In [1]:

```
#importing packages
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import warnings

warnings.filterwarnings("ignore")

#Importing the csv data
data = pd.read_csv(r"C:\Users\arukh\Desktop\Applied AI\Assignment 2\haberman.csv")
data.head()
```

Out[1]:

	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

checking the data in csv

copied from kaggle Attribute Information:

Age of patient at time of operation (numerical) Patient's year of operation (year - 1900, numerical) Number of positive axillary nodes detected (numerical) Survival status (class attribute) 1 = the patient survived 5 years or longer 2 = the patient died within 5 year

nodes: lymp nodes mostly present under pits, necks, groins are affected due to metastasis of cancer. Knowledge from Biotech.

In [2]:

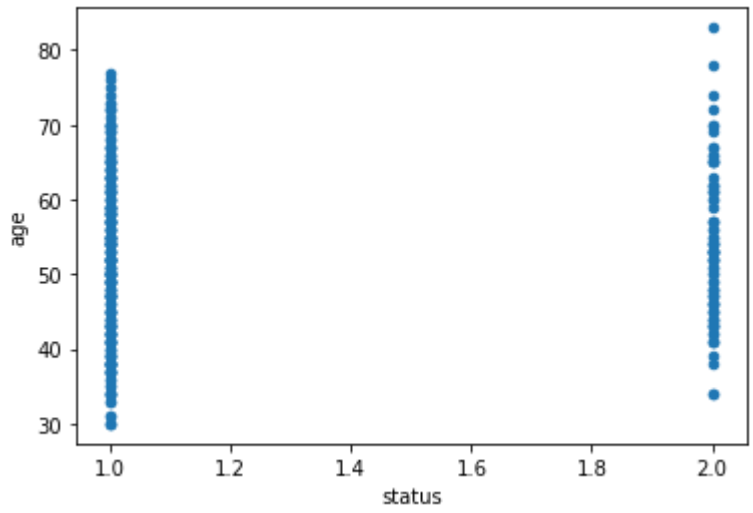
```
print(data.columns) #checking the name of the columns(all are object data type)
print(data.shape)  #checking the number of columns and rows in the given data
#print(data['status'].value_counts)
print(data.status.unique()) #checking the unique values of the column 'status'. works efficiently than function value_counts
print(data.nodes.unique())  #same as above for column/feature 'nodes'. in both cases return type seems to be list.
print(data.info())
```

```
Index(['age', 'year', 'nodes', 'status'], dtype='object')
(306, 4)
[1 2]
[ 1  3  0  2  4 10  9 30  7 13  6 15 21 11  5 23  8 20 52 14 19 16 12 24
 46 18 22 35 17 25 28]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 306 entries, 0 to 305
Data columns (total 4 columns):
age       306 non-null int64
year      306 non-null int64
nodes     306 non-null int64
status    306 non-null int64
dtypes: int64(4)
memory usage: 9.7 KB
None
```

plotting various graphs

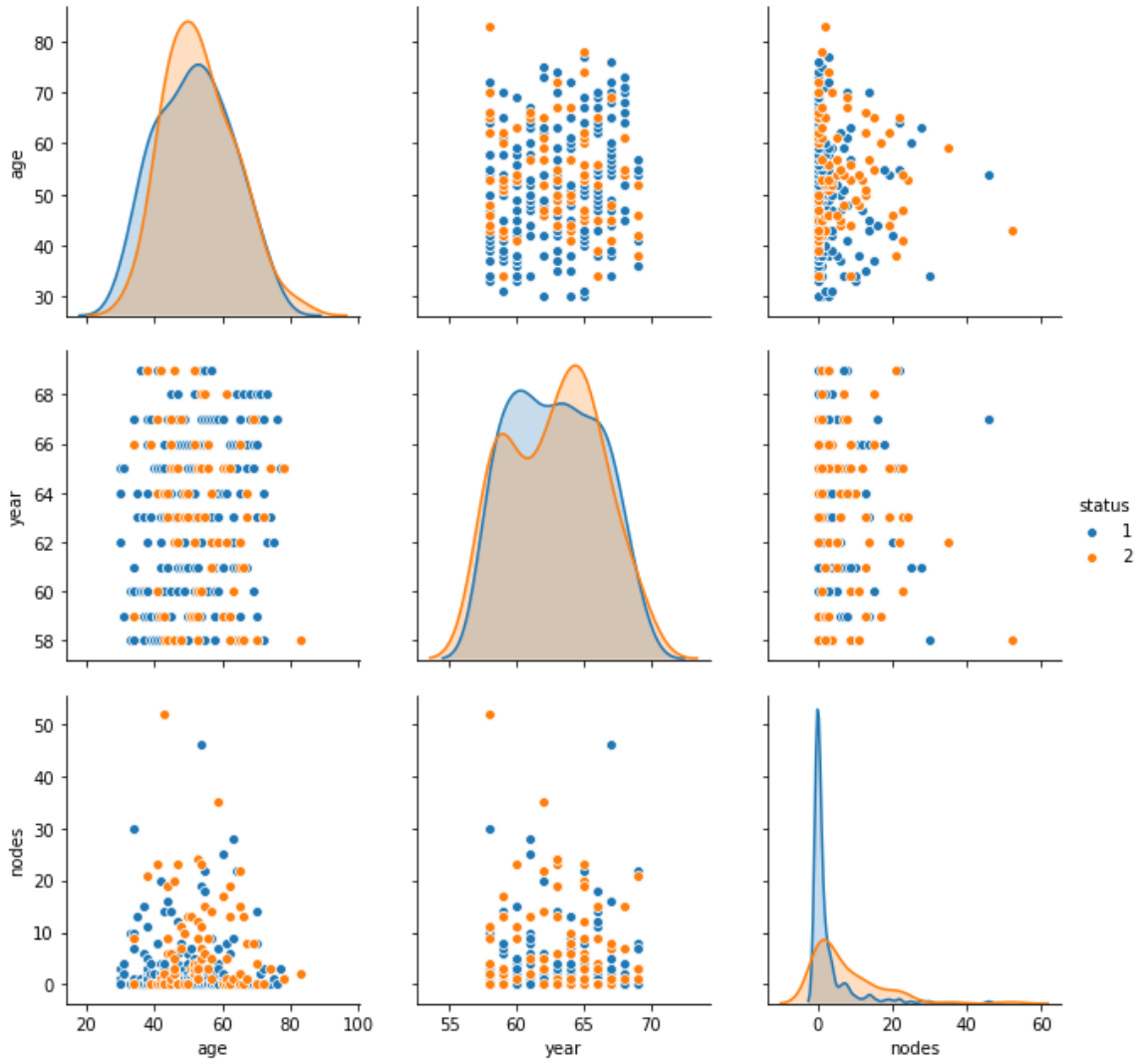
In [3]:

```
# 2-D scatter plot
data.plot(kind='scatter', x='status', y='age') ;
plt.show()
#observation: not clear
```



In [4]:

```
sns.pairplot(data, hue="status", vars= ["age", "year", "nodes"],size=3);
warnings.filterwarnings("ignore")
plt.show()
```



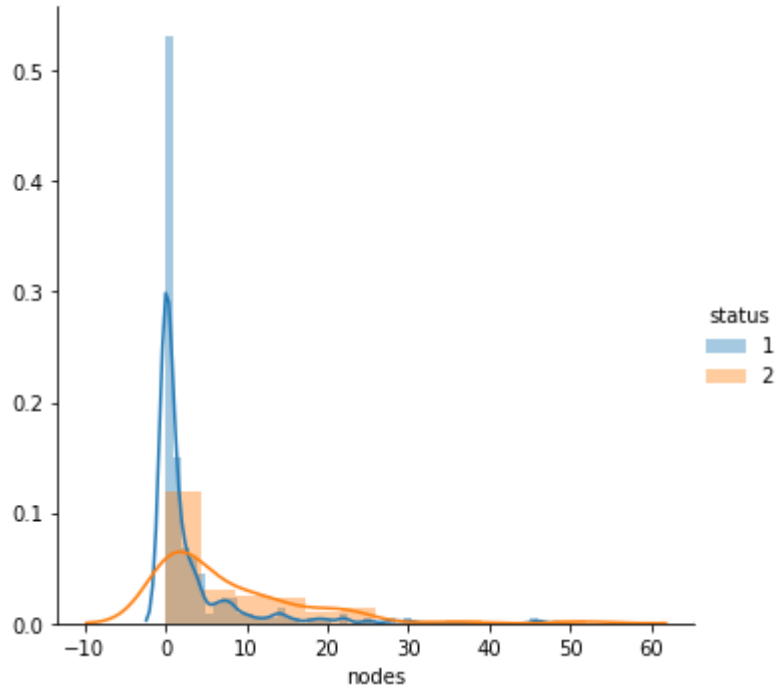
Summary of pair plot

1) graph between age- year: status of the patient is not clear as we can not determine a threshold which could separate the status of patients. All the data points are very close to each other. 2) graph between age-nodes: same as point 1. 3) graph between age - status : below a certain age (33) the patient survives. if age<=33, patient survives 4) graph between year - nodes : the data points are much far away as compared to point 1 and 2. still its not clear.

Uni-variate Analysis

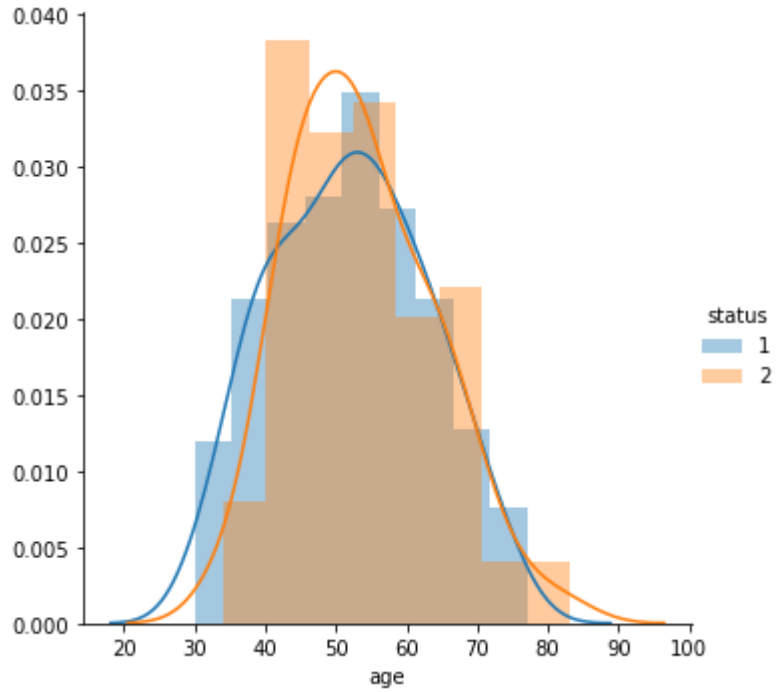
In [5]:

```
sns.FacetGrid(data, hue="status", size=5) \
   .map(sns.distplot, "nodes") \
   .add_legend();
plt.show();
```



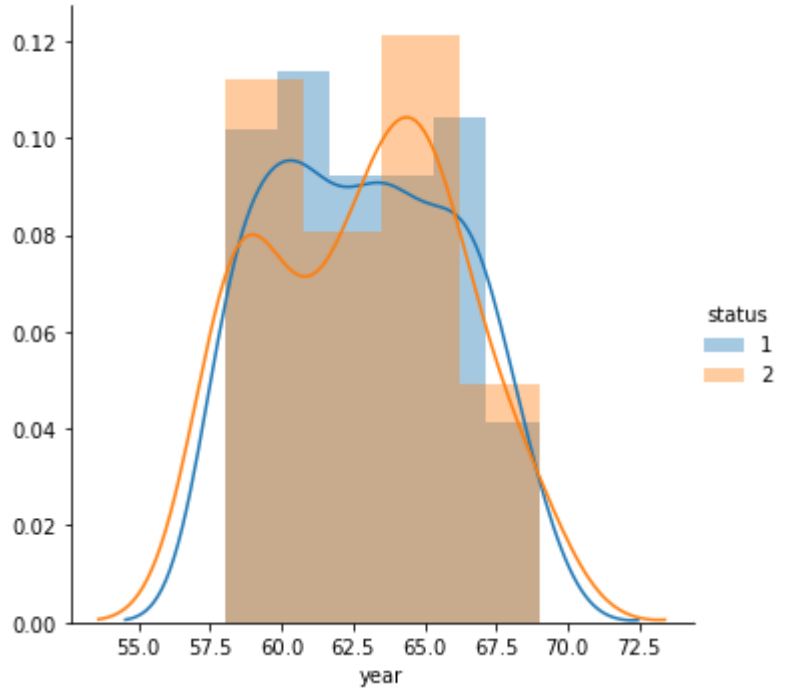
In [6]:

```
sns.FacetGrid(data, hue="status", size=5) \
   .map(sns.distplot, "age") \
   .add_legend();
plt.show();
```



In [7]:

```
sns.FacetGrid(data, hue="status", size=5) \
   .map(sns.distplot, "year") \
   .add_legend();
plt.show();
```



Summary:

This analysis against all the features doesnt give much information as the PDF overlaps a lot in all cases.

In [8]:

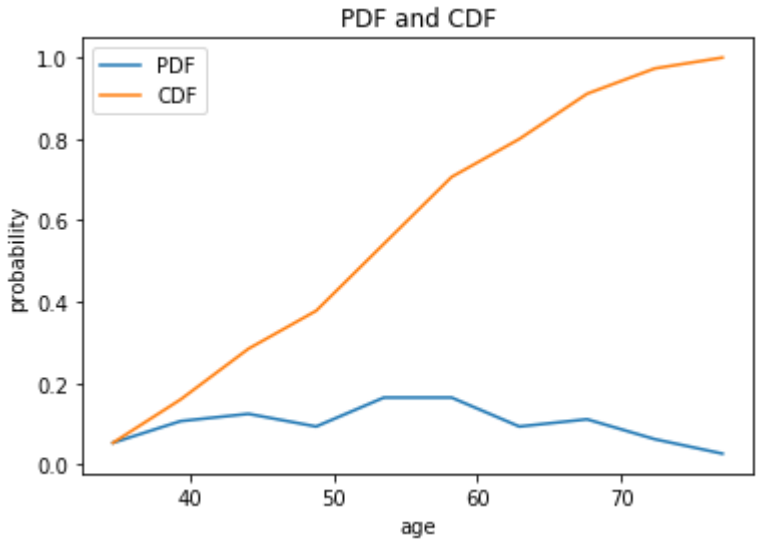
```
data_survived = data.loc[data["status"] == 1];
data_notsurvived = data.loc[data["status"] == 2];


count, edges =  np.histogram(data_survived['age'], bins=10, density = True)
#print(count)
pdf = count/(sum(count))  #calculating probability density function
print(pdf);
print(edges)
cdf = np.cumsum(pdf)      #calculating CDF (cumulative distribution function)
plt.plot(edges[1:],pdf);
plt.plot(edges[1:], cdf)
plt.title("PDF and CDF")  #(referred: https://pythonprogramming.net/legends-titles-labels-matplotlib-tutorial/)
plt.xlabel('age')
plt.ylabel('probability')
plt.legend(['PDF', 'CDF'])
```

```
[0.05333333 0.10666667 0.12444444 0.09333333 0.16444444 0.16444444
 0.09333333 0.11111111 0.06222222 0.02666667]
[30.  34.7 39.4 44.1 48.8 53.5 58.2 62.9 67.6 72.3 77. ]
```

Out[8]:

```
<matplotlib.legend.Legend at 0x242a7714f08>
```



summary

Individual plot shows that at age of 44.1 there almost 11% probability of survival. and so on for the rest of the data points. But when we include the same for status = 2 (patients did not survive) the pdfs and cdfs overlaps each other to a great extent.

In [ ]:

In [9]:

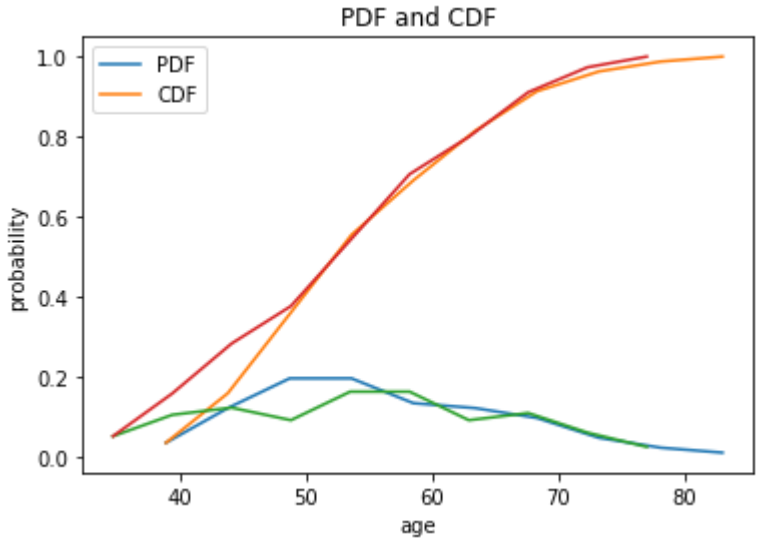
```
count, edges =  np.histogram(data_notsurvived['age'], bins=10, density = True)
pdf = count/(sum(count))  #calculating probability density function
print(pdf);
print(edges)
cdf = np.cumsum(pdf)      #calculating CDF (cumulative distribution function)
plt.plot(edges[1:],pdf);
plt.plot(edges[1:], cdf)

count, edges =  np.histogram(data_survived['age'], bins=10, density = True)
pdf = count/(sum(count))  #calculating probability density function
print(pdf);
print(edges)
cdf = np.cumsum(pdf)      #calculating CDF (cumulative distribution function)
plt.plot(edges[1:],pdf);
plt.plot(edges[1:], cdf)
plt.title("PDF and CDF")  #(referred: https://pythonprogramming.net/legends-titles-labels-matplotlib-tutorial/)
plt.xlabel('age')
plt.ylabel('probability')
plt.legend(['PDF', 'CDF'])
```

```
[0.03703704 0.12345679 0.19753086 0.19753086 0.13580247 0.12345679
 0.09876543 0.04938272 0.02469136 0.01234568]
[34.  38.9 43.8 48.7 53.6 58.5 63.4 68.3 73.2 78.1 83. ]
[0.05333333 0.10666667 0.12444444 0.09333333 0.16444444 0.16444444
 0.09333333 0.11111111 0.06222222 0.02666667]
[30.  34.7 39.4 44.1 48.8 53.5 58.2 62.9 67.6 72.3 77. ]
```

Out[9]:

```
<matplotlib.legend.Legend at 0x242a7982a08>
```



In [10]:

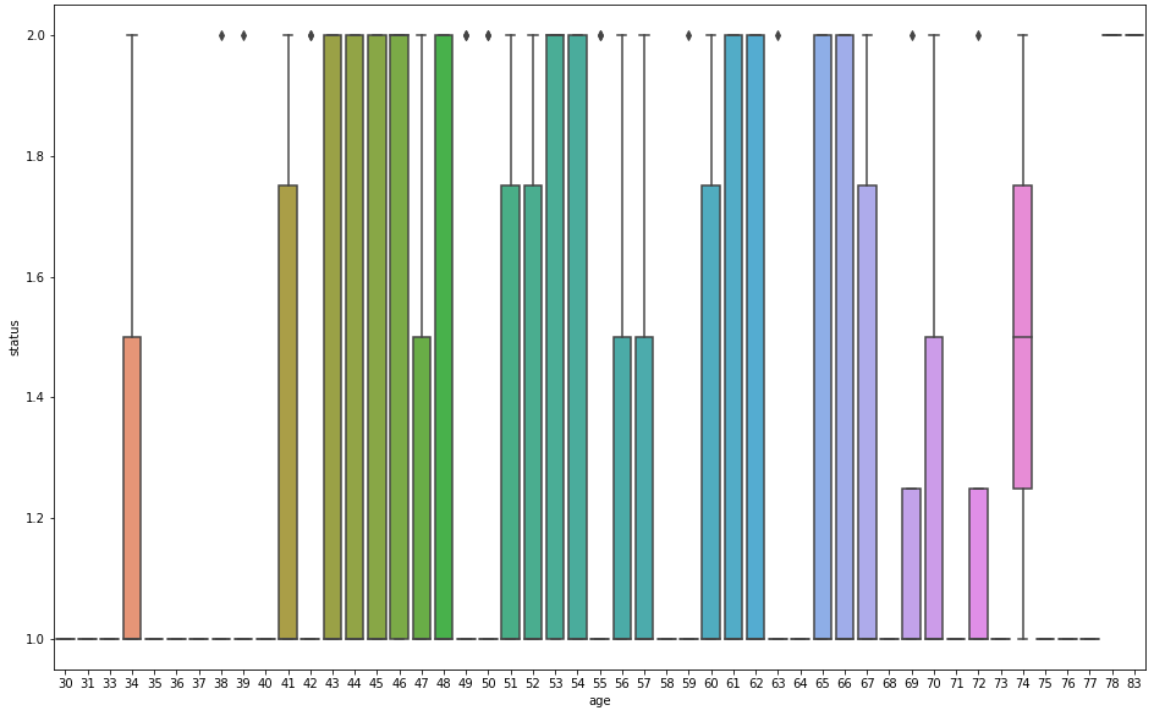
```
#Calculating mean, standard deviation, median, quantiles
print("Means:")
print(np.mean(data_survived["age"]))
print(np.mean(data_notsurvived["age"]))
print("***********")
print("Standard Deviation:")
print(np.std(data_survived["age"]))
print(np.std(data_notsurvived["age"]))
print("***********")
print("Median:")
print(np.median(data_survived["age"]))
print(np.median(data_notsurvived["age"]))
print("***********")
print("Median:")
print(np.percentile(data_survived["age"],np.arange( 0,100,25)))
print(np.percentile(data_notsurvived["age"],np.arange( 0,100,25)))
```

```
Means:
52.01777777777778
53.67901234567901
***********
Standard Deviation:
10.98765547510051
10.10418219303131
***********
Median:
52.0
53.0
***********
Median:
[30. 43. 52. 60.]
[34. 46. 53. 61.]
```

BoxPlot and Violin plot

In [11]:

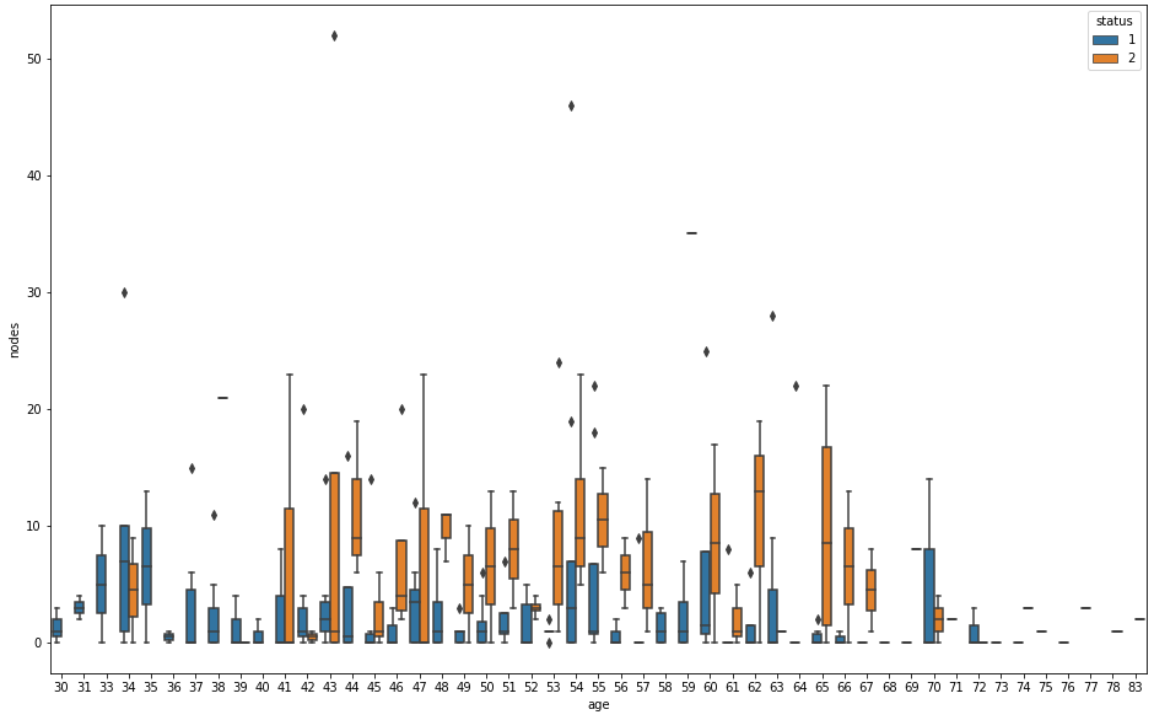
```
plt.figure(figsize=(16, 10)) #figure size reference(https://www.drawingfromdata.com/setting-figure-size-using-seaborn-and-matplotlib)
sns.boxplot(x='age',y='status', data=data)
plt.show()
```



In [12]:

```
plt.figure(figsize=(16, 10))
sns.boxplot(x='age',y='nodes', data=data, hue = 'status')

plt.show()
```



Summary:

in graph between age-nodes with status as hue : it is possible to define a threshold. but there would be some mistake or inaccuracy. if nodes<= 4, patient tends to live if age>=40 , patient tends to survive.