Importing the dataset

First of all the excel file must be read.

In [1]:
```
import pandas as pd


file = pd.read_excel('data.xlsx', engine='openpyxl')
file
```

```
/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-pa
ckages/openpyxl/worksheet/_reader.py:329: UserWarning: Unknown extension
is not supported and will be removed
  warn(msg)
```

Out[1]:

	date	Time	Ave. wind speed-40m	Max. wind speed-40m	Min. wind speed-40m	SDev. wind speed-40m	Ave. wind speed-30m	Max. wind speed-30m	Min. wind speed-30m	SDev. wind speed-30m	...	Ma wi spee 1
0	2014-12-06 00:00:00	06:30:00	1.5	2.0	1.0	0.15	1.0	2.1	0.6	0.36	...	(
1	2014-12-06 00:00:00	07:40:00	1.4	1.6	0.7	0.21	1.1	1.8	0.7	0.22	...	(
2	2014-12-06 00:00:00	07:50:00	1.2	1.6	0.0	0.33	1.3	1.9	0.0	0.28	...	(

Cleaning and normalizing the dataset

Wind Speed is the only value that matters. Therefore other extra columns are deleted.

In [2]:
```

count = 1
for column in file.columns:
    if "speed" not in column:
        file.drop([column], axis=1, inplace=True)

file
```

Out[2]:

	Ave. wind speed-40m	Max. wind speed-40m	Min. wind speed-40m	SDev. wind speed-40m	Ave. wind speed-30m	Max. wind speed-30m	Min. wind speed-30m	SDev. wind speed-30m	Ave. wind speed-10m	Max. wind speed-10m	Min. wind speed-10m	S sp
0	1.5	2.0	1.0	0.15	1.0	2.1	0.6	0.36	0.0	0.0	0.0	
1	1.4	1.6	0.7	0.21	1.1	1.8	0.7	0.22	0.0	0.0	0.0	
2	1.2	1.6	0.0	0.33	1.3	1.9	0.0	0.28	0.0	0.0	0.0	
3	0.6	0.9	0.2	0.14	0.8	1.1	0.4	0.16	0.0	0.0	0.0	
4	0.2	0.9	0.0	0.31	0.3	1.2	0.0	0.42	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	
46617	3.5	4.6	2.4	0.50	3.4	4.4	2.2	0.44	3.3	5.1	2.2	
46618	3.5	5.7	2.0	0.71	3.2	4.7	1.9	0.54	3.0	4.7	1.7	
46619	3.3	4.7	1.9	0.53	3.3	4.7	2.1	0.56	3.1	4.5	1.5	

Providing a properly structured dataframe

Most of the speed values are zero, which should be removed due to causing problems in the algorithm.

Then, a smaller dataframe containing all the non-zero speed values along with the number of repetitions and the probability of each occurrence will result.

In [3]:

```
speedValues = []
for i in range(len(file)):
    for column in file.columns:
        speed = round(float(file[column][i]) , 1)
        if speed != 0.0:
            speedValues.append(speed)
```

In [4]:

```
newData = {"Speed" : list(set(speedValues)) , "Repetition Number" : [] , "P

for item in newData["Speed"]:
    m = speedValues.count(item)
    newData["Repetition Number"].append(m)
    newData["Probability"].append(m / len(speedValues))

df = pd.DataFrame(newData)
df
```

Out[4]:

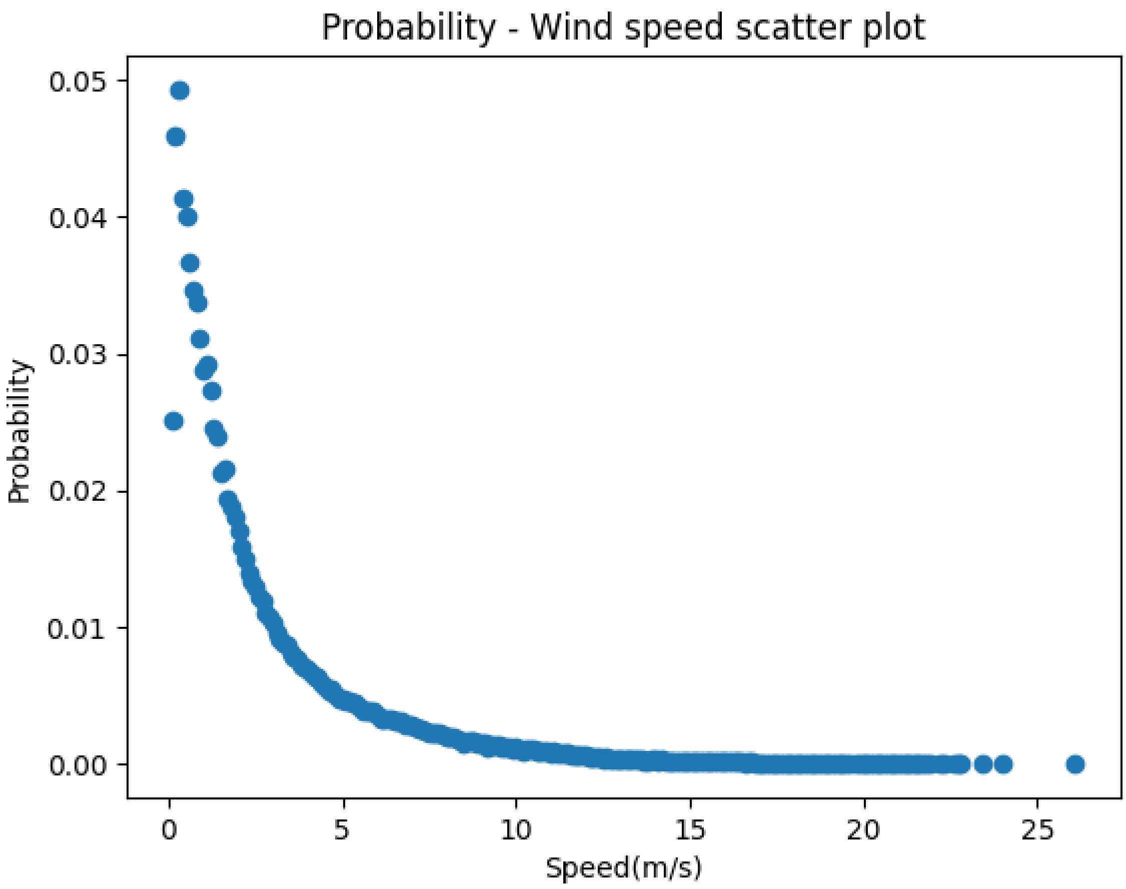
```
     Speed  Repetition Number  Probability
  0    0.7              13037     0.034585
  1    1.5               8023     0.021284
  2    2.0               6407     0.016997
  3    2.1               5995     0.015904
  4    1.0              10838     0.028751
...    ...                ...          ...
233   13.1                100     0.000265
234   14.6                 61     0.000162
235   14.1                 73     0.000194
236   15.1                 55     0.000146
237   15.6                 40     0.000106
```

Drawing a graph from experimental data

The discrete probability distribution in terms of speed is plotted below.

In [5]:
```
import matplotlib.pyplot as plt

plt.scatter(df["Speed"] , df["Probability"])
plt.title("Probability - Wind speed scatter plot")
plt.xlabel("Speed(m/s)")
plt.ylabel("Probability")
plt.show()
```



Determination of Weibull parameters

The scale parameter c and the shape parameter k in the Weibull function are determined.

In [6]:
```
import numpy as np

averageSpeed = np.mean(df["Speed"])
std = np.std(df["Speed"])

c = 1.12 * averageSpeed
k = (std / averageSpeed) ** (-1.086)

print(f"The scale parameter is {c} and the shape parameter is {k}")
```

```
The scale parameter is 12.76304424778761 and the shape parameter is 1.808
470123990429
```

Drawing the Weibull curve along with the graph of experimental data

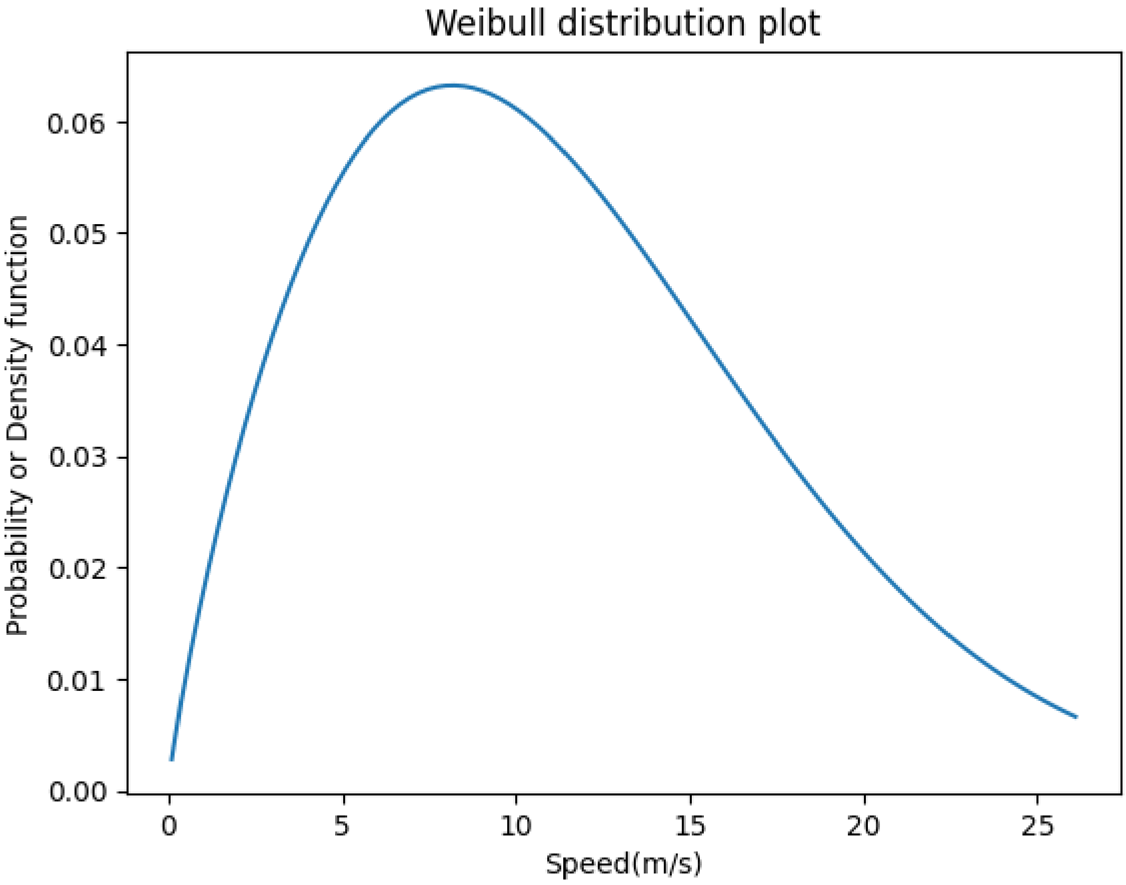
Finally the Weibull distribution and the discrete probability distribution are plotted.

It can be seen that the two do not coincide.

In [7]:

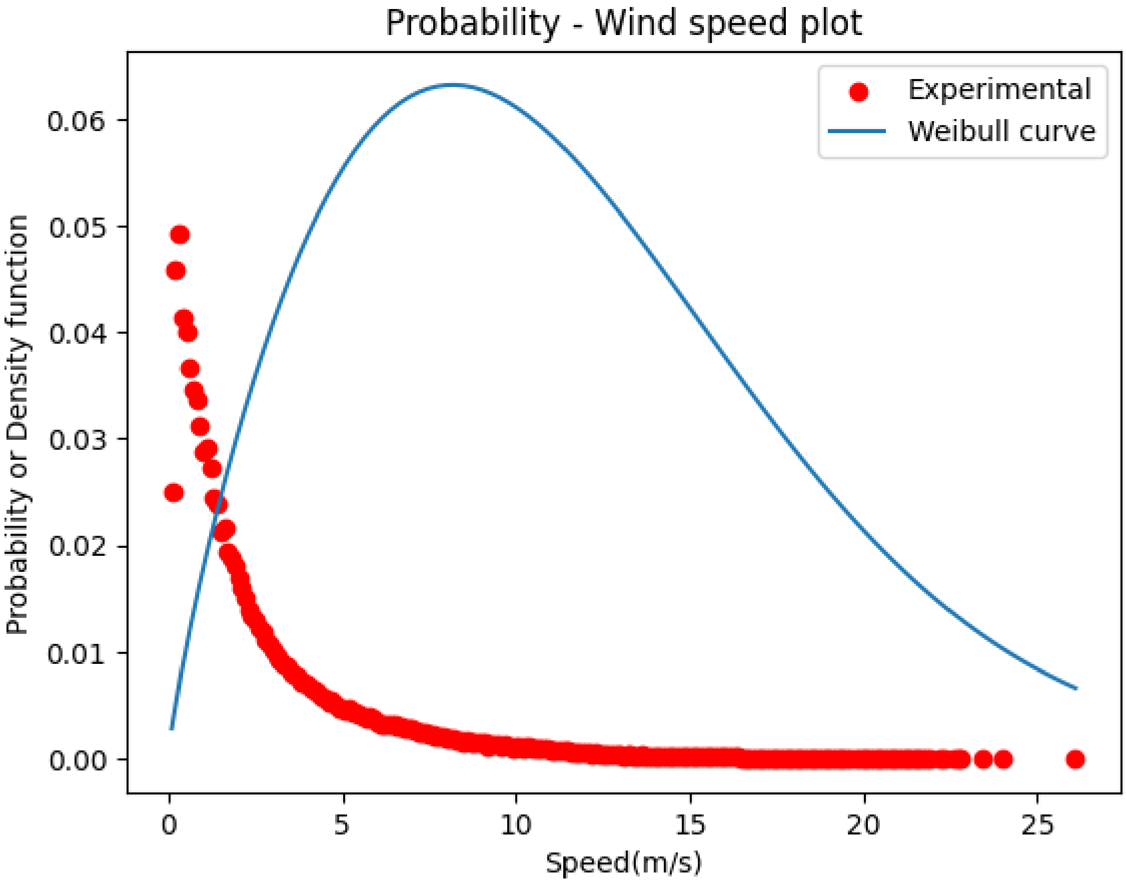
```
u = np.linspace(min(df["Speed"]) , max(df["Speed"]) , 100)
p = ((k / c) * (u / c) ** (k - 1)) * np.exp(-1 * (u / c) ** k)

plt.plot(u , p)
plt.title("Weibull distribution plot")
plt.xlabel("Speed(m/s)")
plt.ylabel("Probability or Density function")
plt.show()
```



In [8]:

```
plt.scatter(df["Speed"] , df["Probability"] , label = "Experimental" , colo
plt.title("Probability - Wind speed plot")
plt.plot(u , p , label = "Weibull curve")
plt.xlabel("Speed(m/s)")
plt.ylabel("Probability or Density function")
plt.legend()
plt.show()
```


Probability - Wind speed plot
Experimental
Weibull curve
Probability or Density function
Speed(m/s)