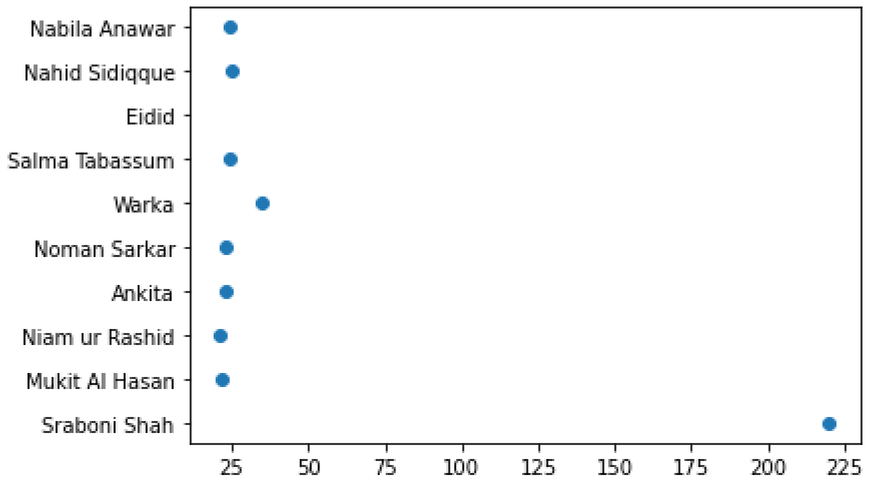
In [12]:
```
# reading dataset
import pandas as pd
df = pd.read_csv('student_info.csv')
df
```

Out[12]:

	Name	Age	CGPA	Degree	Job Probability	Unnamed: 5	Intern
0	Sraboni Shah	220.0	3.60	BSc	Medium	NaN	0
1	Mukit Al Hasan	22.0	3.70	BSc	Medium	NaN	0
2	Niam ur Rashid	21.0	NaN	BSc	Medium	NaN	0
3	Ankita	23.0	2.95	BSc	Low	NaN	0
4	Noman Sarkar	23.0	3.95	MSc	High	NaN	1
5	Warka	35.0	3.00	MSc	Medium	NaN	0
6	Salma Tabassum	24.0	3.96	PhD	High	NaN	1
7	Eidid	NaN	3.85	PhD	High	NaN	1
8	Nahid Sidiqque	25.0	4.00	PhD	High	NaN	1
9	Nabila Anawar	24.0	NaN	MSc	High	NaN	1

In [14]:
```
# showing outlier using scatter plot
import matplotlib.pyplot as plt
%matplotlib inline
plt.scatter(df['Age'], df['Name'])
```

Out[14]:
```
<matplotlib.collections.PathCollection at 0x1a5865dd3d0>
```



In [15]:
```
# calculating age by mode
age_mode = df['Age'].mode()[0]
age_mode
```

Out[15]:
```
23.0
```

In [16]:
```
# replacing outlier age value with age_mode value
df['Age'][0] = age_mode
df
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_16588\3369074777.py:2: SettingWithCo
pyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df['Age'][0] = age_mode
```

Out[16]:

	Name	Age	CGPA	Degree	Job Probability	Unnamed: 5	Intern
0	Sraboni Shah	23.0	3.60	BSc	Medium	NaN	0
1	Mukit Al Hasan	22.0	3.70	BSc	Medium	NaN	0
2	Niam ur Rashid	21.0	NaN	BSc	Medium	NaN	0
3	Ankita	23.0	2.95	BSc	Low	NaN	0
4	Noman Sarkar	23.0	3.95	MSc	High	NaN	1
5	Warka	35.0	3.00	MSc	Medium	NaN	0
6	Salma Tabassum	24.0	3.96	PhD	High	NaN	1
7	Eidid	NaN	3.85	PhD	High	NaN	1
8	Nahid Sidiqque	25.0	4.00	PhD	High	NaN	1
9	Nabila Anawar	24.0	NaN	MSc	High	NaN	1

In [17]:
```
# calculating total NaN value
df.isna().sum()
```

Out[17]:
```
Name                0
Age                 1
CGPA                2
Degree              0
Job Probability     0
Unnamed: 5         10
Intern              0
dtype: int64
```

In [18]:
```
# Filling the NaN value with age_mode value
df["Age"].fillna(age_mode, inplace=True)
df
```

Out[18]:

	Name	Age	CGPA	Degree	Job Probability	Unnamed: 5	Intern
0	Sraboni Shah	23.0	3.60	BSc	Medium	NaN	0
1	Mukit Al Hasan	22.0	3.70	BSc	Medium	NaN	0
2	Niam ur Rashid	21.0	NaN	BSc	Medium	NaN	0
3	Ankita	23.0	2.95	BSc	Low	NaN	0
4	Noman Sarkar	23.0	3.95	MSc	High	NaN	1
5	Warka	35.0	3.00	MSc	Medium	NaN	0
6	Salma Tabassum	24.0	3.96	PhD	High	NaN	1
7	Eidid	23.0	3.85	PhD	High	NaN	1
8	Nahid Sidiqque	25.0	4.00	PhD	High	NaN	1
9	Nabila Anawar	24.0	NaN	MSc	High	NaN	1

In [20]:
```
# calculating mean value of CGPA column
cg = df['CGPA'].mean()
cg
```

Out[20]:
```
3.62625
```

In [21]:
```
# Filling the NaN value with cg value
df['CGPA'].fillna(cg, inplace=True)
df
```

Out[21]:

	Name	Age	CGPA	Degree	Job Probability	Unnamed: 5	Intern
0	Sraboni Shah	23.0	3.60000	BSc	Medium	NaN	0
1	Mukit Al Hasan	22.0	3.70000	BSc	Medium	NaN	0
2	Niam ur Rashid	21.0	3.62625	BSc	Medium	NaN	0
3	Ankita	23.0	2.95000	BSc	Low	NaN	0
4	Noman Sarkar	23.0	3.95000	MSc	High	NaN	1
5	Warka	35.0	3.00000	MSc	Medium	NaN	0
6	Salma Tabassum	24.0	3.96000	PhD	High	NaN	1
7	Eidid	23.0	3.85000	PhD	High	NaN	1
8	Nahid Sidiqque	25.0	4.00000	PhD	High	NaN	1
9	Nabila Anawar	24.0	3.62625	MSc	High	NaN	1

In [22]:
```
# calculating total NaN value
df.isna().sum()
```

Out[22]:
```
Name                0
Age                 0
CGPA                0
Degree              0
Job Probability     0
Unnamed: 5         10
Intern              0
dtype: int64
```

In [23]:
```
# Removing the 'Unnamed: 5' column
df2 = df.drop(['Unnamed: 5'], axis=1)
df2
```

Out[23]:

	Name	Age	CGPA	Degree	Job Probability	Intern
0	Sraboni Shah	23.0	3.60000	BSc	Medium	0
1	Mukit Al Hasan	22.0	3.70000	BSc	Medium	0
2	Niam ur Rashid	21.0	3.62625	BSc	Medium	0
3	Ankita	23.0	2.95000	BSc	Low	0
4	Noman Sarkar	23.0	3.95000	MSc	High	1
5	Warka	35.0	3.00000	MSc	Medium	0
6	Salma Tabassum	24.0	3.96000	PhD	High	1
7	Eidid	23.0	3.85000	PhD	High	1
8	Nahid Sidiqque	25.0	4.00000	PhD	High	1
9	Nabila Anawar	24.0	3.62625	MSc	High	1

In [35]:
```
# converting the string value with number using LabelEncoder
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df2['Name'] = le.fit_transform(df2['Name'])
df2['Degree'] = le.fit_transform(df2['Degree'])
df2['Job Probability'] = le.fit_transform(df2['Job Probability'])
```

In [36]:
```
# removing the intern column
x = df2.drop(['Intern'], axis=1)
x
```

Out[36]:

	Name	Age	CGPA	Degree	Job Probability
0	8	23.0	3.60000	0	2
1	2	22.0	3.70000	0	2
2	5	21.0	3.62625	0	2
3	0	23.0	2.95000	0	1
4	6	23.0	3.95000	1	0
5	9	35.0	3.00000	1	2
6	7	24.0	3.96000	2	0
7	1	23.0	3.85000	2	0
8	4	25.0	4.00000	2	0
9	3	24.0	3.62625	1	0

In [37]:
```
# taking the Intern column
y = df2['Intern']
y
```

Out[37]:
```
0    0
1    0
2    0
3    0
4    1
5    0
6    1
7    1
8    1
9    1
Name: Intern, dtype: int64
```

In [48]:
```
# dividing the dataset for training & testing
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_
```

In [49]:
```
x_train
```

Out[49]:

	Name	Age	CGPA	Degree	Job Probability
5	9	35.0	3.00000	1	2
0	8	23.0	3.60000	0	2
7	1	23.0	3.85000	2	0
2	5	21.0	3.62625	0	2
9	3	24.0	3.62625	1	0
4	6	23.0	3.95000	1	0
3	0	23.0	2.95000	0	1
6	7	24.0	3.96000	2	0

In [50]:
```
y_train
```

Out[50]:
```
5    0
0    0
7    1
2    0
9    1
4    1
3    0
6    1
Name: Intern, dtype: int64
```

In [51]:
```
# using LogisticRegression model
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

In [52]:
```
# training the model
model.fit(x_train, y_train)
```

Out[52]:
```
LogisticRegression()
```

In [54]:
```
# predicting by test data
y_pred = model.predict(x_test)
y_pred
```

Out[54]:
```
array([1, 0], dtype=int64)
```

In [55]:
```
# calculating the confusion matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[55]:
```
array([[1, 0],
       [0, 1]], dtype=int64)
```

In [56]:
```
# calculating the accuracy of our model
accuracy_score(y_test, y_pred)
```

Out[56]:
```
1.0
```

In [57]:
```
# predicting
model.predict([[8,23.0,3.60000,0,2]])
```

```
C:\Users\USER\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X d
oes not have valid feature names, but LogisticRegression was fitted with featur
e names
  warnings.warn(
```

Out[57]:
```
array([0], dtype=int64)
```