In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
haberman = pd.read_csv("haberman (1).csv")
```

In [3]:
```
haberman
```

Out[3]:

	age	year	nodes	status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1
...	...	...	...	...
301	75	62	1	1
302	76	67	0	1
303	77	65	3	1
304	78	65	1	2
305	83	58	2	2

306 rows × 4 columns

In [4]:
```
#this dataset consist of total list of 306 patients that classified on basis of their age,year of operation ,n
#status has numerical value of either 1 (that represents patients survive 5 years or more after operation) and
```

In [5]:
```
haberman["status",.value_counts()
```

Out[5]:
```
1    225
2     81
Name: status, dtype: int64
```

In [6]:
```
# so in dataset we have total of 225 number of patients with survival years or more and 81 no of patients with
```

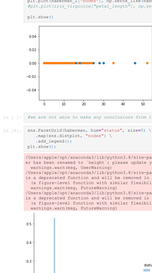
In [7]:
```
haberm = haberman[["status"]==1"]
haberm
```

Out[8]:
```
0    False
1    False
2    False
3    False
4    False
     ...
301  False
302  False
303  False
304  False
305  False
Name: Length: 306, dtype: bool
```

In [9]:
```
#for status 1 represents patients survivng 5 years or more and status 2 represents patients surviving 5 years
# So out of 306 patients we can say 225 patients survived 5 years or more and 81 patients survived 5 years or
```

In [10]:
```
sns.FacetGrid(haberman, hue="status", size=6) \
    .map(plt.scatter, "age", "year") \
    .add_legend()
plt.show()
```

```
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:316: UserWarning: The `size` parameter
er has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```



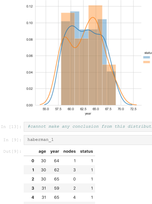
In [11]:
```
sns.set_style("whitegrid")
sns.pairplot(haberman, hue="status", size=3);
plt.show()
```

```
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:912: UserWarning: The `size` parame
ter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

In [12]:
```
# from above pairplots we find age and nodes to most suitable to analyze the data further as it has least over
```

In [13]:
```
import numpy as np
haberman_1 = haberman.loc[haberman["status"] == 1]
haberman_2 = haberman.loc[haberman["status"] == 2]
plt.plot(haberman_1["nodes"], np.zeros_like(haberman_1["nodes"]), 'o')
plt.plot(haberman_2["nodes"], np.zeros_like(haberman_2["nodes"]), 'o')
#plt.plot(iris_virginica["petal_length"], np.zeros_like(iris_virginica["petal_length"]), 'o')
```

plt.show()



In [14]:
```
#we are not able to make any conclusion from 1-D scatter plot as most of points are overlapping each other
```

In [15]:
```
sns.FacetGrid(haberman, hue="status", size=6) \
    .map(sns.distplot, "nodes") \
    .add_legend()
plt.show();
```

```
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:316: UserWarning: The `size` parame
er has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

In [16]:
```
#from above plot we can say patients with less number of nodes tend to survive more
# if nodes<= 2 patients survival is more
```

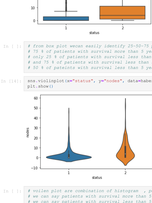
In [17]:
```
sns.FacetGrid(haberman, hue="status", size=6) \
    .map(sns.distplot, "age") \
    .add_legend();
plt.show();
```

```
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:316: UserWarning: The `size` parame
er has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

In [18]:
```
# from this plot though there is no clear saturation of age still we can make a rough prediction as
# if age >=75 patients survive less than 5 years
# almost age <=35 patients usually survive less than 5 years
```

In [19]:
```
sns.FacetGrid(haberman, hue="status", size=6) \
    .map(sns.distplot, "year") \
    .add_legend();
plt.show();
```

```
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:316: UserWarning: The `size` parame
er has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/apple/opt/anaconda3/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot
is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [20]:
```
#cannot make any conclusion from this distribution plot as it is overlapping at every point
```

In [21]:
```
haberman_1
```

Out[21]:

	age	year	nodes	status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1
...	...	...	...	...
298	73	68	0	1
300	74	63	0	1
301	75	62	1	1
302	76	67	0	1
303	77	65	3	1

225 rows × 4 columns

In [22]:
```
counts, bin_edges = np.histogram(haberman_1["age"], bins=20,
                                 density = True)
print(counts)
pdf = counts/(sum(counts))

print(pdf);
print(bin_edges)

#compute CDF
cdf = np.cumsum(pdf)
plt.plot(bin_edges[1:],pdf)
plt.plot(bin_edges[1:], cdf)


plt.show();
```

```
[0.01134752 0.02268504 0.02647754 0.01985816 0.03403816 0.03403816
 0.01985816 0.02268504 0.01134752 0.01134752 0.01134752 0.
 0.00567376 0.01134752 0.00567376 0.00567376 0.00567376 0.00567376
 0.00567376 0.00567376]
[0.01134752 0.02268504 0.02647754 0.01985816 0.03403816 0.03403816
 0.01985816 0.02268504 0.01134752 0.01134752 0.01134752 0.
 0.00567376 0.01134752 0.00567376 0.00567376 0.00567376 0.00567376
 0.00567376 0.00567376]
[30. 34.7 39.4 44.1 48.8 53.5 58.2 62.9 67.6 72.3 77. ]
```


In [23]:
```
# we can conclude from cdf that atleast 79% of patients with age <=60 tend to live more than five years
```

In [24]:
```
print("Means:")
print(np.mean(haberman_1["nodes"]))

print(np.mean(haberman_2["nodes"]))

print("\nStd-dev:");
print(np.std(haberman_1["nodes"]))
print(np.std(haberman_2["nodes"]))
```

```
Means:
2.7911111111111113
5.456790123456789

Std-dev:
5.857258449412131
7.228702078782632
```

In [25]:
```
print("\nMedians:")
print(np.median(haberman_1["nodes"]))
#median with an outlier
print(np.median(np.append(haberman_1["nodes"],50)));
print(np.median(haberman_2["nodes"]))
```

```
Medians:
0.0
0.0
4.0
```

```
# average node size of patients with survival more than 5 years is 0 .
# average node size of patients with survival less than 5 years is 4.
```

In [26]:
```
print("\nQuantiles:")
print(np.percentile(haberman_1["nodes"],np.arange(0, 100, 25)))
print(np.percentile(haberman_2["nodes"],np.arange(0, 100, 25)))

print("\n90th Percentiles:")
print(np.percentile(haberman_1["nodes"],90))
print(np.percentile(haberman_2["nodes"],90))
```

```
Quantiles:
[0. 0. 0. 3.]
[ 0.  1.  4. 11.]

90th Percentiles:
7.0
20.0
```

```
# with quantiles we can say 90 % of patients having survival more than 5 years have node 0 .
#atleast 75% of patients having survival more than 5 years have node less than 3
# 90 % of patients with survival less than 5 years have node <= 8.
# with quantiles we can also say 90 % of patients with survival less than 5 years have node <= and 75% of pat
# 90 % of patients with survival less than 5 years have node <= 20
```

In [27]:
```
sns.boxplot(x="status",y="nodes", data=haberman)
plt.show()
```


In [28]:
```
# from box plot mean easily identify 25-50 75 percentile of values .
# 75 % of patients with survival less than 5 years have node size <=11
# only 25 % of patients with survival less than 5 years have node size >=1
# and 75 % of patients with survival less than 5 years have node size <=11
# 25 % of patients with survival less than 5 years have node size >=0
```

In [29]:
```
sns.violinplot(x="status", y="nodes", data=haberman, size=8)
plt.show()
```


In [30]:
```
# violien plot are combination of histogram , pdf and boxplot
# we can say patients with survival more than 5 years have node range 0<=node <=3.
# we can say patients with survival less than 5 years have node range 0<=node <=20.
```