Haberman Dataset

Haberman Dataset: [https://www.kaggle.com/gilsousa/habermans-survival-data-set/version/1 (https://www.kaggle.com/gilsousa/habermans-survival-data-set/version/1)]

1. Description : The dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer.
2. Number of Instances: 306
3. Number of Attributes: 4 (including the class attribute)
4. 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

Importing Libraries and the dataset ¶

In [2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

In [3]:
```
haber = pd.read_csv('haberman.csv')
```

Understanding dataset

In [4]:
```
#prints the number of rows and number of columns
print (haber.shape)
```

```
(306, 4)
```

In [5]:
```
#prints column names of the dataset
print (haber.columns)
```

```
Index(['age', 'year', 'nodes', 'status'], dtype='object')
```

In [6]:
```
# brief info of the dataset and dataypes
print (haber.info())
```

```
<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: 9.6 KB
None
```

Observations

1. No mising values in the dataset
2. All the features have integer datatype
3. Datatype of class variable i.e. 'status' needs to be converted to Categorical
4. As a next step, we will map value 1 to ‘yes’ which means the patient has survived 5 years or longer. And the value 2 to ‘no’ which means the patient died within 5 years.

In [7]:
```
haber['status'] = np.where(haber['status'] == 1,'yes','no')
haber['status'] = haber['status'].astype('category')
```

In [8]:
```
haber.head()
```

Out[8]:

	age	year	nodes	status
0	30	64	1	yes
1	30	62	3	yes
2	30	65	0	yes
3	31	59	2	yes
4	31	65	4	yes

In [9]:
```
#gives each count of the status type
haber['status'].value_counts()
```

Out[9]:
```
yes    225
no      81
Name: status, dtype: int64
```

Observations

1. Out of 306 patients, 225 patients survived and 81 did not.
2. The dataset is imbalanced.

In [10]:
```
#haber_yes dataframe stores all the records where status is yes
#haber_no dataframe stores all the records where status is no

haber_yes = haber[haber['status'] == 'yes']
haber_no = haber[haber['status'] == 'no']

##describes the dataset
print  ("Overall")
print (haber.describe())
print ('\n')
print ("Status : Yes")
print (haber_yes.describe())
print ('\n')
print ("Status : No")
print (haber_no.describe())
```

```
Overall
              age        year       nodes
count  306.000000  306.000000  306.000000
mean    52.457516   62.852941    4.026144
std     10.803452    3.249405    7.189654
min     30.000000   58.000000    0.000000
25%     44.000000   60.000000    0.000000
50%     52.000000   63.000000    1.000000
75%     60.750000   65.750000    4.000000
max     83.000000   69.000000   52.000000


Status : Yes
              age        year       nodes
count  225.000000  225.000000  225.000000
mean    52.017778   62.862222    2.791111
std     11.012154    3.222915    5.870318
min     30.000000   58.000000    0.000000
25%     43.000000   60.000000    0.000000
50%     52.000000   63.000000    0.000000
75%     60.000000   66.000000    3.000000
max     77.000000   69.000000   46.000000


Status : No
             age       year      nodes
count  81.000000  81.000000  81.000000
mean   53.679012  62.827160   7.456790
std    10.167137   3.342118   9.185654
min    34.000000  58.000000   0.000000
25%    46.000000  59.000000   1.000000
50%    53.000000  63.000000   4.000000
75%    61.000000  65.000000  11.000000
max    83.000000  69.000000  52.000000
```

In [11]:
```
print('\n90th percentile:')
print ("Status : Yes")
print( np.percentile(haber_yes['nodes'],90))
print ("Status : No")
print(np.percentile(haber_no['nodes'],90))
```

```
90th percentile:
Status : Yes
8.0
Status : No
20.0
```

Observations

1. Age of patients vary from 30 to 83 with median of 52
2. The mean age and the year in which the patients got operated are almost similar of both the classes
3. In class with status 'yes', nearly 75% of the population have less than 3 positive axillary nodes and nearly 50% of the patients have no positive axillary nodes. 90% of the population have less than 8 positive axillary nodes
4. In class with status 'no' ,50% of the population have less than or equal to 4 positive axillary nodes and 75% of the population have less than 11 positive axillary nodes. 90% of the population have less than 20 positive axillary nodes
5. The mean of the nodes of both the classes differs by 5 units approximately.
6. The nodes of patients who survived are less when compared to patients who did not survive.

Univariate Analysis

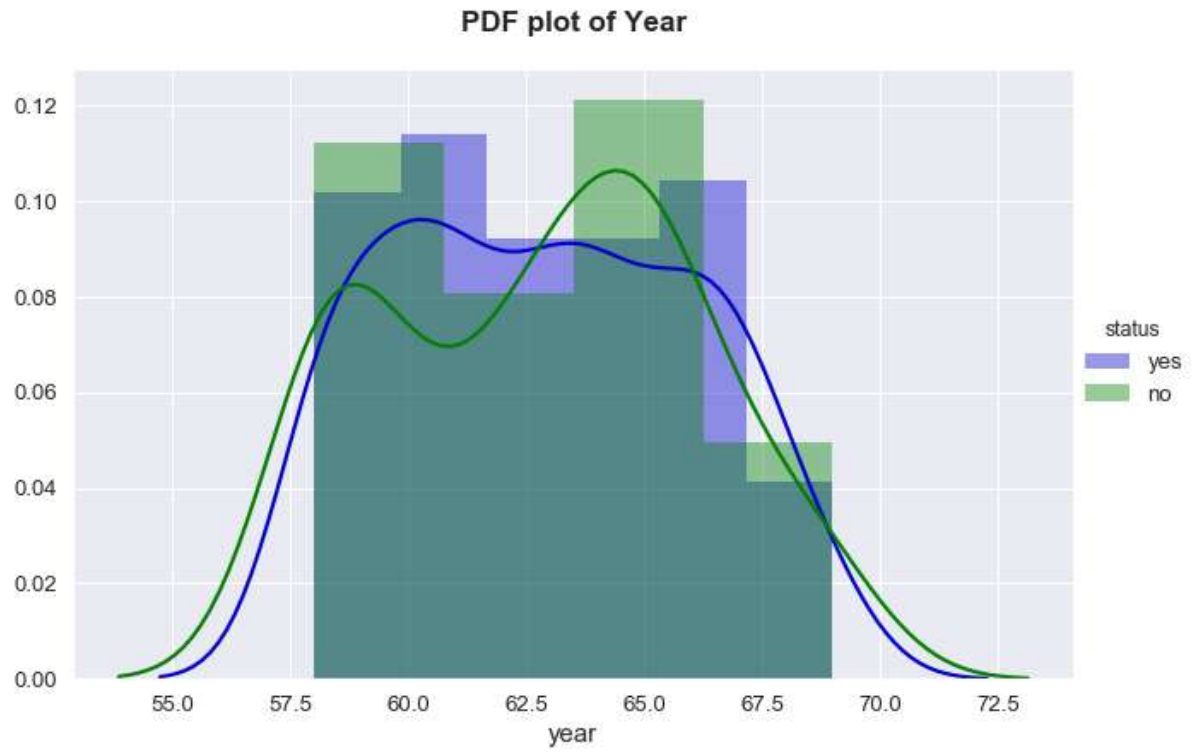
Probability Density Function (PDF)

In [94]:
```
## Univariate Analysis - To understand each feature

sns.set_style("darkgrid")
pal = ['mediumblue','green']
sns.set(font_scale=1.2)

g = sns.FacetGrid(haber,hue = 'status',palette=pal,height=6,aspect=1.5, hue_or
der=['yes','no'])
g.map(sns.distplot,"age",hist_kws = {'edgecolor': 'none'}, kde_kws = {'linewid
th': 2.2} ).add_legend()
plt.title('\nPDF plot of Age\n',fontsize = 17,fontweight="bold")
plt.xlabel ('age',fontsize=15)
plt.show()
```

Observations

1. Major overlapping is observed
2. We can vaguely tell that people whose age is in the range between 30-40 are more likely to survive and 40-58 are less likely to survive.
3. People whose age is in the range between 60-75 have equal chances of surviving and not suviving.
4. However, considering the major overlapping we cannot decide the survival of a patient just by age.

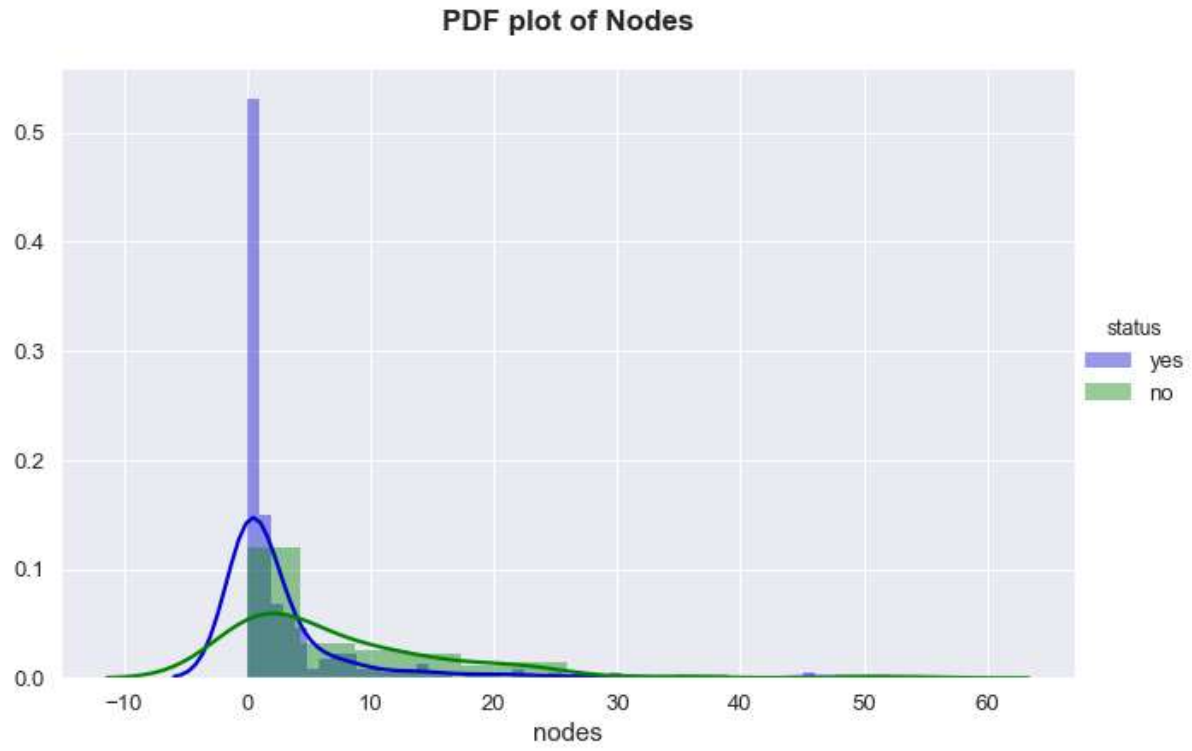
In [93]:
```
sns.set(font_scale=1.2)
g = sns.FacetGrid(haber,hue = 'status',palette=pal,height=6,aspect=1.5, hue_or
der=['yes','no'])
g.map(sns.distplot,"year",hist_kws = {'edgecolor': 'none'}, kde_kws = {'linewi
dth': 2.2} ).add_legend()
plt.xlabel ('year',fontsize=15)
plt.title('\nPDF plot of Year\n',fontsize = 17,fontweight="bold")
plt.show()
```



Observations

1. Major overlapping is observed
2. This graph only tells about number of successful and unsucessful surgeries.
3. This cannot be a parameter to classify survival status.

In [92]:
```
sns.set(font_scale=1.2)
g = sns.FacetGrid(haber,hue = 'status',palette=pal,height=6,aspect=1.5, hue_or
der=['yes','no'])
g.map(sns.distplot,"nodes",hist_kws = {'edgecolor': 'none'}, kde_kws = {'linew
idth': 2.2} ).add_legend()
plt.title('\nPDF plot of Nodes\n',fontsize = 17,fontweight="bold")
plt.xlabel ('nodes',fontsize=15)
plt.show()
```



Observations

1. It can be observed that people who survived have less positive axillary nodes
2. Patients with high surviving chances have positive axillary nodes between 0-4 approximately
3. Survival chances are getting comparitively low, when positive axillary nodes greater are than 4,

   Trying to build simple model using this feature :

   if (nodes) <= 0 :

   ```
   status = 'Very High Survival chances'
   ```

   elif ((nodes) > 0 and (nodes <= 4)):

   ```
   status = 'High Survival chances'
   ```

   elif nodes > 4:

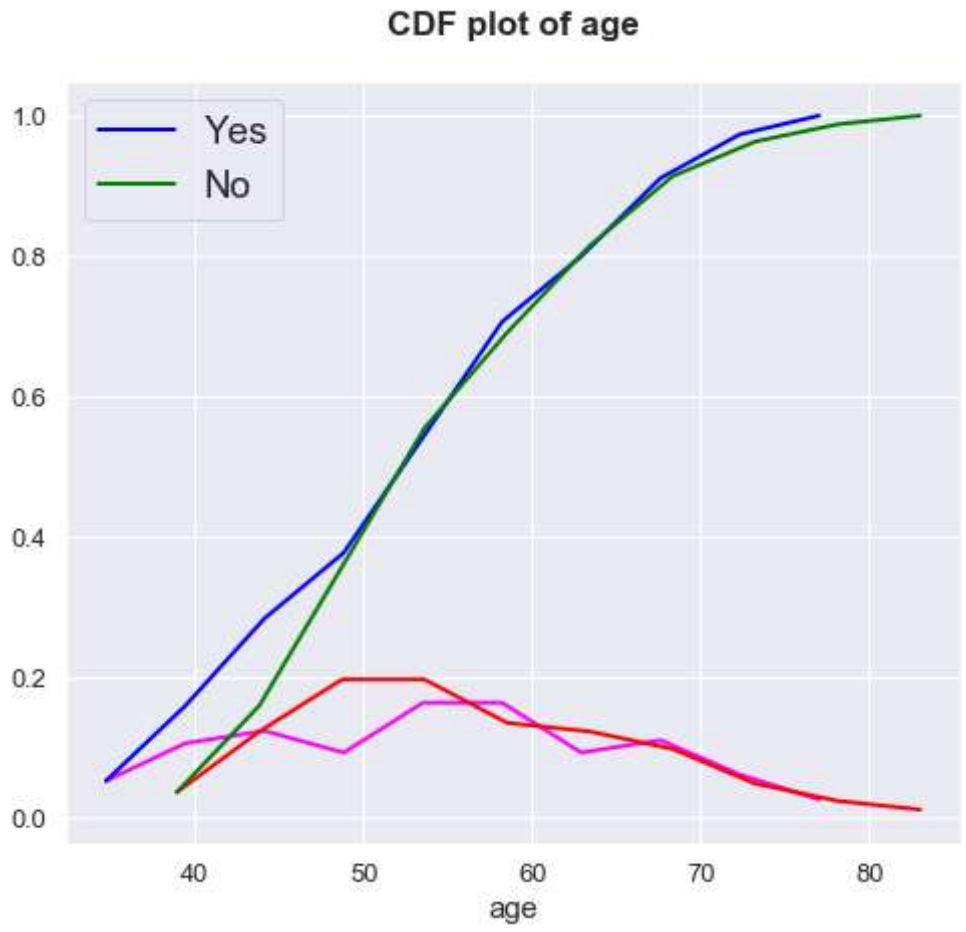
   ```
   status = 'Low Survival chances'
   ```

4. We can focus on feature 'nodes' for further EDA as it's the best we can choose among all features.

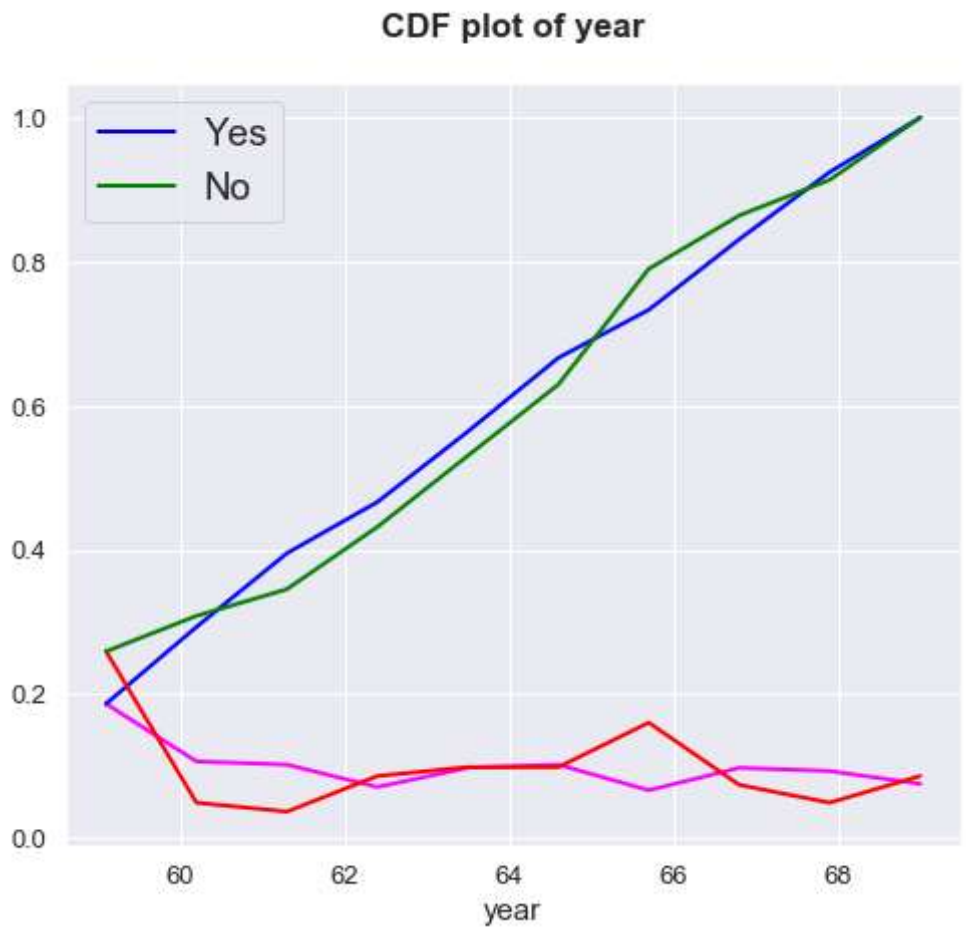
Cumulative Distribution Function (CDF)

In [91]:
```
## Cumulative Distribution Function (CDF)

for feature in list(haber.columns)[:-1]:

    plt.figure(figsize=(8,7))

    counts_y,bin_edges_y = np.histogram(haber_yes[feature],bins = 10,density = True)

    pdf_y  = counts_y/sum(counts_y)
    cdf_y = np.cumsum(pdf_y)

    print ("****************************{0}*********************************************".format(feature.upper()))
    print ('\n')
    print ("Status : Yes")
    print ('------------')
    print ("Bin Edges : {0}".format(bin_edges_y))
    print ("      PDF : {0}".format(np.round(pdf_y,3)))
    print ("      CDF : {0}".format(np.round(cdf_y,3)))


    plt.plot(bin_edges_y[1:],pdf_y,lw = 2,color = 'fuchsia')
    plt.plot(bin_edges_y[1:], cdf_y,label = 'Yes',lw = 2, color = 'blue')
    plt.xlabel (feature,fontsize=15)
    plt.title('\nCDF plot of {0}\n'.format(feature),fontsize = 17,fontweight= "bold")
    plt.tick_params(labelsize=13)


    counts_n,bin_edges_n = np.histogram(haber_no[feature],bins = 10,density = True)

    pdf_n  = counts_n/sum(counts_n)
    cdf_n = np.cumsum(pdf_n)

    print ('\n')
    print ("Status : No")
    print ('------------')
    print ("Bin Edges : {0}".format(bin_edges_n))
    print ("      PDF : {0}".format(np.round(pdf_n,3)))
    print ("      CDF : {0}".format(np.round(cdf_n,3)))


    plt.plot(bin_edges_n[1:],pdf_n,lw = 2,color = "red")
    plt.plot(bin_edges_n[1:], cdf_n,label = "No",lw = 2,color = "green")
    plt.legend()
    plt.show()
```

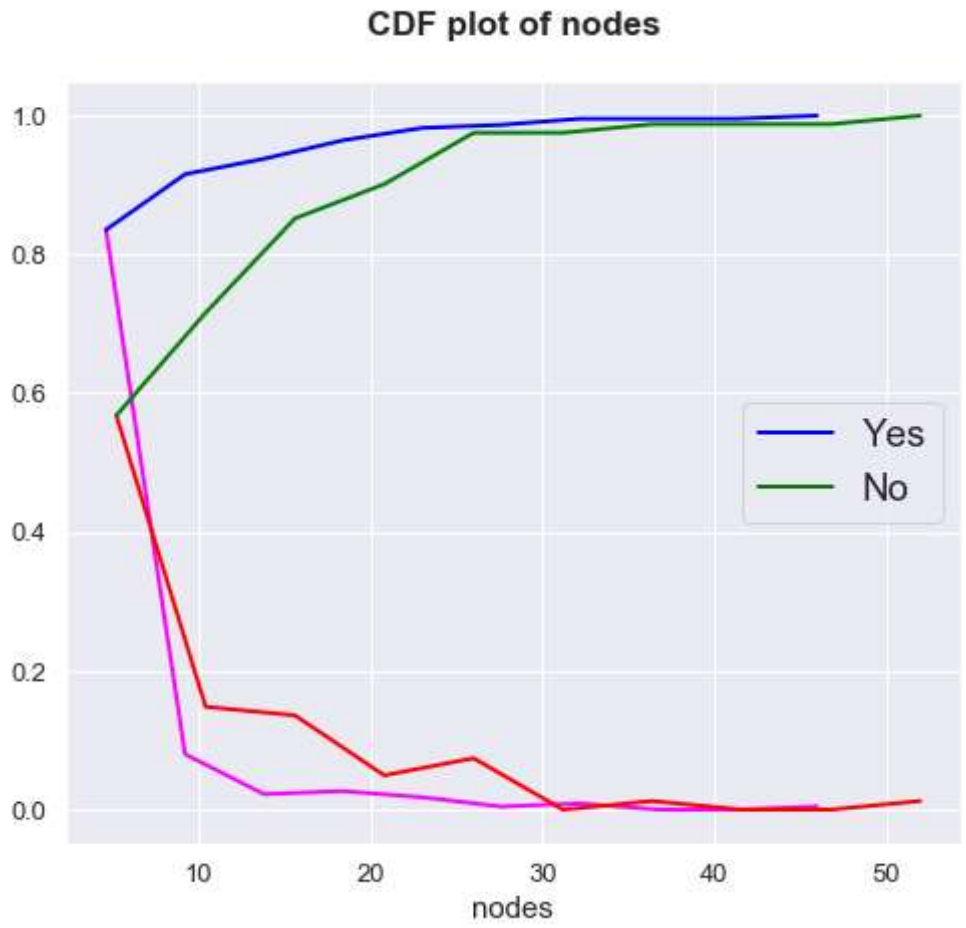
```
**************************AGE*****************************************
*******


Status : Yes
------------
Bin Edges : [30.  34.7 39.4 44.1 48.8 53.5 58.2 62.9 67.6 72.3 77. ]
      PDF : [0.053 0.107 0.124 0.093 0.164 0.164 0.093 0.111 0.062 0.027]
      CDF : [0.053 0.16  0.284 0.378 0.542 0.707 0.8   0.911 0.973 1.   ]


Status : No
------------
Bin Edges : [34.  38.9 43.8 48.7 53.6 58.5 63.4 68.3 73.2 78.1 83. ]
      PDF : [0.037 0.123 0.198 0.198 0.136 0.123 0.099 0.049 0.025 0.012]
      CDF : [0.037 0.16  0.358 0.556 0.691 0.815 0.914 0.963 0.988 1.   ]
```



```
**************************YEAR*****************************************
********


Status : Yes
------------
Bin Edges : [58.  59.1 60.2 61.3 62.4 63.5 64.6 65.7 66.8 67.9 69. ]
      PDF : [0.187 0.107 0.102 0.071 0.098 0.102 0.067 0.098 0.093 0.076]
      CDF : [0.187 0.293 0.396 0.467 0.564 0.667 0.733 0.831 0.924 1.   ]


Status : No
------------
Bin Edges : [58.  59.1 60.2 61.3 62.4 63.5 64.6 65.7 66.8 67.9 69. ]
      PDF : [0.259 0.049 0.037 0.086 0.099 0.099 0.16  0.074 0.049 0.086]
      CDF : [0.259 0.309 0.346 0.432 0.531 0.63  0.79  0.864 0.914 1.   ]
```



```
**************************NODES*****************************************
*********


Status : Yes
------------
Bin Edges : [ 0.   4.6  9.2 13.8 18.4 23.  27.6 32.2 36.8 41.4 46. ]
      PDF : [0.836 0.08  0.022 0.027 0.018 0.004 0.009 0.    0.    0.004]
      CDF : [0.836 0.916 0.938 0.964 0.982 0.987 0.996 0.996 0.996 1.   ]


Status : No
------------
Bin Edges : [ 0.   5.2 10.4 15.6 20.8 26.  31.2 36.4 41.6 46.8 52. ]
      PDF : [0.568 0.148 0.136 0.049 0.074 0.    0.012 0.    0.    0.012]
      CDF : [0.568 0.716 0.852 0.901 0.975 0.975 0.988 0.988 0.988 1.   ]
```



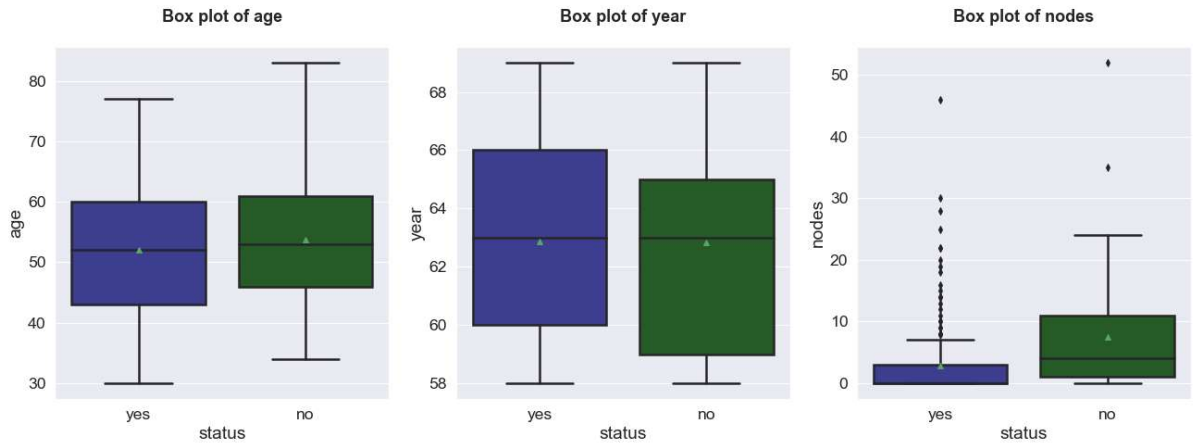
Observations:

By observing the combined CDF of feature 'nodes' for both the 'status':

1. Almost 83% of the survived patients have positive axillary nodes <= 5
2. As the number of nodes increases survival chances reduces

Box Plot

In [48]:
```
## Box Plot & Violin Plot

fig, axes = plt.subplots(1, 3, figsize=(22, 7))
sns.set(font_scale=1.6)

for i, feature in enumerate(list(haber.columns)[:-1]):
    g = sns.boxplot( x='status', y = feature, data=haber, ax=axes[i],
                linewidth= 2.5,palette=pal,order=['yes','no'],saturation=0.4,s
howmeans = True)
    ax = axes[i]
    ax.set_title('\nBox plot of {0}\n'.format(feature),fontsize = 18,fontweigh
t="bold")

plt.show()
```



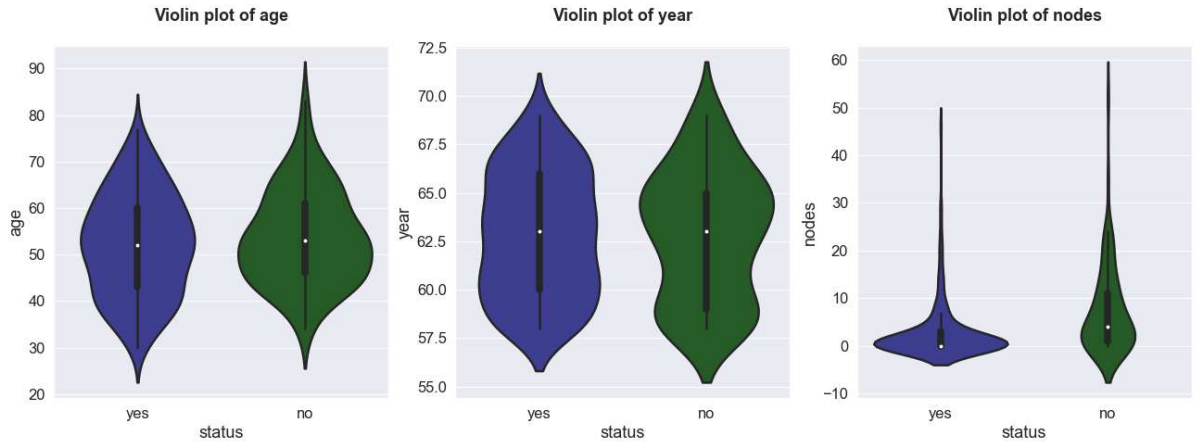
Observations

By observing box-plot of feature 'nodes':

1. For status = 'yes', 25th and 50th percentile are same at zero. This means 50% of the surviving population have 0 positive axillary nodes

Violin Plot

In [50]:
```
fig, axes = plt.subplots(1, 3, figsize=(22, 7))
sns.set(font_scale=1.5)

for i, feature in enumerate(list(haber.columns)[:-1]):
    g = sns.violinplot( x='status', y = feature, data=haber, ax=axes[i],
                linewidth= 2.5,palette=pal,order=['yes','no'],saturation=0.4,s
plit = True)
    ax = axes[i]
    ax.set_title('\nViolin plot of {0}\n'.format(feature),fontsize = 18,fontwe
ight="bold")
plt.show()
```



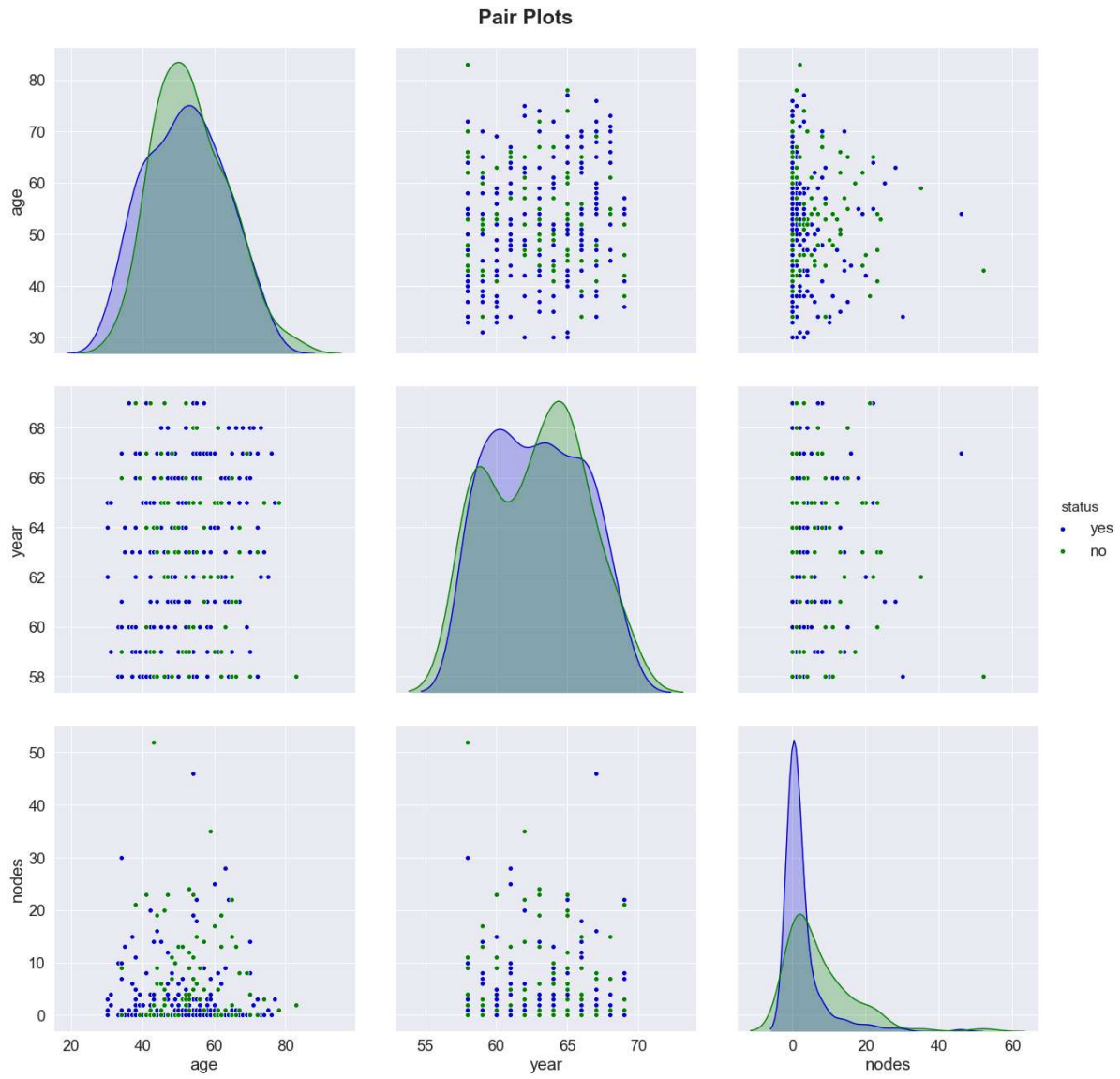
Observations

1. Patients with more than 1 positive axillary nodes are less likely to survive.More the number of nodes, lesser the survival chances. However it cannot gurantee 100% survival as there are some percentages of people with no positive nodes didn't survive.
2. Patients treated after 1965 have slightly higher chances of survival. Whereas, patients treated before 1960 have slightly lower chances of survival
3. There were comparitively more number of people between age-group 45-55 who didn't survive.

Bi-variate Analysis

Pair Plots

In [88]:
```
## Bi-variate Analysis

sns.set_style("darkgrid")
pal = ['mediumblue','green']
sns.set(font_scale=1.7)

sns.pairplot(haber, hue="status",palette=pal,height = 6,hue_order=['yes','no'
])
plt.title('\nPair Plots\n',fontsize = 25,fontweight="bold",y=2.1,x = -0.7)
plt.show()
```

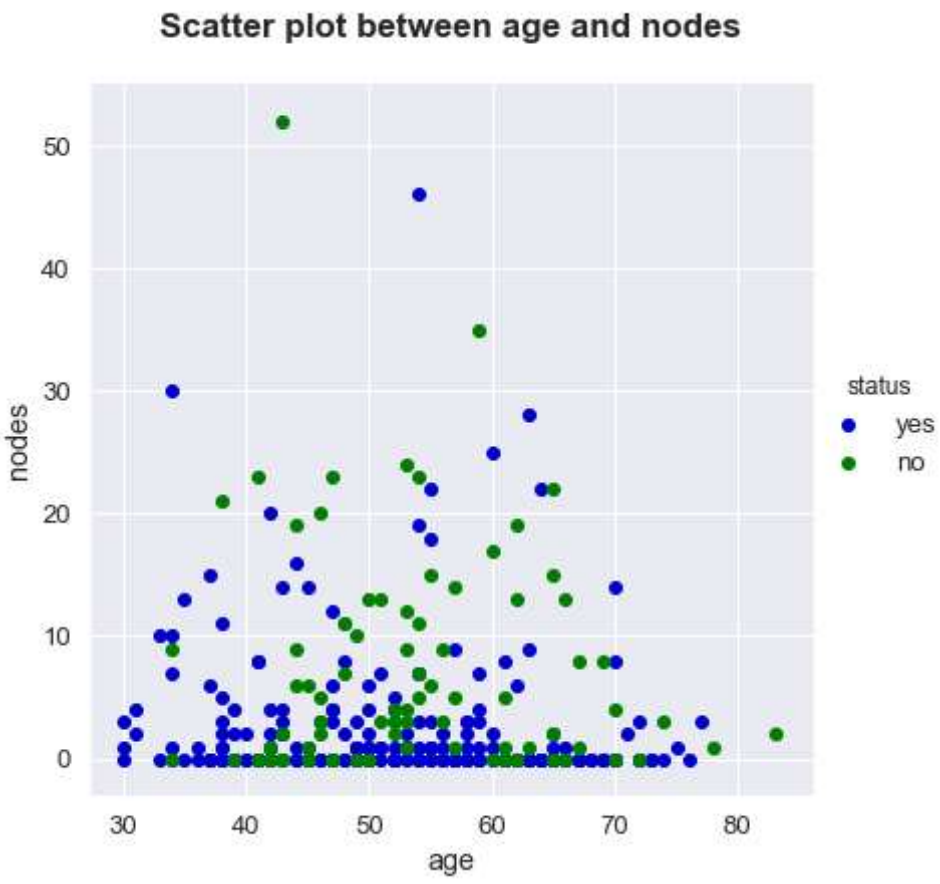


Observations

1. The plot between age and nodes is comparatively better. Hence exploring it seperately.

2-D Scatter Plots

In [70]:
```
sns.set_style("darkgrid")
pal = ['mediumblue','green']
sns.set(font_scale=1.2)

g = sns.FacetGrid(haber, hue = 'status',palette=pal,height = 6,hue_order=['ye
s','no'])
g.map(plt.scatter,'age','nodes').add_legend()
plt.title('\nScatter plot between age and nodes\n',fontsize = 17,fontweight="b
old")
plt.show()
```



Observations

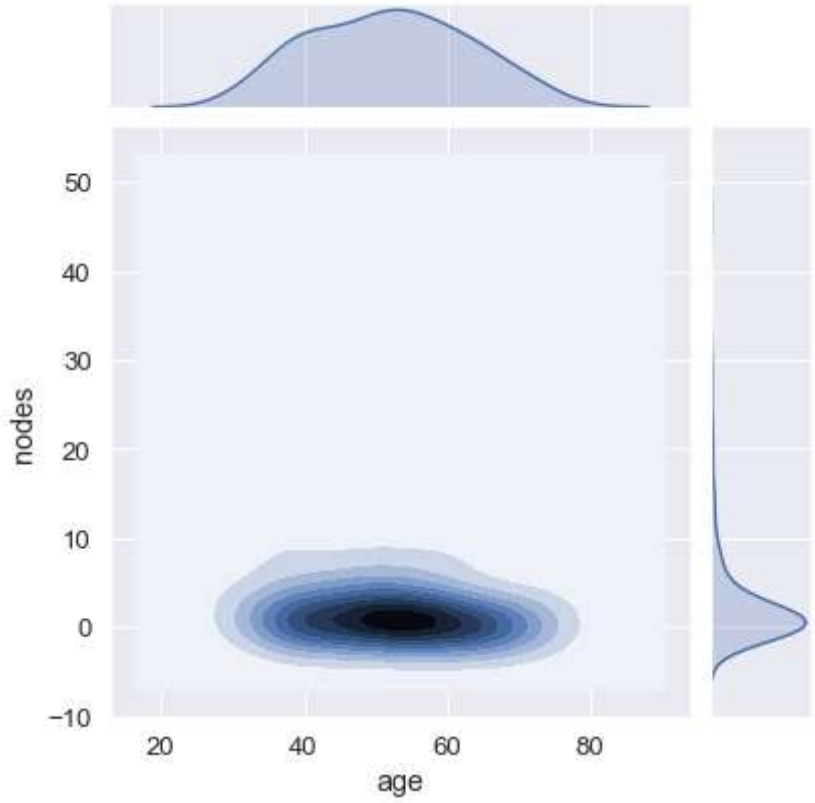
1. Patients with zero nodes are more likely to survive irrespective of their age
2. There are very less number of patients with positive axillary nodes greater than 25
3. Patients with age < 40 and positive axillary nodes < 10 are more likely to survive.
4. Patients with age > 50 and positive axillary nodes > 10 have low chances of survival

Multivariate Analysis

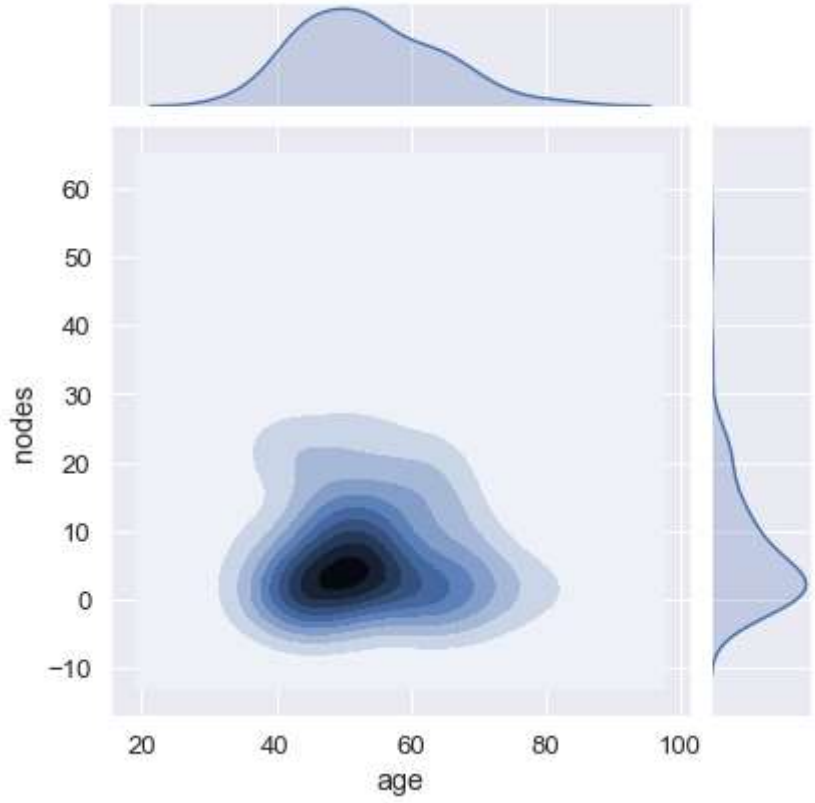
Contour Plot

In [85]:
```
sns.jointplot(x = "age", y = "nodes", data = haber_yes, kind = "kde")
plt.title('\nContour plot between Age and Nodes for Status : Yes\n',fontsize =
15,fontweight="bold",y = 1.2,x = -2.5)
plt.show()

print ('\n')
sns.jointplot(x = "age", y = "nodes", data = haber_no, kind = "kde")
plt.title('\nContour plot between Age and Nodes for Status : No\n',fontsize =
15,fontweight="bold",y = 1.2,x = -2.5)
plt.show()
```

Contour plot between Age and Nodes for Status : Yes



Contour plot between Age and Nodes for Status : No

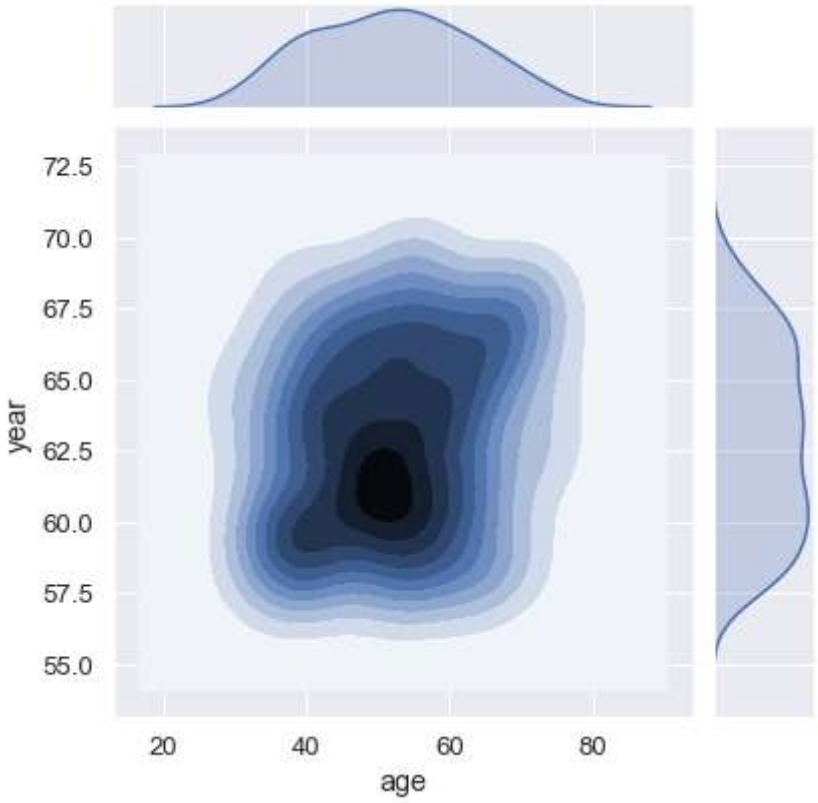


Observations :

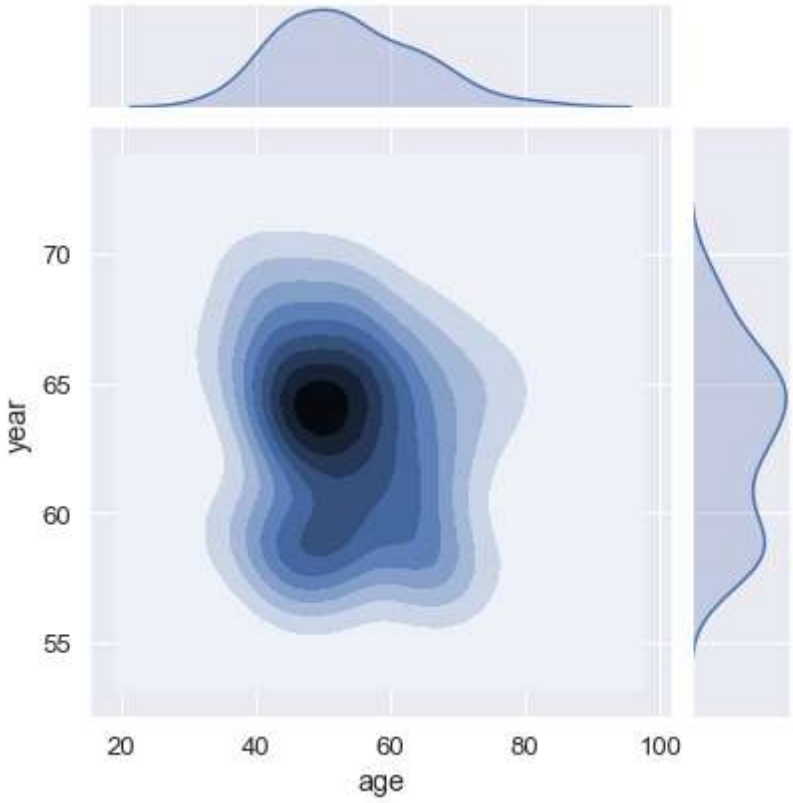
Density of plot with survival status as 'yes' is more between age 43-60 and positive axillary nodes between 0-3

In [87]:
```
sns.jointplot(x = "age", y = "year", data = haber_yes, kind = "kde")
plt.title('\nContour plot between Age and Year for Status : Yes\n',fontsize =
15,fontweight="bold",y = 1.2,x = -2.5)
plt.show()

sns.jointplot(x = "age", y = "year", data = haber_no, kind = "kde")
plt.title('\nContour plot between Age and Year for Status : No\n',fontsize = 1
5,fontweight="bold",y = 1.2,x = -2.5)
plt.show()
```

Contour plot between Age and Year for Status : Yes



Contour plot between Age and Year for Status : No



Observations

1. Plot with survival status as 'yes' is more dense between age-range 45-55 and year 1960-1963
2. Plot with survival status as 'no' is more dense between age-range 45-55 and year 1964-1965