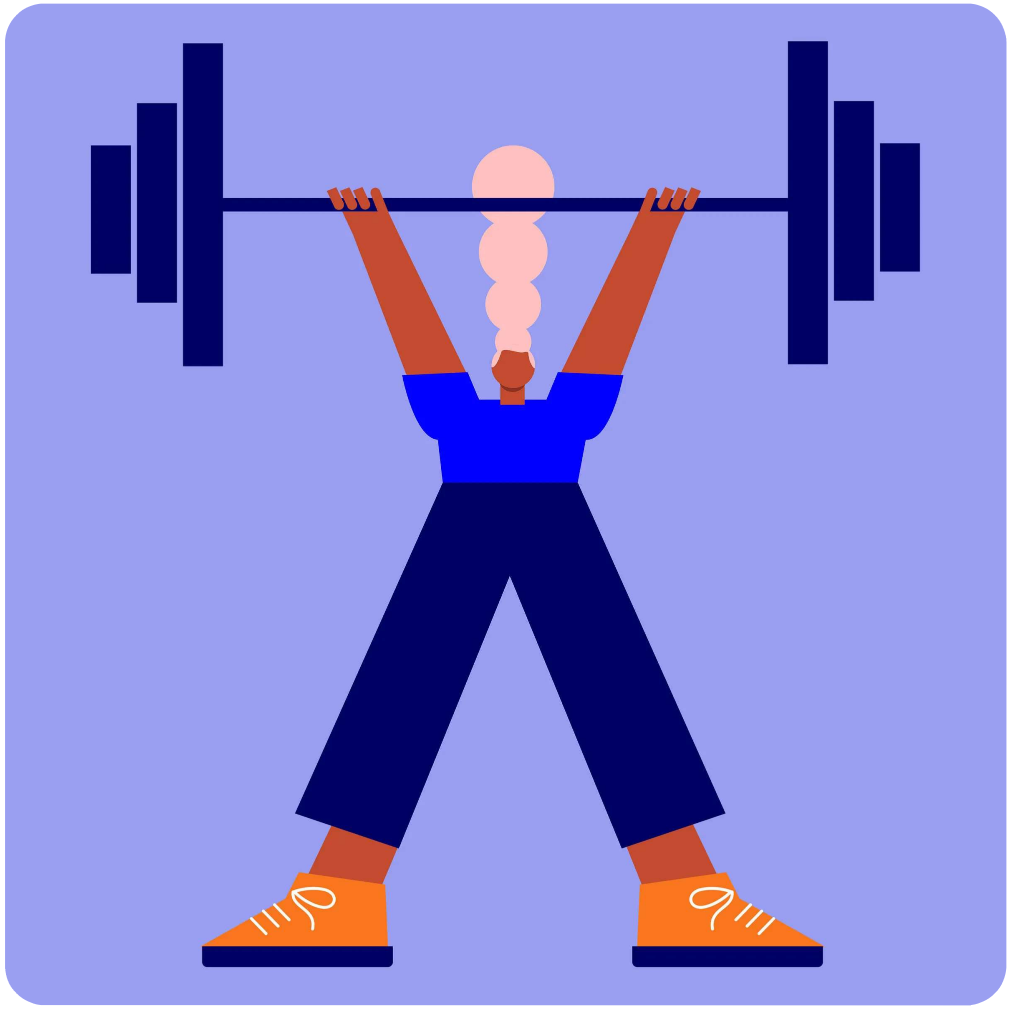
Fitness Club



This notebook looks into using various Pyhton-based machine learning and data science libraries in an attempt to build a machine learning model capable of predicting a member will not attend the class, they can make another space available..

We're going to take the following approach:

1. Problem definition
2. Data
3. Evaluation
4. Features
5. Modeling
6. Experimentation

1. Problem Definition

In a statement,

> Given Many features of popular brands' phones, including price, ram, storage etc. Can we predict the price for the phones ?

2. Data

GoalZone is a fitness club chain in Canada. GoalZone offers a range of fitness classes in two capacities - 25 and 15. Some classes are always fully booked. Fully booked classes often have a low attendance rate. GoalZone wants to increase the number of spaces available for classes. They want to do this by predicting whether the member will attend the class or not.

https://www.kaggle.com/datasets/ddosad/datacamps-data-science-associate-certification (https://www.kaggle.com/datasets/ddosad/datacamps-data-science-associate-certification)

3. Evaluation

> If we can predict a member will not attend the class, they can make another space available.

4. Features

Preparing Tools

We're going to use pandas, matplotlib and numpy for data analysis and manipulations

In [1]:

```
# Import Tools


import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```python
df_fitnes = pd.read_csv("fitness_class_2212.csv")
df_fitnes
```

	booking_id	months_as_member	weight	days_before	day_of_week	time	category	attended
0	1	17	79.56	8	Wed	PM	Strength	0
1	2	10	79.01	2	Mon	AM	HIIT	0
2	3	16	74.53	14	Sun	AM	Strength	0
3	4	5	86.12	10	Fri	AM	Cycling	0
4	5	15	69.29	8	Thu	AM	HIIT	0
...	...	...	...	...	...	...	...	...
1495	1496	21	79.51	10	Fri	AM	HIIT	0
1496	1497	29	89.55	2	Mon	AM	Strength	0
1497	1498	9	87.38	4	Tue	AM	HIIT	0
1498	1499	34	68.64	14	Sun	AM	Aqua	0
1499	1500	20	94.39	8	Thu	AM	Cycling	1

```
1500 rows × 8 columns
```

Data Understanding

In [3]:
```
df_fitnes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   booking_id        1500 non-null   int64  
 1   months_as_member  1500 non-null   int64  
 2   weight            1480 non-null   float64
 3   days_before       1500 non-null   object 
 4   day_of_week       1500 non-null   object 
 5   time              1500 non-null   object 
 6   category          1500 non-null   object 
 7   attended          1500 non-null   int64  
dtypes: float64(1), int64(3), object(4)
memory usage: 93.9+ KB
```

In [4]:
```python
for col in df_fitnes.columns:
    print(col, '-->', len(col), '-->', df_fitnes[col].unique())
```

```
booking_id --> 10 --> [   1    2    3 ... 1498 1499 1500]
months_as_member --> 16 --> [ 17  10  16   5  15   7  11   9  23  13   8  22   6  33  24  14   2  12
  26  28  27   1   3  21  18  19  53  20  34  25  32  73  55   4  35  54
  76  62  42 105  90  29  60  30 107  52  37  38  48  51  40  89  57  36
  44  39  41  47  58  66  45  43  61  50  65  31  97  59  93 148 111  69]
weight --> 6 --> [79.56 79.01 74.53 ... 87.38 68.64 94.39]
days_before --> 11 --> ['8' '2' '14' '10' '6' '4' '9' '12' '5' '3' '7' '13' '12 days' '20' '1'
 '15' '6 days' '11' '13 days' '3 days' '16' '1 days' '7 days' '8 days'
 '10 days' '14 days' '17' '5 days' '2 days' '4 days' '29']
day_of_week --> 11 --> ['Wed' 'Mon' 'Sun' 'Fri' 'Thu' 'Wednesday' 'Fri.' 'Tue' 'Sat' 'Monday']
time --> 4 --> ['PM' 'AM']
category --> 8 --> ['Strength' 'HIIT' 'Cycling' 'Yoga' '-' 'Aqua']
attended --> 8 --> [0 1]
```

There are some data that we need to correction in this dataset, like in columns `days_before` , `day_of_week` , and `category`

In [5]:
```python
df_fitnes['days_before'] = df_fitnes['days_before'].str.replace(' days', '').astype(int)
```

In [6]:
```python
df_fitnes['days_before'].unique()
```

```
array([ 8,  2, 14, 10,  6,  4,  9, 12,  5,  3,  7, 13, 20,  1, 15, 11, 16,
       17, 29])
```

```
In [7]:  df_fitnes['day_of_week'] = df_fitnes['day_of_week'].str[:3]
         day_mapping={
             'Mon':1,
             'Tue':2,
             'Wed':3,
             'Thu':4,
             'Fri':5,
             'Sat':6,
             'Sun':7,
         }

         df_fitnes['day_of_week'] = df_fitnes['day_of_week'].map(day_mapping)
```

```
In [8]:  for col in df_fitnes.columns:
             print(col, '-->', len(col), '-->', df_fitnes[col].unique())
```

```
booking_id --> 10 --> [   1    2    3 ... 1498 1499 1500]
months_as_member --> 16 --> [ 17  10  16   5  15   7  11   9  23  13   8  22   6  33  24  14   2  12
  26  28  27   1   3  21  18  19  53  20  34  25  32  73  55   4  35  54
  76  62  42 105  90  29  60  30 107  52  37  38  48  51  40  89  57  36
  44  39  41  47  58  66  45  43  61  50  65  31  97  59  93 148 111  69]
weight --> 6 --> [79.56 79.01 74.53 ... 87.38 68.64 94.39]
days_before --> 11 --> [ 8  2 14 10  6  4  9 12  5  3  7 13 20  1 15 11 16 17 29]
day_of_week --> 11 --> [3 1 7 5 4 2 6]
time --> 4 --> ['PM' 'AM']
category --> 8 --> ['Strength' 'HIIT' 'Cycling' 'Yoga' '-' 'Aqua']
attended --> 8 --> [0 1]
```

```
In [9]: df_fitnes['category'] = df_fitnes['category'].replace('-', 'Unknown')
        df_fitnes['category'].unique()
```

```
array(['Strength', 'HIIT', 'Cycling', 'Yoga', 'Unknown', 'Aqua'],
      dtype=object)
```

```
In [10]: df_fitnes.describe()
```

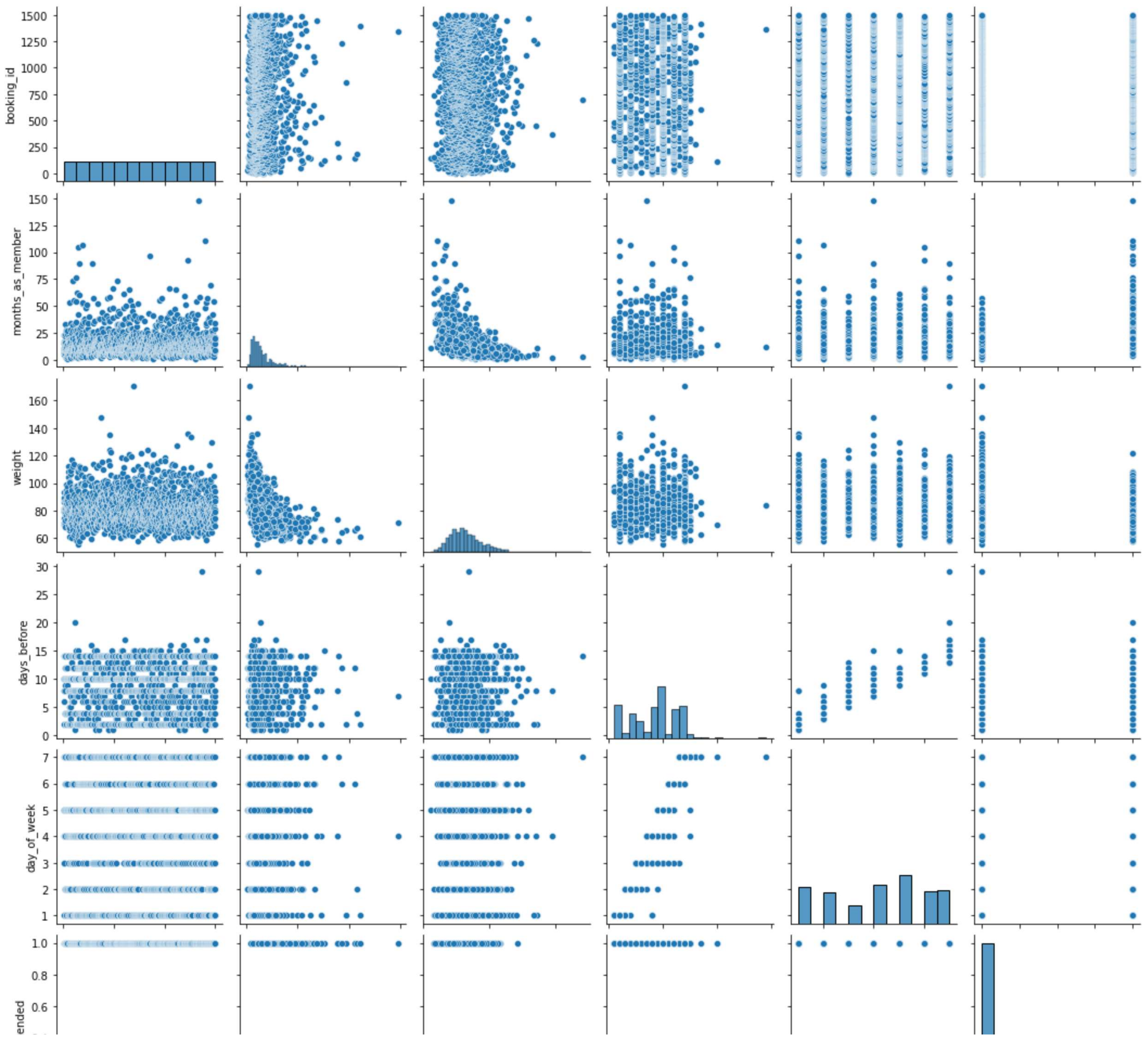
	booking_id	months_as_member	weight	days_before	day_of_week	attended
count	1500.000000	1500.000000	1480.000000	1500.000000	1500.000000	1500.000000
mean	750.500000	15.628667	82.610378	8.346667	4.105333	0.302667
std	433.157015	12.926543	12.765859	4.077938	1.994214	0.459565
min	1.000000	1.000000	55.410000	1.000000	1.000000	0.000000
25%	375.750000	8.000000	73.490000	4.000000	2.000000	0.000000
50%	750.500000	12.000000	80.760000	9.000000	4.000000	0.000000
75%	1125.250000	19.000000	89.520000	12.000000	6.000000	1.000000
max	1500.000000	148.000000	170.520000	29.000000	7.000000	1.000000

```
In [11]: df_fitnes.describe(include=['object'])
```

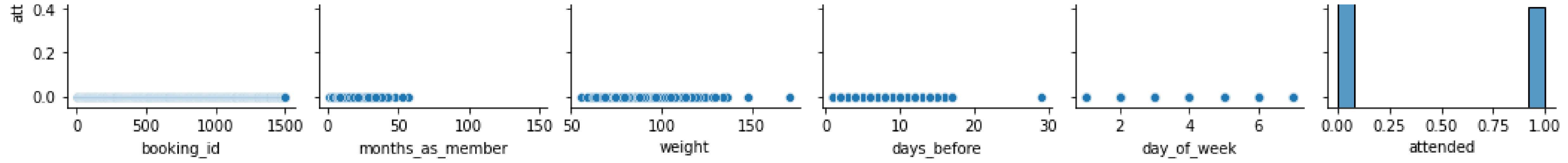
	time	category
count	1500	1500
unique	2	6
top	AM	HIIT
freq	1141	667

In [12]:
```
plt.figure(figsize=(14,14))
sns.pairplot(df_fitnes)
plt.show()
```

```
<Figure size 1008x1008 with 0 Axes>
```



booking_id
months_as_member
weight
days_before
day_of_week
ended
1500
1250
1000
750
500
250
0
150
125
100
75
50
25
0
160
140
120
100
80
60
30
25
20
15
10
5
0
7
6
5
4
3
2
1
1.0
0.8
0.6

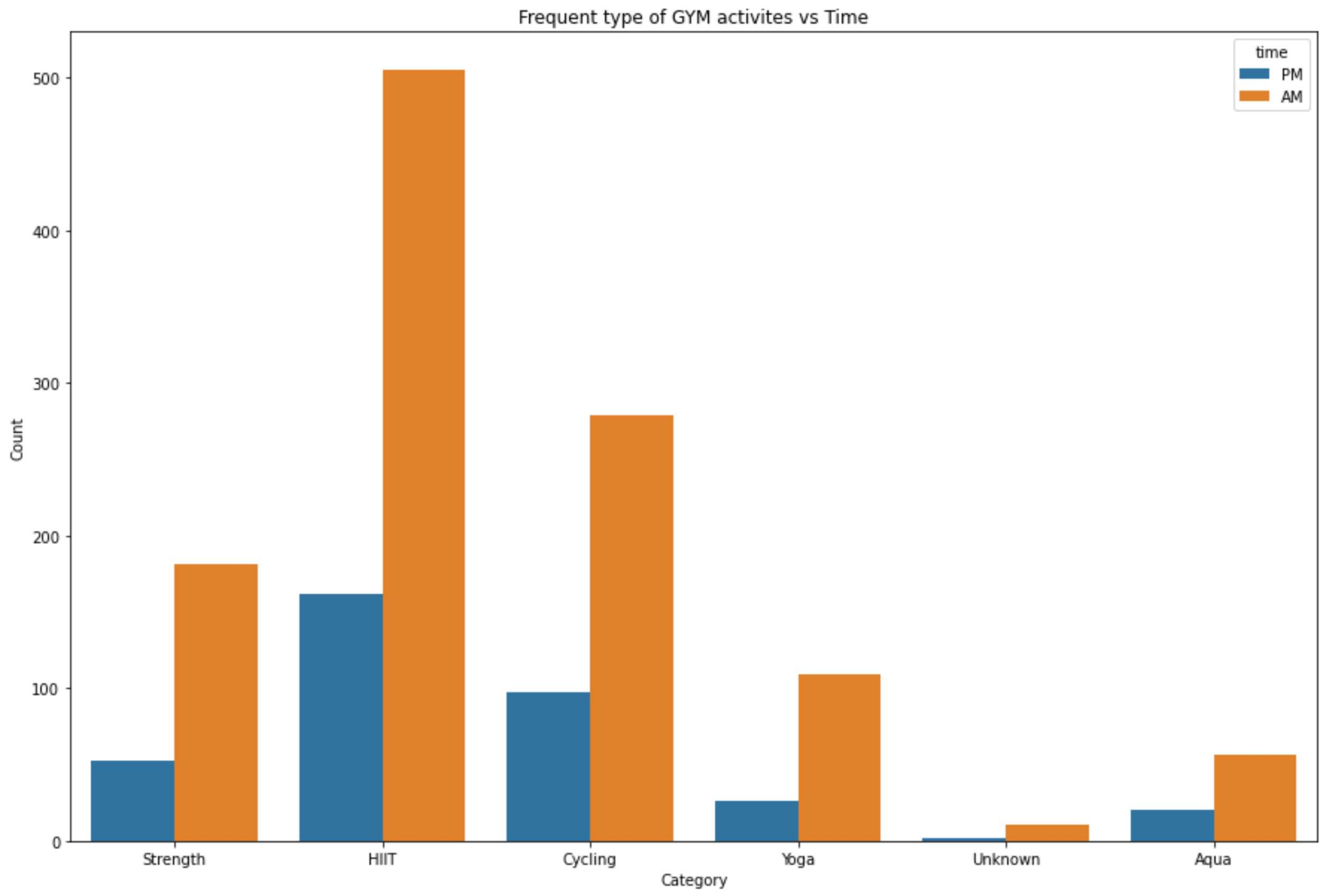


In [13]:

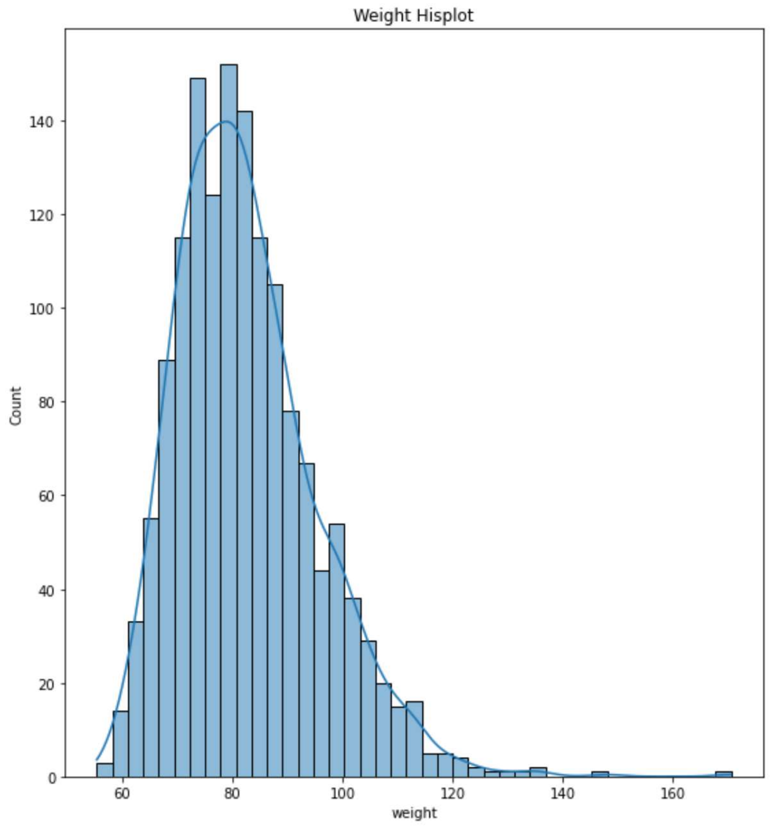
```
df_fitnes['weight'].isnull().sum()
```

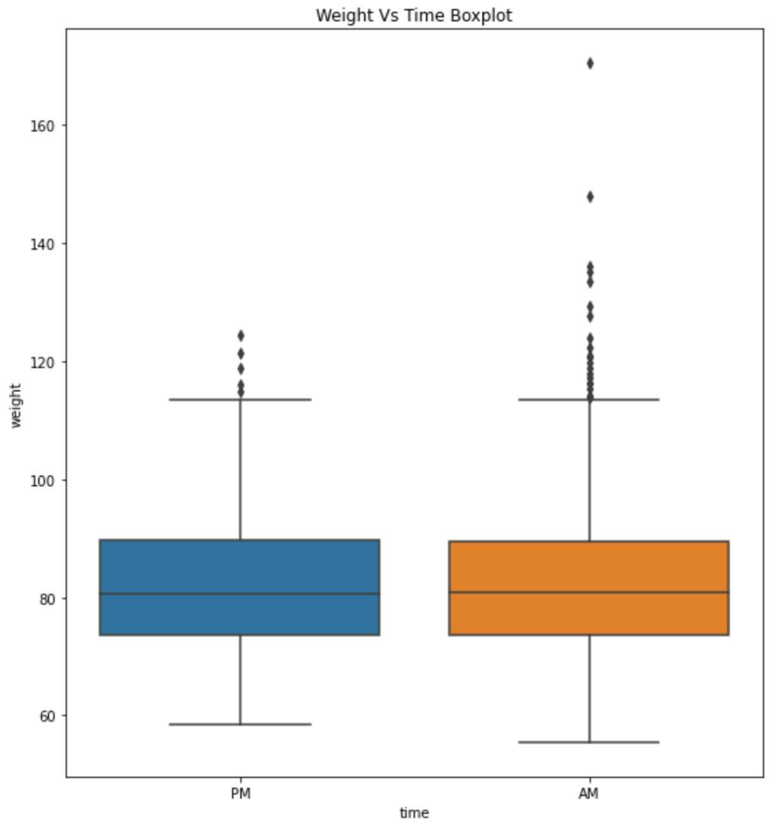
```
20
```

```
In [14]:
# Data Visualization
plt.figure(figsize=(15,10))
sns.countplot(df_fitnes, x='category', hue='time')
plt.xlabel('Category')
plt.ylabel('Count')
plt.title('Frequent type of GYM activites vs Time')
plt.show();
```


Frequent type of GYM activites vs Time
time
PM
AM
Count
0
100
200
300
400
500
Strength
HIIT
Cycling
Yoga
Unknown
Aqua
Category

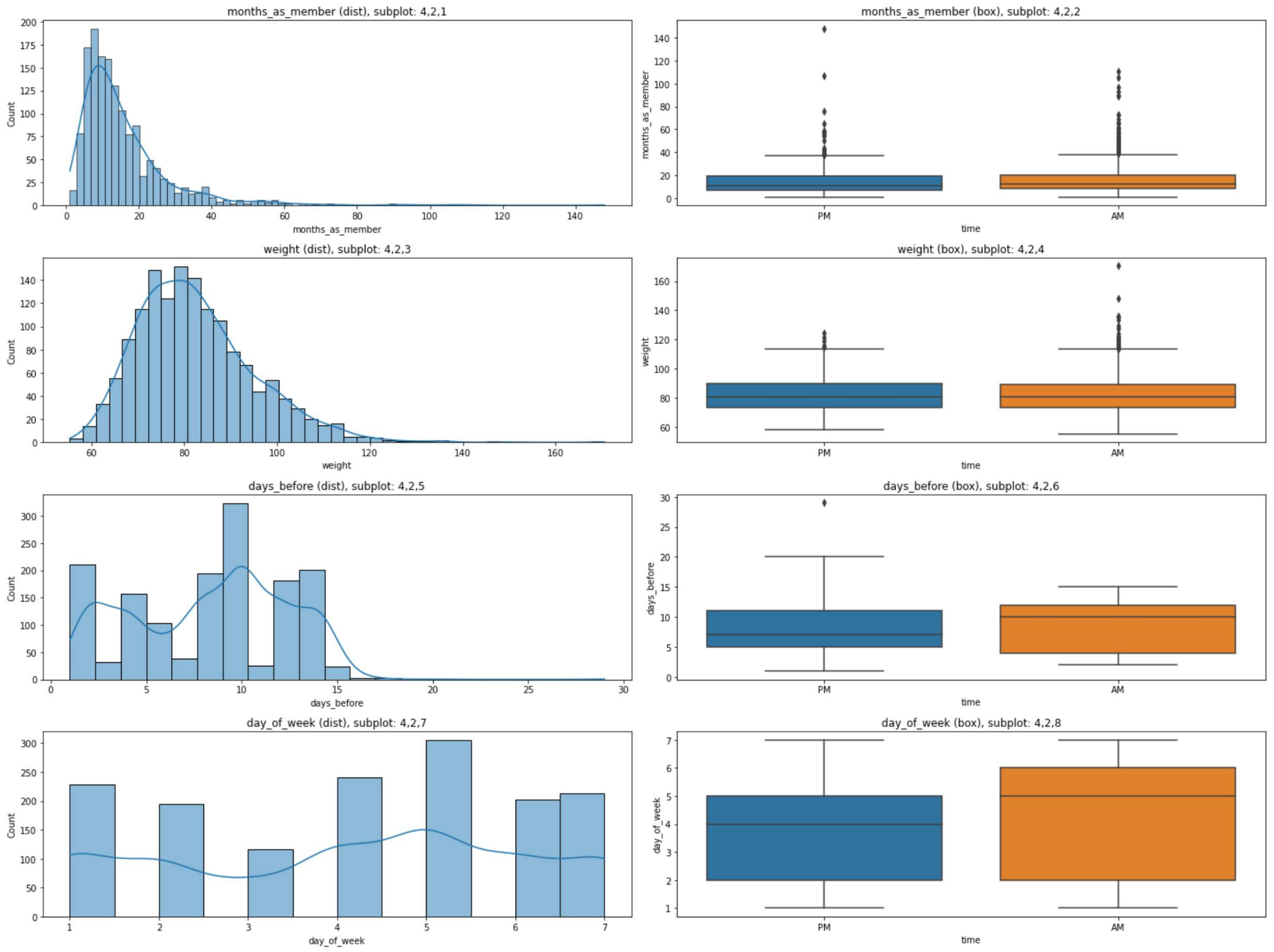
```
In [15]:
# Historgram plot
plt.figure(figsize=(20,10))

plt.subplot(121)
plt.title('Weight Hisplot')
sns.histplot(df_fitnes['weight'], kde='True')

plt.subplot(122)
plt.title('Weight Vs Time Boxplot')
sns.boxplot(df_fitnes, x='time', y='weight', orient='v')

plt.show();
```


Weight Hisplot
Count
weight
0
20
40
60
80
100
120
140
60
80
100
120
140
160


Weight Vs Time Boxplot
weight
time
PM
AM
60
80
100
120
140
160

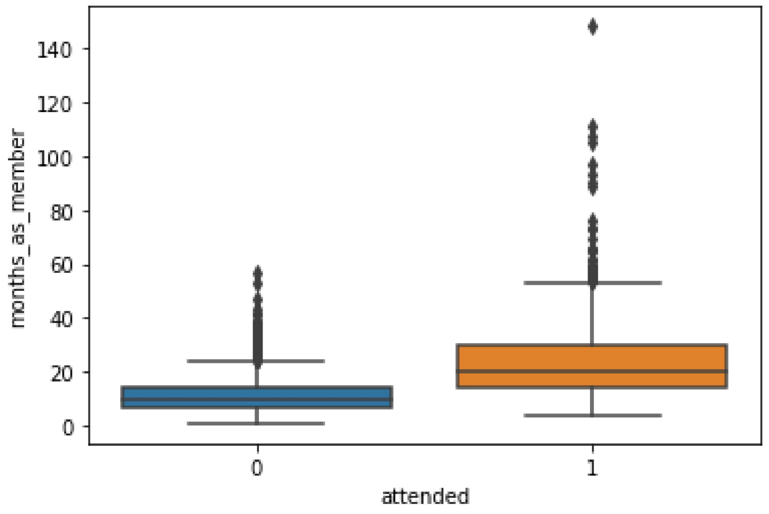
In [16]:

```python
col_hist = ['months_as_member', 'weight', 'days_before', 'day_of_week']

row = len(col_hist)
col = 2
counter = 1

plt.figure(figsize=(20,15))

for i in col_hist:
    plt.subplot(row, col, counter)
    plt.title('{} (dist), subplot: {},{},{}'.format(i, row, col, counter))
    sns.histplot(df_fitnes[i], kde='True')
    counter = counter + 1

    plt.subplot(row, col, counter)
    plt.title('{} (box), subplot: {},{},{}'.format(i, row, col, counter))
    sns.boxplot(df_fitnes, x='time', y=i, orient='v')
    counter = counter + 1

plt.tight_layout()
plt.show()
```


months_as_member (dist), subplot: 4,2,1
Count
months_as_member
months_as_member (box), subplot: 4,2,2
months_as_member
PM
AM
time
weight (dist), subplot: 4,2,3
Count
weight
weight (box), subplot: 4,2,4
weight
PM
AM
time
days_before (dist), subplot: 4,2,5
Count
days_before
days_before (box), subplot: 4,2,6
days_before
PM
AM
time
day_of_week (dist), subplot: 4,2,7
Count
day_of_week
day_of_week (box), subplot: 4,2,8
day_of_week
PM
AM
time

```
In [17]: sns.boxplot(data=df_fitnes, x='attended', y='months_as_member', orient='v')
```

```
<AxesSubplot:xlabel='attended', ylabel='months_as_member'>
```



```
In [ ]:
```