PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

SAAD DAHLEB BLIDA 01 UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

MASTER’S INTELLIGENT SYSTEMS ENGINEERING

KNOWLEDGE REPRESENTATION 02 (RC2)

PROJECT REPORT

BAYESIEN NETWORK IMPLEMENTATION on the BREAST CANCER DATASET

Made by :

ABDELATIF MEKRI
NAHLA YASMINE MIHOUBI

Academic year : 2023-2024

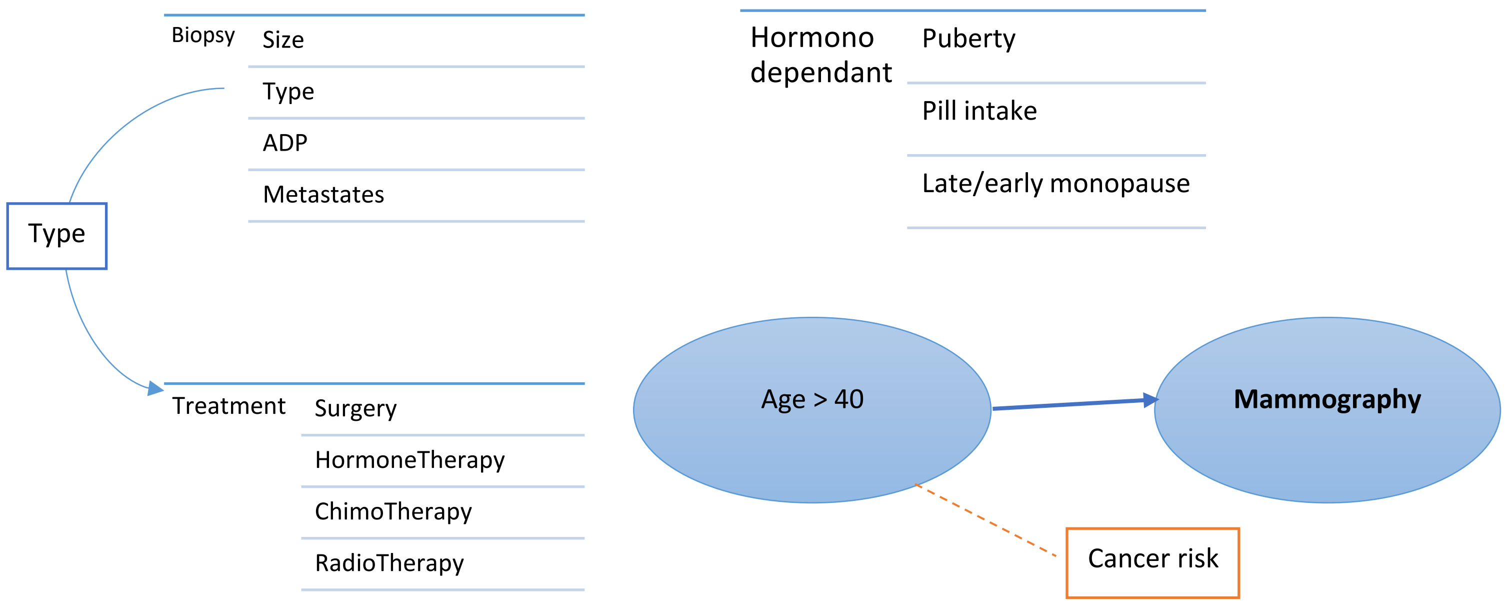
Contents

1. BUILDING THE BAYESIEN NETWORK :

1.1. CONSULTING THE EXPERT :

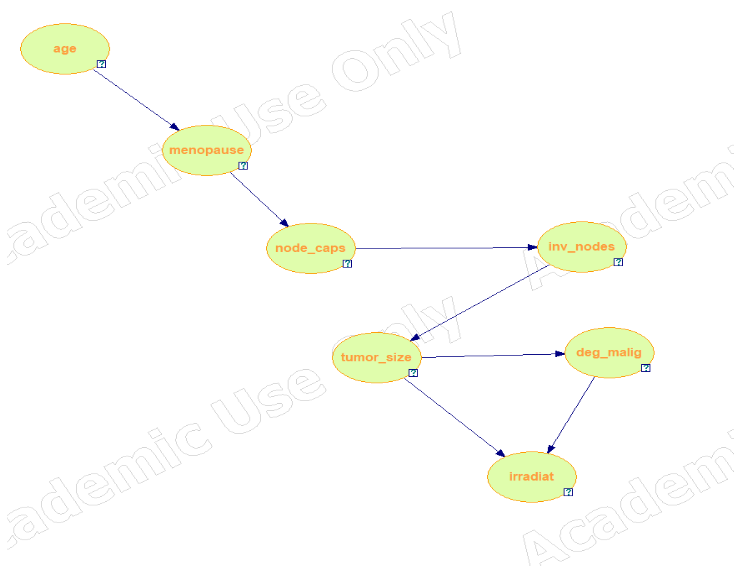
The doctor's goal was to provide us with a comprehensive understanding of cancer diagnosis in general, covering everything from its causes to detection methods, determining its stage and size, and identifying suitable treatment options.

Here is a look on some of the information that was provided by the expert :



1.2. BUILDING THE INITIAL NETWORK

After a long discussion witht the doctor we agreed finally on a suitable network structure that would be as efficient for cancer detection without having a redendance in the information that would influence or misslead the results :

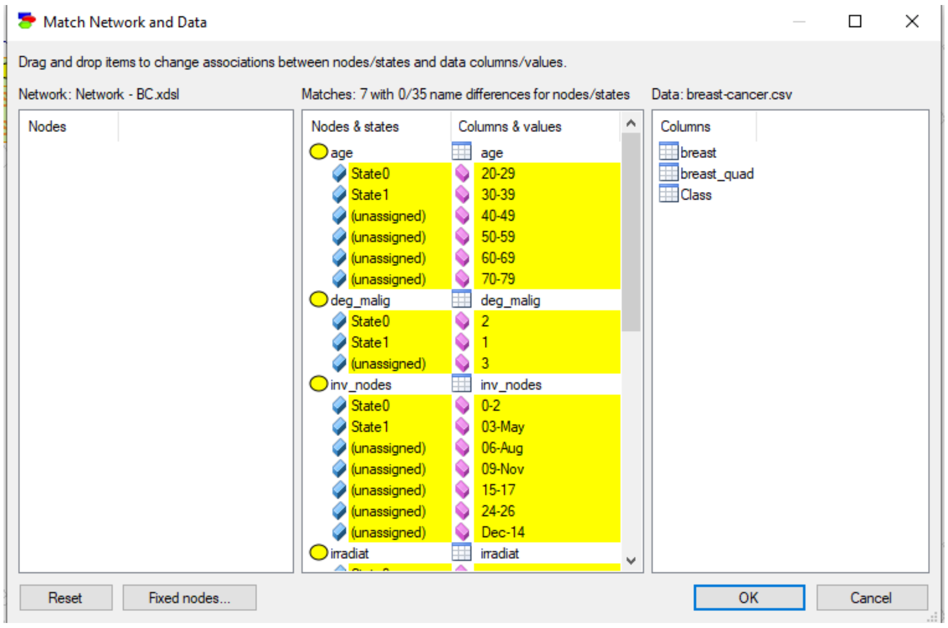


2. LOADING DATA AND LEARNING PARAMETERS

2.1. ON GeNIE

2.1.1. Loading data

After loading the data we went to the parameter learning phase , we entered the data and checked if both the noeds and the data headers are under the same name and executed .



Match Network and Data

Drag and drop items to change associations between nodes/states and data columns/values.

Network: Network - BC.xdsl

Matches: 7 with 0/35 name differences for nodes/states

Data: breast-cancer.csv

Nodes

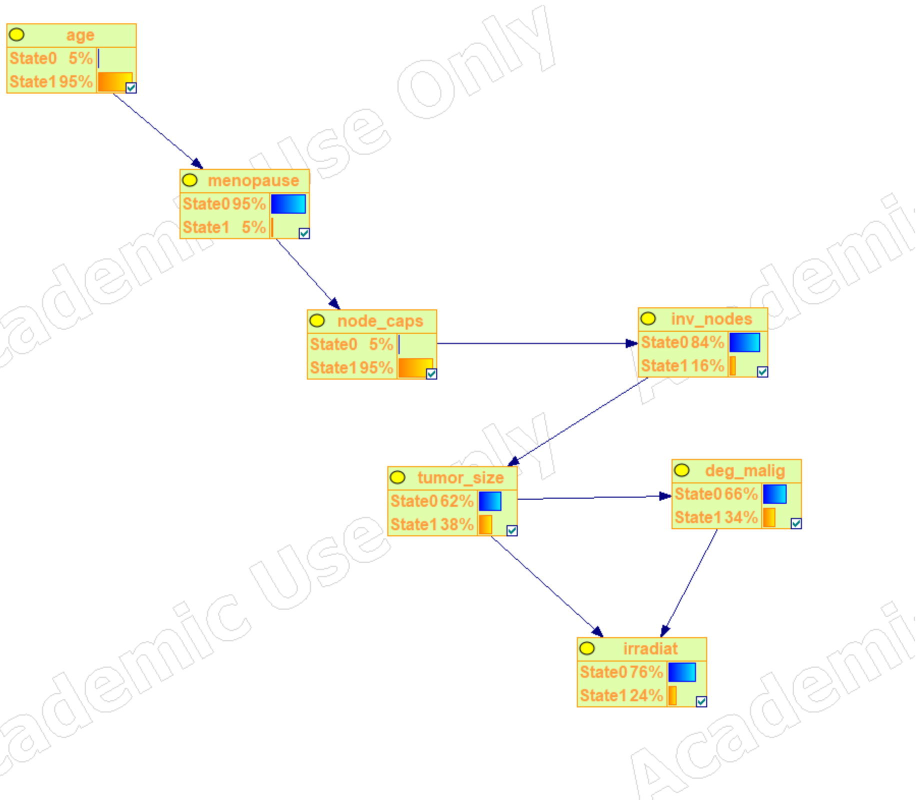
Nodes & states	Columns & values
age	age
State0	20-29
State1	30-39
(unassigned)	40-49
(unassigned)	50-59
(unassigned)	60-69
(unassigned)	70-79
deg_malig	deg_malig
State0	2
State1	1
(unassigned)	3
inv_nodes	inv_nodes
State0	0-2
State1	03-May
(unassigned)	06-Aug
(unassigned)	09-Nov
(unassigned)	15-17
(unassigned)	24-26
(unassigned)	Dec-14
irradiat	irradiat

Columns

breast
breast_quad
Class

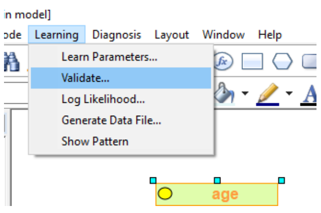
Reset · Fixed nodes... · OK · Cancel

We got the following values :

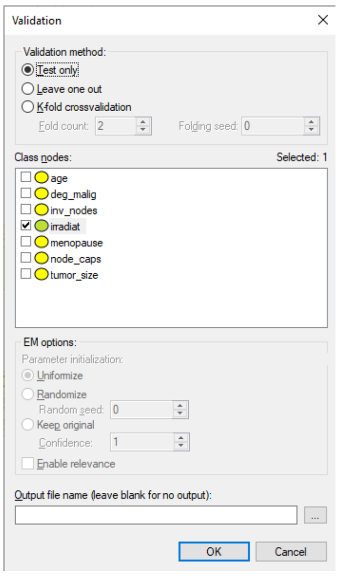


2.1.2. Testing

Started by using the Validate methode found in GeNIE's software ,after loading all the data



And then chosing the target value of which we would like to do the training on


Validation

Validation method:
(●) Test only
( ) Leave one out
( ) K-fold crossvalidation
Fold count: 2   Folding seed: 0

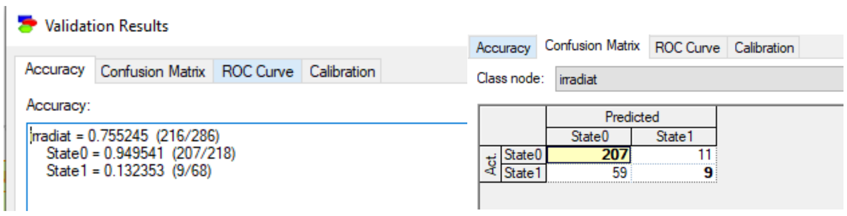
Class nodes: Selected: 1
☐ age
☐ deg_malig
☐ inv_nodes
☑ irradiat
☐ menopause
☐ node_caps
☐ tumor_size

EM options:
Parameter initialization:
(●) Uniformize
( ) Randomize
Random seed: 0
( ) Keep original
Confidence: 1
☐ Enable relevance

Output file name (leave blank for no output):

OK   Cancel

Having the following results :


Validation Results

Accuracy | Confusion Matrix | ROC Curve | Calibration

Accuracy:

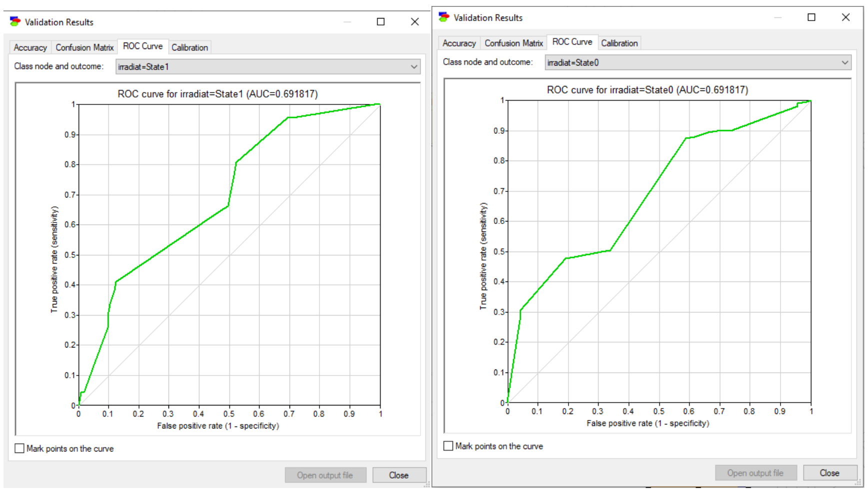
```
irradiat = 0.755245  (216/286)
    State0 = 0.949541  (207/218)
    State1 = 0.132353  (9/68)
```

Accuracy | Confusion Matrix | ROC Curve | Calibration

Class node: irradiat

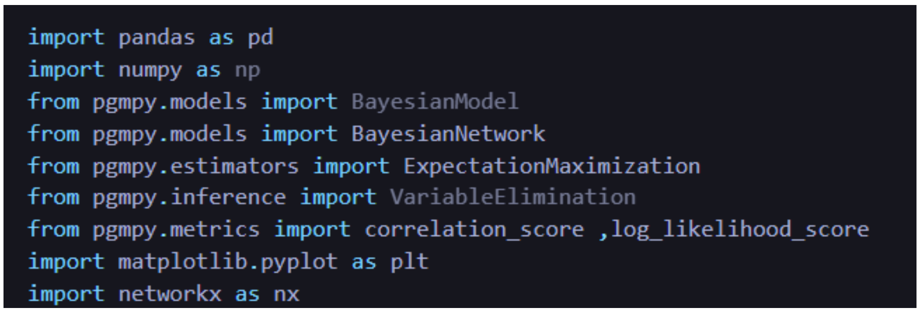
		Predicted	
		State0	State1
Act.	State0	**207**	11
	State1	59	**9**

GeNIE takes by default the 80% 20% principle , where 80% of the datais for the training and 20% of it is for the testing .



2.2. Using PGMPY

2.2.1. loading libraries



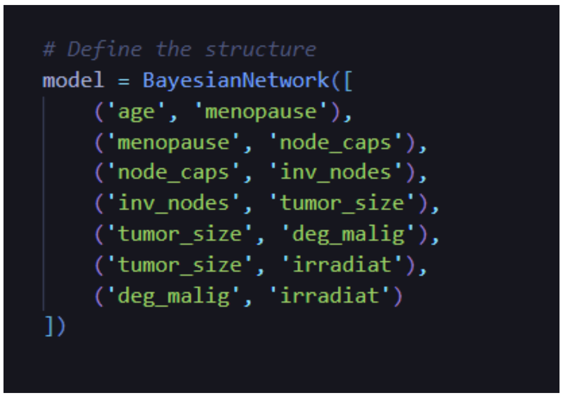
```python
import pandas as pd
import numpy as np
from pgmpy.models import BayesianModel
from pgmpy.models import BayesianNetwork
from pgmpy.estimators import ExpectationMaximization
from pgmpy.inference import VariableElimination
from pgmpy.metrics import correlation_score ,log_likelihood_score
import matplotlib.pyplot as plt
import networkx as nx
```

-The ‘pandas’ ‘numpy’ libraries are for the manipulation of the data , from reading , spltiting to attributing treatments .

-The ‘pmpy’ library we got the Bayesian model , the Expectation Maximization algorithm , matrics calculating methods .

-Where we used the ‘nx’ and ‘matplotlib’ for the visualisation of the result .

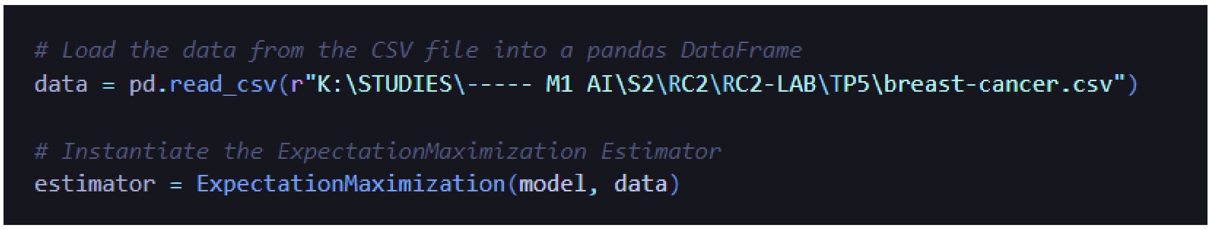
2.2.2. Defining network structure



```python
# Define the structure
model = BayesianNetwork([
    ('age', 'menopause'),
    ('menopause', 'node_caps'),
    ('node_caps', 'inv_nodes'),
    ('inv_nodes', 'tumor_size'),
    ('tumor_size', 'deg_malig'),
    ('tumor_size', 'irradiat'),
    ('deg_malig', 'irradiat')
])
```

The definition of the network structure was based in the previous implementation of the bayesien network .

2.2.3. Reading the data and calling the EM algorithm

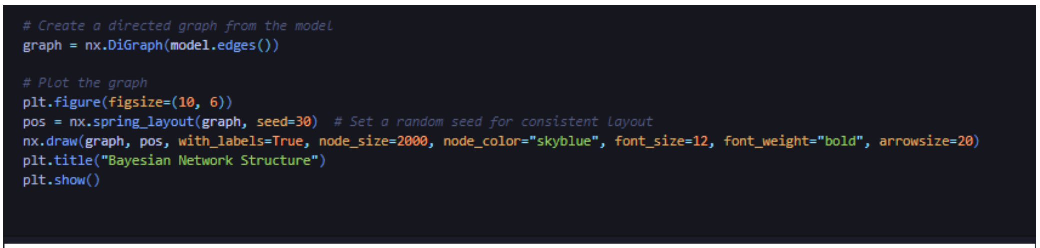


```python
# Load the data from the CSV file into a pandas DataFrame
data = pd.read_csv(r"K:\STUDIES\----- M1 AI\S2\RC2\RC2-LAB\TP5\breast-cancer.csv")

# Instantiate the ExpectationMaximization Estimator
estimator = ExpectationMaximization(model, data)
```

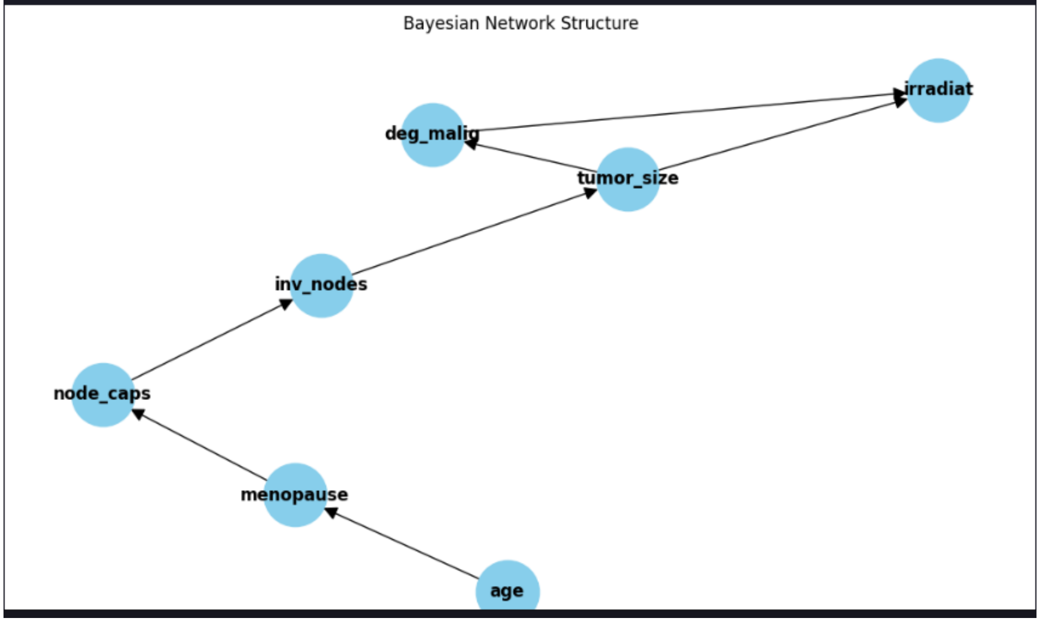
Calling the EM algorithm and feading it the defined model structure and the the headed csv file using the pandas library .

2.2.4. Plotting the network structure

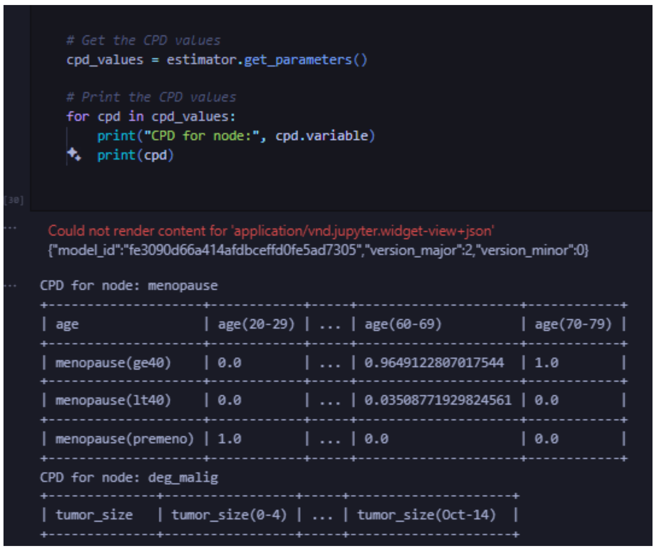
Using the nx and matplotlib libraries to define the structure of the network and plotting it .



```python
# Create a directed graph from the model
graph = nx.DiGraph(model.edges())

# Plot the graph
plt.figure(figsize=(10, 6))
pos = nx.spring_layout(graph, seed=30)  # Set a random seed for consistent layout
nx.draw(graph, pos, with_labels=True, node_size=2000, node_color="skyblue", font_size=12, font_weight="bold", arrowsize=20)
plt.title("Bayesian Network Structure")
plt.show()
```



2.2.5. CPD values

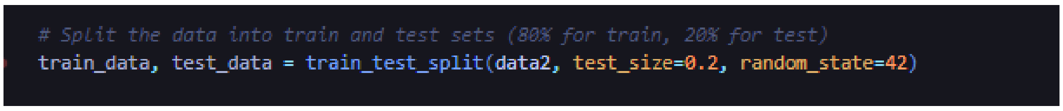


```python
# Get the CPD values
cpd_values = estimator.get_parameters()

# Print the CPD values
for cpd in cpd_values:
    print("CPD for node:", cpd.variable)
    print(cpd)
```

```
Could not render content for 'application/vnd.jupyter.widget-view+json'
{"model_id":"fe3090d66a414afdbceffd0fe5ad7305","version_major":2,"version_minor":0}

CPD for node: menopause
+-------------------+------------+-----+---------------------+------------+
| age               | age(20-29) | ... | age(60-69)          | age(70-79) |
+-------------------+------------+-----+---------------------+------------+
| menopause(ge40)   | 0.0        | ... | 0.9649122807017544  | 1.0        |
+-------------------+------------+-----+---------------------+------------+
| menopause(lt40)   | 0.0        | ... | 0.03508771929824561 | 0.0        |
+-------------------+------------+-----+---------------------+------------+
| menopause(premeno) | 1.0       | ... | 0.0                 | 0.0        |
+-------------------+------------+-----+---------------------+------------+
CPD for node: deg_malig
+------------+---------------+-----+-------------------+
| tumor_size | tumor_size(0-4) | ... | tumor_size(Oct-14) |
+------------+---------------+-----+-------------------+
```

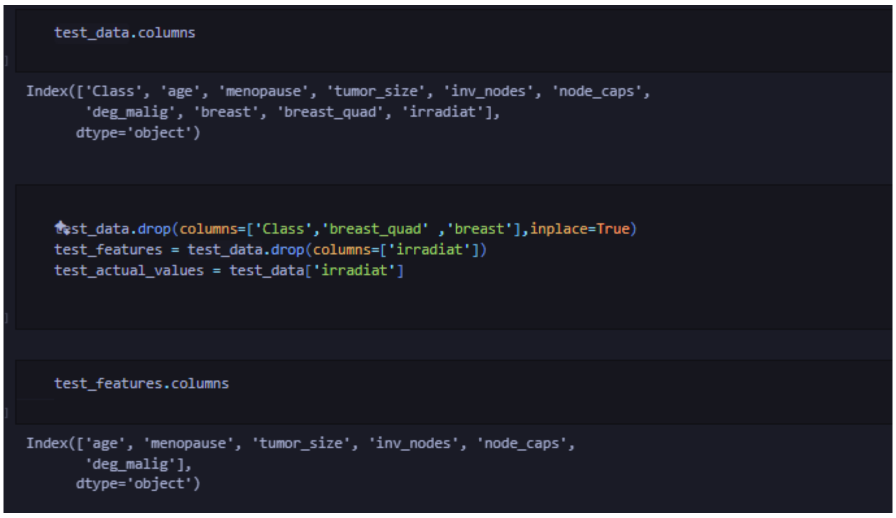
2.2.6. Validation

For this process splitting the data into training data and testing data is necessary and then performing the testing phase .



```python
# Split the data into train and test sets (80% for train, 20% for test)
train_data, test_data = train_test_split(data2, test_size=0.2, random_state=42)
```

Dropping the ‘irradiat’ class as it is our target on the testing slice of data .



```
test_data.columns
```

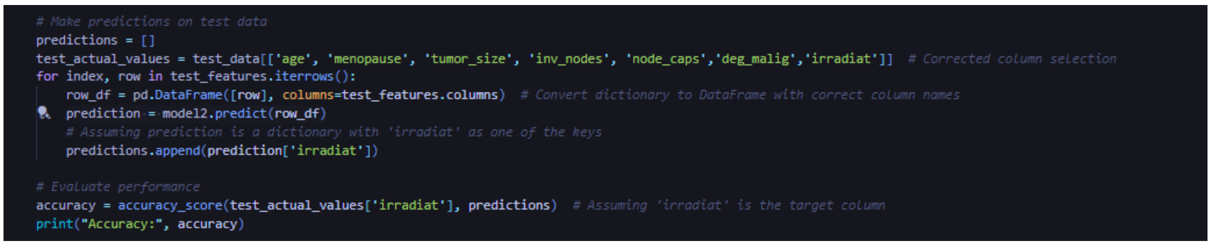
```
Index(['Class', 'age', 'menopause', 'tumor_size', 'inv_nodes', 'node_caps',
       'deg_malig', 'breast', 'breast_quad', 'irradiat'],
      dtype='object')
```

```
test_data.drop(columns=['Class','breast_quad' ,'breast'],inplace=True)
test_features = test_data.drop(columns=['irradiat'])
test_actual_values = test_data['irradiat']
```

```
test_features.columns
```

```
Index(['age', 'menopause', 'tumor_size', 'inv_nodes', 'node_caps',
       'deg_malig'],
      dtype='object')
```

Testing the test data on calculating the accuracy of the result .



```
# Make predictions on test data
predictions = []
test_actual_values = test_data[['age', 'menopause', 'tumor_size', 'inv_nodes', 'node_caps','deg_malig','irradiat']]  # Corrected column selection
for index, row in test_features.iterrows():
    row_df = pd.DataFrame([row], columns=test_features.columns)  # Convert dictionary to DataFrame with correct column names
    prediction = model2.predict(row_df)
    # Assuming prediction is a dictionary with 'irradiat' as one of the keys
    predictions.append(prediction['irradiat'])

# Evaluate performance
accuracy = accuracy_score(test_actual_values['irradiat'], predictions)  # Assuming 'irradiat' is the target column
print("Accuracy:", accuracy)
```