# Simple Linear Regression

- Have **two variables**
- One **dependent** and one **independent**

In [ ]:
```
# Import libraries
import numpy as np
import pandas as pd
```

## Import Dataset

In [ ]:
```
# Load data
df = pd.read_csv('../datasets/mldata.csv')
df.head()
```

Out[ ]:

| | age | weight | gender | likeness | height |
|---|---|---|---|---|---|
| **0** | 27 | 76.0 | Male | Biryani | 170.688 |
| **1** | 41 | 70.0 | Male | Biryani | 165 |
| **2** | 29 | 80.0 | Male | Biryani | 171 |
| **3** | 27 | 102.0 | Male | Biryani | 173 |
| **4** | 29 | 67.0 | Male | Biryani | 164 |

## Data Cleaning

In [ ]:
```
# Find missing values
df.isnull().sum()
```

Out[ ]:
```
age         0
weight      0
gender      0
likeness    0
height      0
dtype: int64
```

In [ ]:
```
# Data information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 245 entries, 0 to 244
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   age       245 non-null    int64 
 1   weight    245 non-null    float64
 2   gender    245 non-null    object
 3   likeness  245 non-null    object
 4   height    245 non-null    object
dtypes: float64(1), int64(1), object(3)
memory usage: 9.7+ KB
```

In [ ]:
```
# Convert 'height' data type to int64
df['height'] = df['height'].replace("'", "", regex=True).astype('float64')
```

In [ ]:
```
# Convert categorical columns (gender , likeness) data type to numeric
from sklearn.preprocessing import LabelEncoder
df.gender = LabelEncoder().fit_transform(df.gender)
df.likeness = LabelEncoder().fit_transform(df.likeness)
```

In [ ]:
```
# View dataset
df.head()
```

Out[ ]:

| | age | weight | gender | likeness | height |
|---|---|---|---|---|---|
| **0** | 27 | 76.0 | 1 | 0 | 170.688 |
| **1** | 41 | 70.0 | 1 | 0 | 165.000 |
| **2** | 29 | 80.0 | 1 | 0 | 171.000 |
| **3** | 27 | 102.0 | 1 | 0 | 173.000 |
| **4** | 29 | 67.0 | 1 | 0 | 164.000 |

In [ ]:
```
# Check information and data type again
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 245 entries, 0 to 244
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       245 non-null    int64  
 1   weight    245 non-null    float64
 2   gender    245 non-null    int32  
 3   likeness  245 non-null    int32  
 4   height    245 non-null    float64
dtypes: float64(2), int32(2), int64(1)
memory usage: 7.8 KB
```

## Split Data into Train and Test

In [ ]:
```
# Split data into input (X) and output (y)
X = df[['age', 'weight', 'gender', 'likeness']]
y = df['height']
```

In [ ]:
```
X.head()
```

Out[ ]:

| | age | weight | gender | likeness |
|---|---|---|---|---|
| **0** | 27 | 76.0 | 1 | 0 |
| **1** | 41 | 70.0 | 1 | 0 |
| **2** | 29 | 80.0 | 1 | 0 |
| **3** | 27 | 102.0 | 1 | 0 |
| **4** | 29 | 67.0 | 1 | 0 |

In [ ]:
```
y.head()
```

Out[ ]:
```
0    170.688
1    165.000
2    171.000
3    173.000
4    164.000
Name: height, dtype: float64
```

In [ ]:
```
# Split data into train and test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

In [ ]:
```
# View training dataset
print(f'Training inputs:\n{X_train.head()}')
print(f'Training outputs:\n{y_train.head()}')
print(f'Training input shape: {X_train.shape}')
print(f'Training input shape: {y_train.shape}')
```

```
Training inputs:
      age  weight  gender  likeness
15    27    78.0       1         2
158   22    67.0       1         0
7     34    98.0       1         0
159   27    75.0       1         0
207   28    62.0       0         0
Training outputs:
15     174.0
158    181.0
7      176.5
159    152.4
207      5.3
Name: height, dtype: float64
Training input shape: (196, 4)
Training input shape: (196,)
```

In [ ]:
```
# View testing dataset
print(f'Test inputs:\n{X_test.head()}')
print(f'Test outputs:\n{y_test.head()}')
print(f'Test input shape: {X_test.shape}')
print(f'Test input shape: {y_test.shape}')
```

```
Test inputs:
      age  weight  gender  likeness
92    24    87.0       1         0
238   22    50.0       1         2
73    25    72.5       1         0
55    30    63.0       1         2
101   50    88.0       1         0
Test outputs:
92     173.0
238      6.0
73     173.3
55     180.0
181    188.0
Name: height, dtype: float64
Test input shape: (49, 4)
Test input shape: (49,)
```

## Fit Linear Regression Model

In [ ]:
```
from sklearn.linear_model import LinearRegression
# Create model
model = LinearRegression()
# Fit model
model = model.fit(X_train, y_train)
print('Training completed!!')
```

```
Training completed!!
```

## Make Predictions

In [ ]:
```
y_preds = model.predict(X_test)
y_preds[:10] # first 10 predictions
```

Out[ ]:
```
array([153.49460602, 142.71933451, 149.20152971, 149.44010418,
       162.20573589, 150.97023314, 144.78229321, 143.6306448 ,
       137.7408804 , 150.01968777])
```

## Model Evaluation

In [ ]:
```
# Find MAE and MSE on models predictions with test labels
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error

mae = mean_absolute_error(y_test, y_preds)
mse = mean_squared_error(y_test, y_preds)
print(f'Model MAE score: {mae}')
print(f'Model MSE score: {mse}')
```

```
Model MAE score: 40.64544094885285
Model MSE score: 3495.0973392418664
```