The aim of this project is to perform a EDA on Breast Cancer Survival data set Using classifier models to get the best prediction accuracy.

In [4]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import RandomForestClassifier
import warnings
warnings.filterwarnings('ignore')
```

In [5]:

```
datas = pd.read_csv("BRCA.csv")
print(datas.head())
```

```
     Patient_ID   Age  Gender  Protein1  Protein2  Protein3  Protein4  \
0  TCGA-D8-A1XD  36.0  FEMALE  0.080353   0.42638   0.54715  0.273680
1  TCGA-EW-A1OX  43.0  FEMALE -0.420320   0.57807   0.61447 -0.031505
2  TCGA-A8-A079  69.0  FEMALE  0.213980   1.31140  -0.32747 -0.234260
3  TCGA-D8-A1XR  56.0  FEMALE  0.345090  -0.21147  -0.19304  0.124270
4  TCGA-BH-A0BF  56.0  FEMALE  0.221550   1.90680   0.52045 -0.311990

  Tumour_Stage                      Histology ER status PR status HER2 status  \
0          III  Infiltrating Ductal Carcinoma  Positive  Positive    Negative
1           II             Mucinous Carcinoma  Positive  Positive    Negative
2          III  Infiltrating Ductal Carcinoma  Positive  Positive    Negative
3           II  Infiltrating Ductal Carcinoma  Positive  Positive    Negative
4           II  Infiltrating Ductal Carcinoma  Positive  Positive    Negative

                 Surgery_type Date_of_Surgery Date_of_Last_Visit  \
0  Modified Radical Mastectomy       15-Jan-17          19-Jun-17
1                  Lumpectomy       26-Apr-17          09-Nov-18
2                       Other       08-Sep-17          09-Jun-18
3  Modified Radical Mastectomy       25-Jan-17          12-Jul-17
4                       Other       06-May-17          27-Jun-19

  Patient_Status
0          Alive
1           Dead
2          Alive
3          Alive
4           Dead
```

Finding null values

In [6]:

```
datas.isnull().sum()
```

Out[6]:

```
Patient_ID             7
Age                    7
Gender                 7
Protein1               7
Protein2               7
Protein3               7
Protein4               7
Tumour_Stage           7
Histology              7
ER status              7
PR status              7
HER2 status            7
Surgery_type           7
Date_of_Surgery        7
Date_of_Last_Visit    24
Patient_Status        20
dtype: int64
```

Changing column names

In [7]:

```
datas.rename(columns={'ER status':'ER_status',
 'PR status':'PR_status',
 'HER2 status':'HER2_status'},inplace=True)
```

Filling null values

In [8]:

```
datas['Age']=datas['Age'].fillna(datas['Age'].mean())
datas['Protein1']=datas['Protein1'].fillna(datas['Protein1'].mean())
datas['Protein2']=datas['Protein2'].fillna(datas['Protein2'].mean())
datas['Protein3']=datas['Protein3'].fillna(datas['Protein3'].mean())
datas['Protein4']=datas['Protein4'].fillna(datas['Protein4'].mean())
```

In [9]:

```
datas['Gender'].fillna('FEMALE',inplace=True)
datas['Tumour_Stage'].fillna('II',inplace=True)
datas['Histology'].fillna('Infiltrating Ductal Carcinoma',inplace=True)
datas['ER_status'].fillna('Positive',inplace=True)
datas['PR_status'].fillna('Positive',inplace=True)
datas['HER2_status'].fillna('Negative',inplace=True)
datas['Surgery_type'].fillna('Other',inplace=True)
datas['Patient_Status'].fillna('Alive',inplace=True)
```

In [10]:

```
data=datas.drop(['Patient_ID','Date_of_Surgery','Date_of_Last_Visit'],axis=1)
data
```

Out[10]:

	Age	Gender	Protein1	Protein2	Protein3	Protein4	Tumour_Stage	Histology	ER_status	PR_status	HER2_status	Surgery_type	Patient_Status
0	36.000000	FEMALE	0.080353	0.426380	0.547150	0.273680	III	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Modified Radical Mastectomy	Alive
1	43.000000	FEMALE	-0.420320	0.578070	0.614470	-0.031505	II	Mucinous Carcinoma	Positive	Positive	Negative	Lumpectomy	Dead
2	69.000000	FEMALE	0.213980	1.311400	-0.327470	-0.234260	III	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
3	56.000000	FEMALE	0.345090	-0.211470	-0.193040	0.124270	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Modified Radical Mastectomy	Alive
4	56.000000	FEMALE	0.221550	1.906800	0.520450	-0.311990	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Dead
...	...	...	...	...	...	...	...	...	...	...	...	...	...
336	58.886228	FEMALE	-0.029991	0.946896	-0.090204	0.009819	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
337	58.886228	FEMALE	-0.029991	0.946896	-0.090204	0.009819	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
338	58.886228	FEMALE	-0.029991	0.946896	-0.090204	0.009819	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
339	58.886228	FEMALE	-0.029991	0.946896	-0.090204	0.009819	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
340	58.886228	FEMALE	-0.029991	0.946896	-0.090204	0.009819	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive

341 rows × 13 columns

In [11]:

```
data.isnull().sum()
```

Out[11]:

```
Age               0
Gender            0
Protein1          0
Protein2          0
Protein3          0
Protein4          0
Tumour_Stage      0
Histology         0
ER_status         0
PR_status         0
HER2_status       0
Surgery_type      0
Patient_Status    0
dtype: int64
```

In [12]:

```
data.shape
```

Out[12]:

```
(341, 13)
```

In [13]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 341 entries, 0 to 340
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             341 non-null    float64
 1   Gender          341 non-null    object
 2   Protein1        341 non-null    float64
 3   Protein2        341 non-null    float64
 4   Protein3        341 non-null    float64
 5   Protein4        341 non-null    float64
 6   Tumour_Stage    341 non-null    object
 7   Histology       341 non-null    object
 8   ER_status       341 non-null    object
 9   PR_status       341 non-null    object
 10  HER2_status     341 non-null    object
 11  Surgery_type    341 non-null    object
 12  Patient_Status  341 non-null    object
dtypes: float64(5), object(8)
memory usage: 34.8+ KB
```

In [14]:

```
data.describe()
```

Out[14]:

	Age	Protein1	Protein2	Protein3	Protein4
count	341.000000	341.000000	341.000000	341.000000	341.000000
mean	58.886228	-0.029991	0.946896	-0.090204	0.009819
std	12.827094	0.557756	0.902204	0.579120	0.622545
min	29.000000	-2.340900	-0.978730	-1.627400	-2.025500
25%	49.000000	-0.349620	0.372460	-0.513320	-0.374070
50%	58.886228	0.000000	0.953990	-0.161100	0.030815
75%	67.000000	0.336530	1.609400	0.251210	0.412550
max	90.000000	1.593600	3.402200	2.193400	1.629900

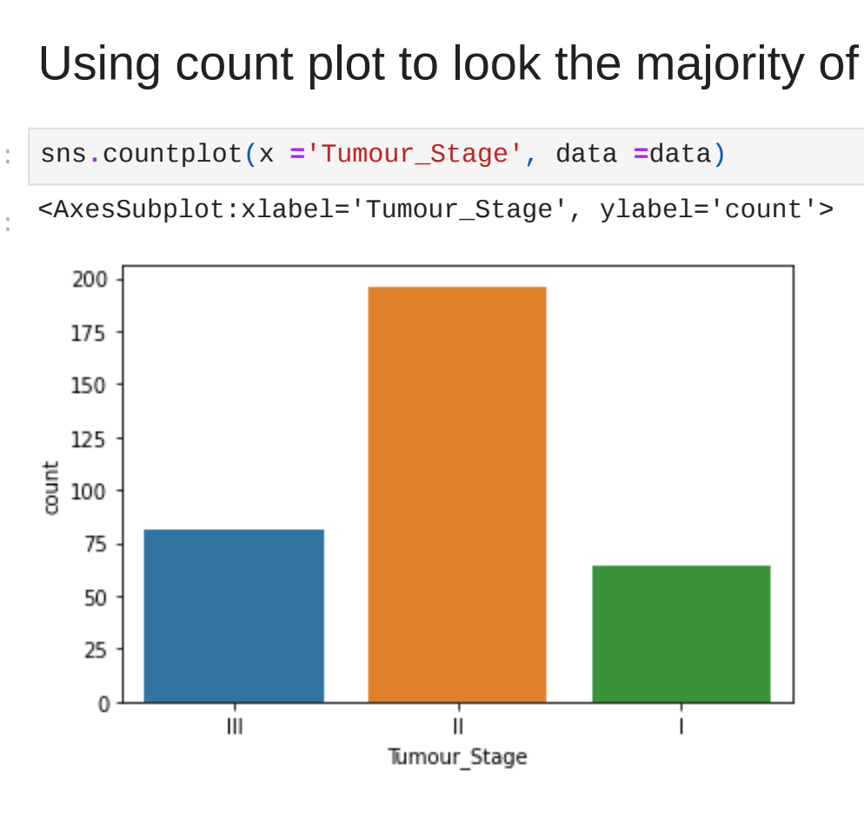
Using count plot to look the majority of Tumour_stage

In [15]:

```
sns.countplot(x ='Tumour_Stage', data =data)
```

Out[15]:

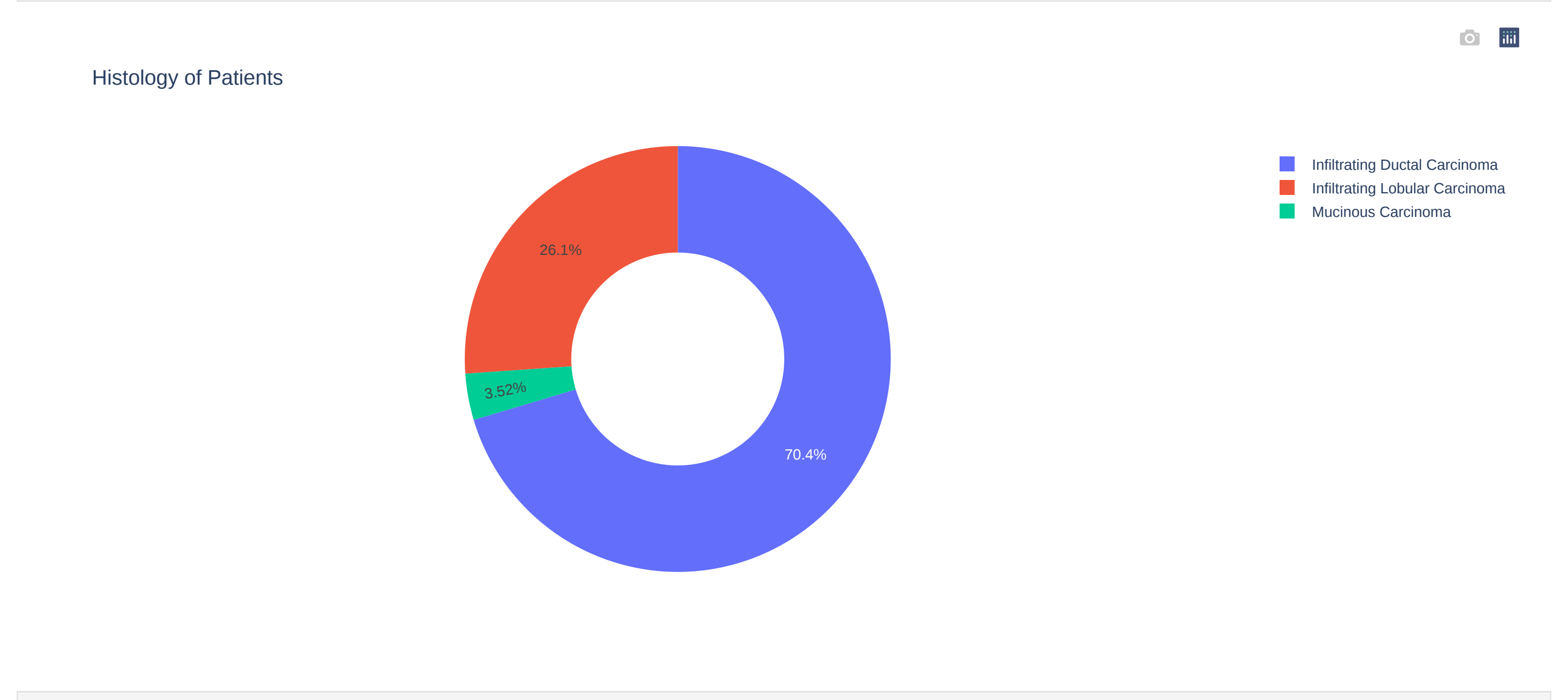
```
<AxesSubplot:xlabel='Tumour_Stage', ylabel='count'>
```



Pie plot

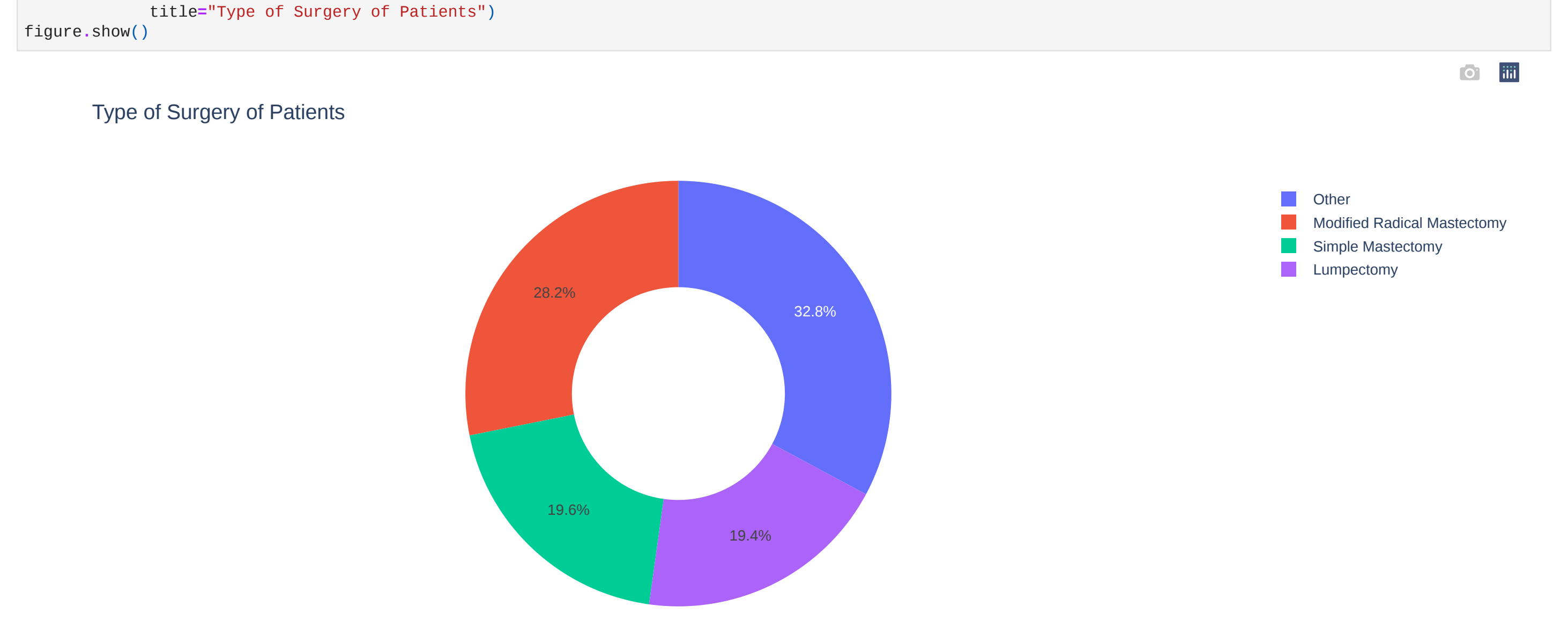
In [16]:

```
histology = data["Histology"].value_counts()
transactions = histology.index
quantity = histology.values
figure = px.pie(data,
             values=quantity,
             names=transactions,hole = 0.5,
             title="Histology of Patients")
figure.show()
```



In [17]:

```
surgery = data["Surgery_type"].value_counts()
transactions = surgery.index
quantity = surgery.values
figure = px.pie(data,
             values=quantity,
             names=transactions,hole = 0.5,
             title="Type of Surgery of Patients")
figure.show()
```



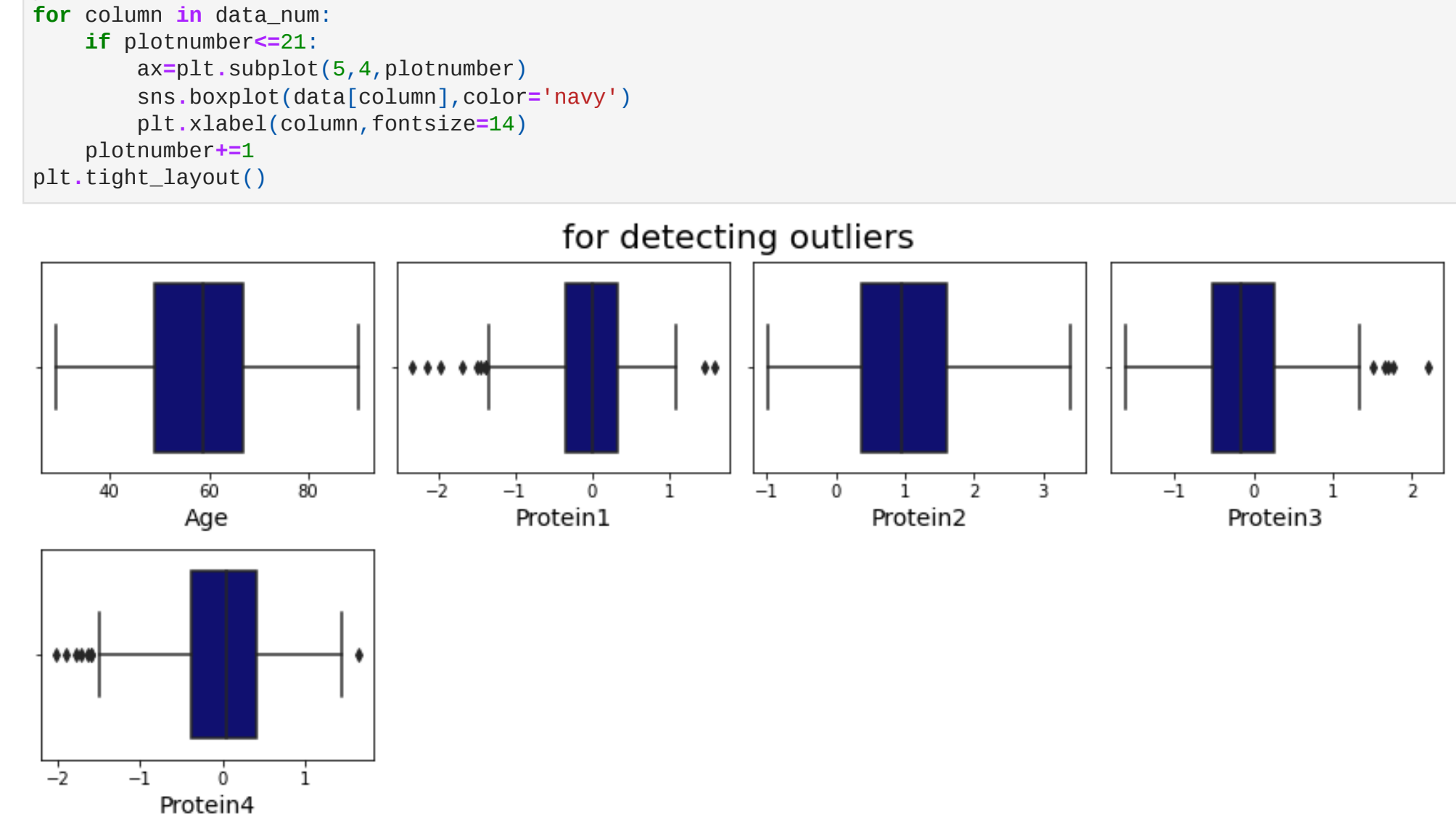
In [18]:

```
data_catg=data.select_dtypes('object')
data_num=data.select_dtypes(['int64','float64'])
```

Finding outliers

In [19]:

```
plt.figure(figsize=(12,12),facecolor = 'white')
plt.suptitle('for detecting outliers',fontsize=20)
plotnumber=1
for column in data_num:
    if plotnumber<=21:
        ax=plt.subplot(5,4,plotnumber)
        sns.boxplot(data[column],color='navy')
        plt.xlabel(column,fontsize=14)
    plotnumber+=1
plt.tight_layout()
```



In [20]:

```
features = data[['Age','Protein1','Protein2','Protein3','Protein4']]
from scipy.stats import zscore
z=np.abs(zscore(features))

z_data = data[(z<3).all(axis=1)]
z_data.head()
```

Out[20]:

	Age	Gender	Protein1	Protein2	Protein3	Protein4	Tumour_Stage	Histology	ER_status	PR_status	HER2_status	Surgery_type	Patient_Status
0	36.0	FEMALE	0.080353	0.42638	0.54715	0.273680	III	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Modified Radical Mastectomy	Alive
1	43.0	FEMALE	-0.420320	0.57807	0.61447	-0.031505	II	Mucinous Carcinoma	Positive	Positive	Negative	Lumpectomy	Dead
2	69.0	FEMALE	0.213980	1.31140	-0.32747	-0.234260	III	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Alive
3	56.0	FEMALE	0.345090	-0.21147	-0.19304	0.124270	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Modified Radical Mastectomy	Alive
4	56.0	FEMALE	0.221550	1.90680	0.52045	-0.311990	II	Infiltrating Ductal Carcinoma	Positive	Positive	Negative	Other	Dead

In [21]:

```
print("shape of dataset before removing outliners:",data.shape)
print("shape of dataset after removing outliners:",z_data.shape)
```

```
shape of dataset before removing outliners: (341, 13)
shape of dataset after removing outliners: (332, 13)
```

In [22]:

```
data_loss=(341-332)/341*100
data_loss
```

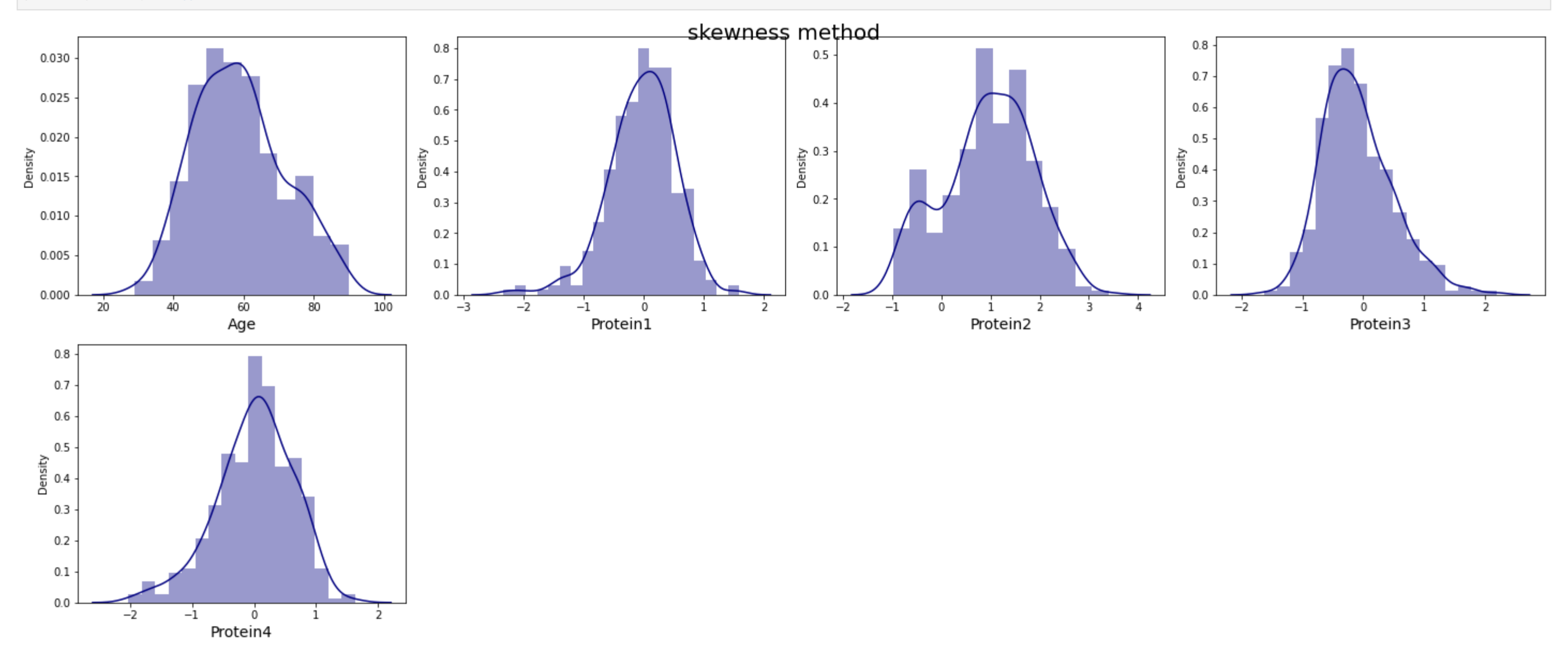
Out[22]:

```
2.6392961876832843
```

Finding distribution

In [23]:

```
plt.figure(figsize=(20,20),facecolor = 'white')
plt.suptitle('skewness method',fontsize=20)
plotnumber=1
for column in data_num:
    if plotnumber<=21:
        ax=plt.subplot(5,4,plotnumber)
        sns.distplot(data[column],color='navy')
        plt.xlabel(column,fontsize=14)
    plotnumber+=1
plt.tight_layout()
```



In [24]:

```
data1=['Age','Protein1','Protein2','Protein3','Protein4']

sc = StandardScaler()
z_data[data1]=sc.fit_transform(z_data[data1].values)
z_data.skew()
```

Out[24]:

```
Age         0.289942
Protein1   -0.264117
Protein2   -0.216804
Protein3    0.449899
Protein4   -0.342820
dtype: float64
```

Label Encoding of categorial columns

In [25]:

```
categorical_col=[]
for i in z_data.dtypes.index:
    if z_data.dtypes[i]=='object':
        categorical_col.append(i)
print("categorical columns present in the dataset are:\n",categorical_col)
```

```
categorical columns present in the dataset are:
 ['Gender', 'Tumour_Stage', 'Histology', 'ER_status', 'PR_status', 'HER2_status', 'Surgery_type', 'Patient_Status']
```

In [26]:

```
from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()
z_data[categorical_col]=z_data[categorical_col].apply(LE.fit_transform)
z_data
```

Out[26]:

	Age	Gender	Protein1	Protein2	Protein3	Protein4	Tumour_Stage	Histology	ER_status	PR_status	HER2_status	Surgery_type	Patient_Status
0	-1.784311	0	0.166068	-0.605227	1.212766	0.402833	2	0	0	0	0	1	0
1	-1.235452	0	-0.794080	-0.435486	1.336365	-0.111055	1	2	0	0	0	0	1
2	0.803165	0	0.422326	0.385107	-0.393029	-0.452465	2	0	0	0	0	2	0
3	-0.216143	0	0.673758	-1.318978	-0.146217	0.151248	1	0	0	0	0	1	0
4	-0.216143	0	0.436843	1.051358	1.163745	-0.583351	1	0	0	0	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
336	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	0	0	2	0
337	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	0	0	2	0
338	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	0	0	2	0
339	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	0	0	2	0
340	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	0	0	2	0

332 rows × 13 columns

Correlation

In [27]:

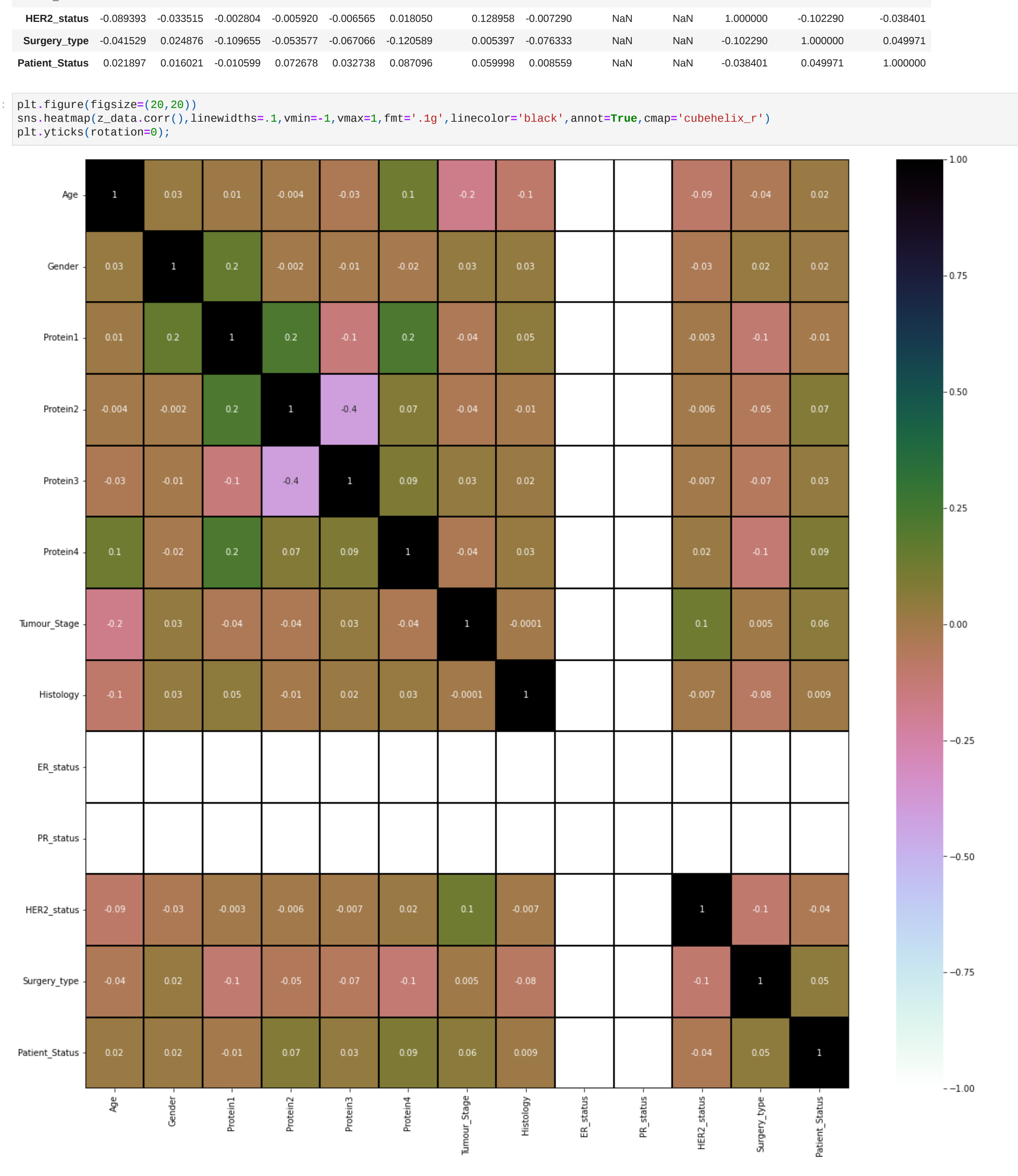
```
cor= z_data.corr()
cor
```

Out[27]:

	Age	Gender	Protein1	Protein2	Protein3	Protein4	Tumour_Stage	Histology	ER_status	PR_status	HER2_status	Surgery_type	Patient_Status
Age	1.000000	0.025919	0.010364	-0.004328	-0.033315	0.129528	-0.178129	-0.099310	NaN	NaN	-0.089393	-0.041529	0.021897
Gender	0.025919	1.000000	0.157991	-0.001784	-0.014337	-0.019740	0.034791	0.034041	NaN	NaN	-0.033515	0.024876	0.016021
Protein1	0.010364	0.157991	1.000000	0.221793	-0.134964	0.223450	-0.039992	0.047186	NaN	NaN	-0.002804	-0.109655	-0.010599
Protein2	-0.004328	-0.001784	0.221793	1.000000	-0.410183	0.072274	-0.035922	-0.014568	NaN	NaN	-0.005920	-0.053577	0.072678
Protein3	-0.033315	-0.014337	-0.134964	-0.410183	1.000000	0.088246	0.032223	0.020230	NaN	NaN	-0.006565	-0.067066	0.032738
Protein4	0.129528	-0.019740	0.223450	0.072274	0.088246	1.000000	-0.037057	0.031191	NaN	NaN	0.018050	-0.120589	0.087096
Tumour_Stage	-0.178129	0.034791	-0.039992	-0.035922	0.032223	-0.037057	1.000000	-0.000130	NaN	NaN	0.128958	0.005397	0.059998
Histology	-0.099310	0.034041	0.047186	-0.014568	0.020230	0.031191	-0.000130	1.000000	NaN	NaN	-0.007290	-0.076333	0.008559
ER_status	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
PR_status	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
HER2_status	-0.089393	-0.033515	-0.002804	-0.005920	-0.006565	0.018050	0.128958	-0.007290	NaN	NaN	1.000000	-0.102290	-0.038401
Surgery_type	-0.041529	0.024876	-0.109655	-0.053577	-0.067066	-0.120589	0.005397	-0.076333	NaN	NaN	-0.102290	1.000000	0.049971
Patient_Status	0.021897	0.016021	-0.010599	0.072678	0.032738	0.087096	0.059998	0.008559	NaN	NaN	-0.038401	0.049971	1.000000

In [28]:

```
plt.figure(figsize=(20,20))
sns.heatmap(z_data.corr(),linewidths=.1,vmin=-1,vmax=1,fmt='.1g',linecolor='black',annot=True,cmap='cubehelix_r')
plt.yticks(rotation=0);
```



Splitting columns into featues and target and removing fearures which have correlation

In [29]:

```
X=z_data.drop(['ER_status','PR_status','Patient_Status'],axis=1)
y=z_data['Patient_Status']
X
```

Out[29]:

	Age	Gender	Protein1	Protein2	Protein3	Protein4	Tumour_Stage	Histology	HER2_status	Surgery_type
0	-1.784311	0	0.166068	-0.605227	1.212766	0.402833	2	0	0	1
1	-1.235452	0	-0.794080	-0.435486	1.336365	-0.111055	1	2	0	0
2	0.803165	0	0.422326	0.385107	-0.393029	-0.452465	2	0	0	2
3	-0.216143	0	0.673758	-1.318978	-0.146217	0.151248	1	0	0	1
4	-0.216143	0	0.436843	1.051358	1.163745	-0.583351	1	0	0	2
...	...	...	...	...	...	...	...	...	...	...
336	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	2
337	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	2
338	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	2
339	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	2
340	0.010161	0	-0.045541	-0.022771	0.042589	-0.041471	1	0	0	2

332 rows × 10 columns

Creating training and testing sets

In [30]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=0)
```

In [31]:

```
def mymodel(model):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    train = model.score(X_train, y_train)
    test = model.score(X_test, y_test)

    print(f"Training Accuracy : {train}\nTesting Accuracy : {test}\n\n")


    return model
```

In [32]:

```
svc=mymodel(SVC())
```

```
Training Accuracy : 0.8060344827586207
Testing Accuracy : 0.81
```

In [33]:

```
knn=mymodel(KNeighborsClassifier())
```

```
Training Accuracy : 0.8232758620689655
Testing Accuracy : 0.78
```

In [34]:

```
dtr=mymodel(DecisionTreeClassifier())
```

```
Training Accuracy : 1.0
Testing Accuracy : 0.64
```

In [35]:

```
f=mymodel(RandomForestClassifier())
```

```
Training Accuracy : 1.0
Testing Accuracy : 0.8
```

In [36]:

```
gb=mymodel(ExtraTreesClassifier(random_state=1))
```

```
Training Accuracy : 1.0
Testing Accuracy : 0.82
```

In [ ]:

In [ ]:

In [ ]: