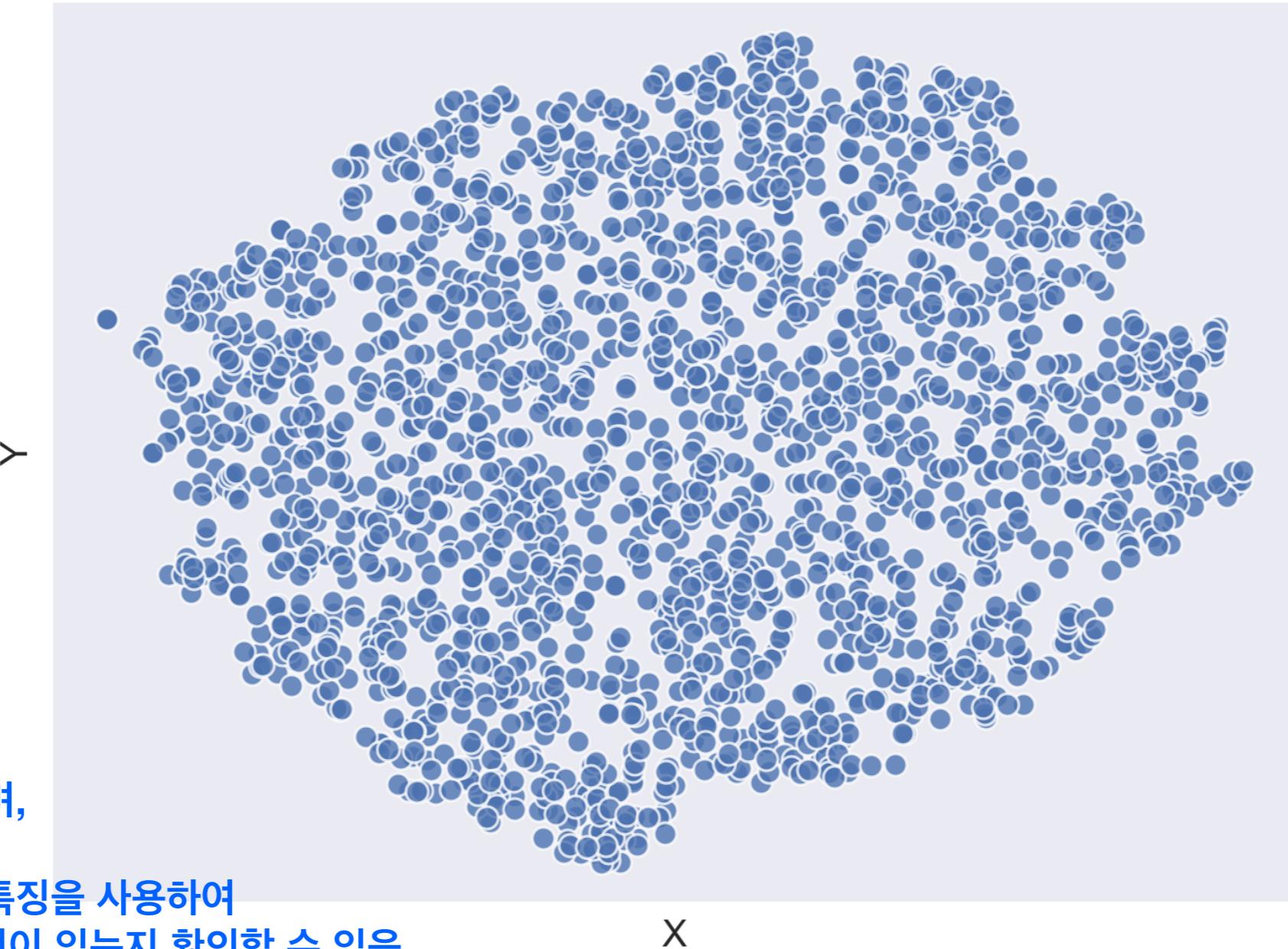
```
df['x'] = tsne_features[:, 0]
```

```
df['y'] = tsne_features[:, 1]
```

# Plotting t-SNE

```
import seaborn as sns  
  
sns.scatterplot(x="x", y="y", data=df)  
  
plt.show()
```

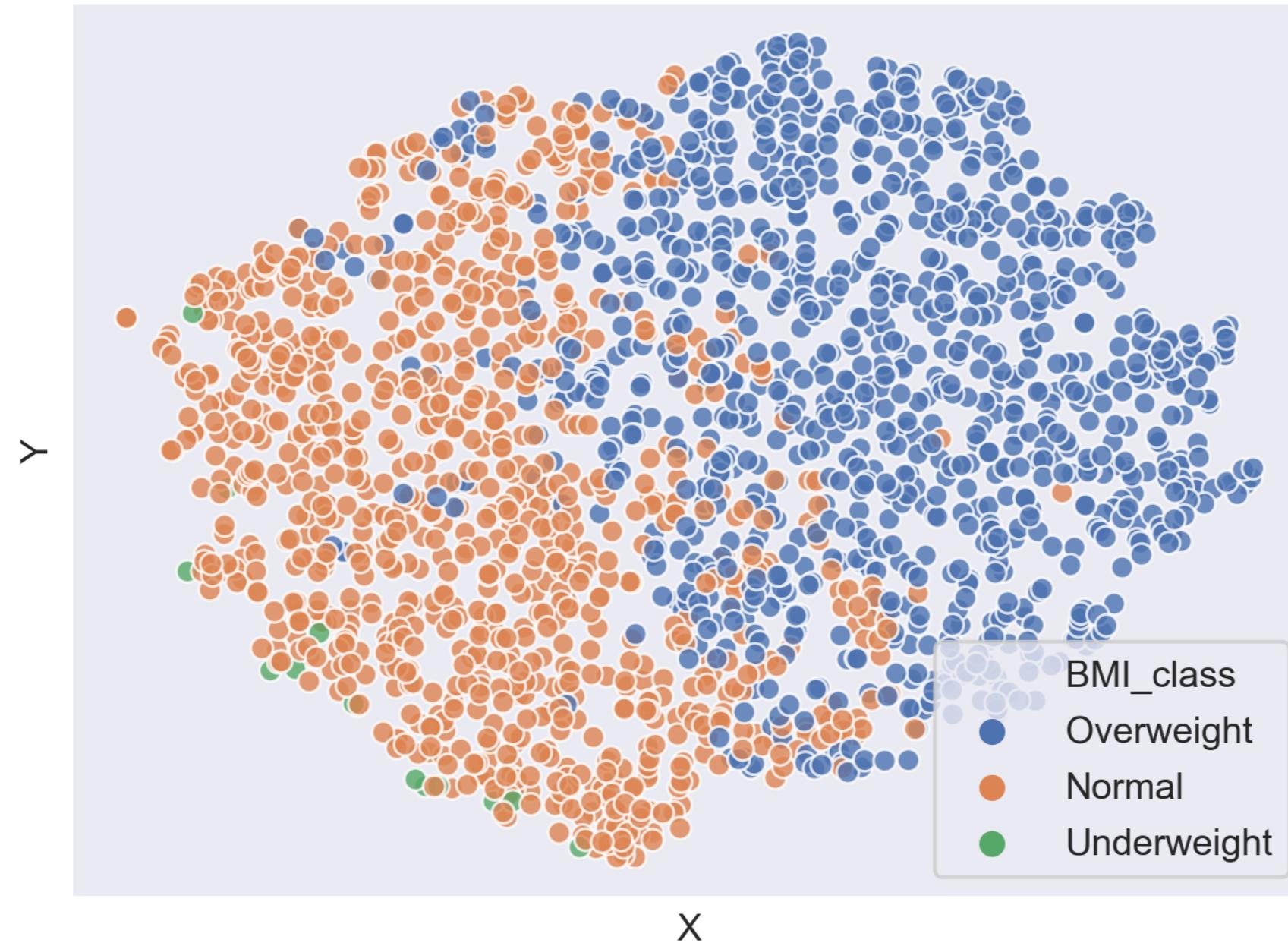
# Plotting t-SNE



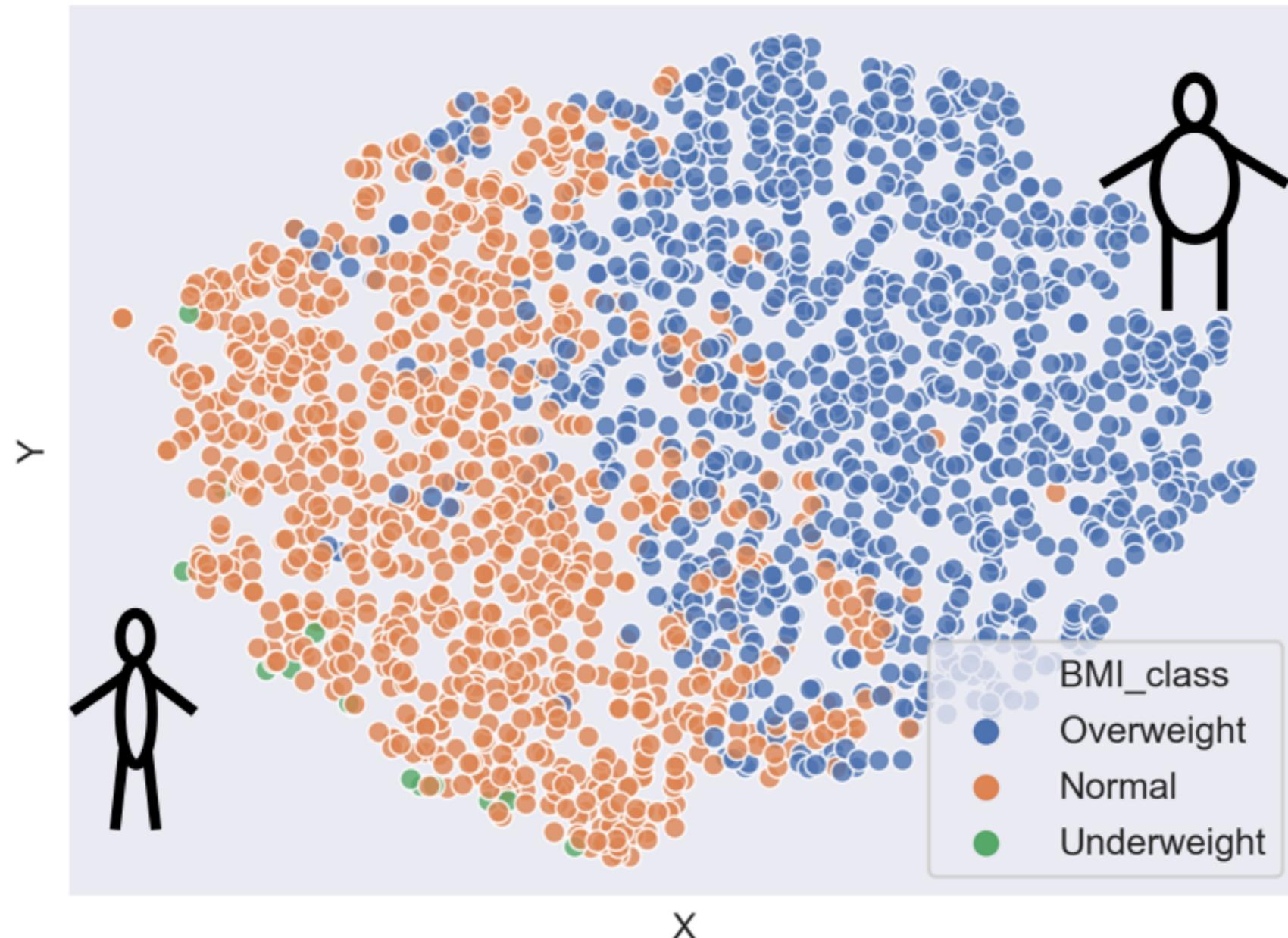
# Coloring points according to BMI category

```
import seaborn as sns  
import matplotlib.pyplot as plt  
  
sns.scatterplot(x="x", y="y", hue='BMI_class', data=df)  
  
plt.show()
```

# Coloring points according to BMI category



# Coloring points according to BMI category



왼쪽에는 저체중자가,  
오른쪽에는 과체중자가 있는 X축을 따라 포인트를 분산시킴

# Coloring points according to height category

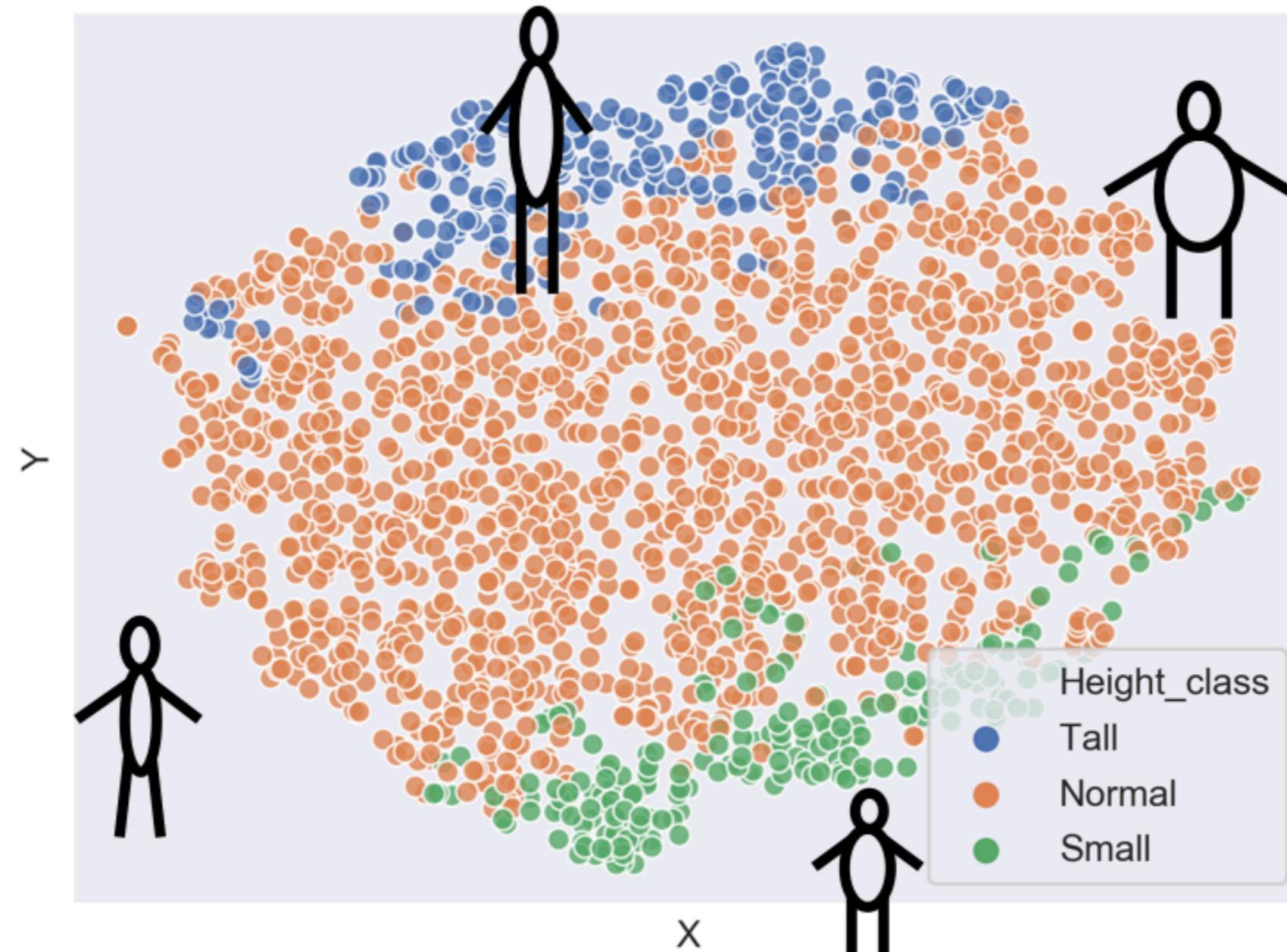
```
import seaborn as sns  
  
import matplotlib.pyplot as plt  
  
sns.scatterplot(x="x", y="y", hue='Height_class', data=df)  
  
plt.show()
```

# Coloring points according to height category

수직 방향에서 사람의 키에 의해 분산이 설명됨



# Coloring points according to height category



키가 큰 사람들이 맨 위에 있고  
키가 작은 사람들이 맨 아래에 있음.

=> t-SNE는 우리가 dataset를 시각적으로 탐색하고  
체형의 가장 중요한 변동 요인을 식별하는데 도움이 됨

# **Let's practice!**

**DIMENSIONALITY REDUCTION IN PYTHON**