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

%matplotlib inline
```

Performing EDA

In [2]:
```
dataset.read_csv("C:\Users\User\Desktop\Lymphography.csv")
```

In [3]:
```
data
```

Out[3]:
```
   1  4  2  1  L1  L2  L3  L4  21  L5  L2  23  23  24  41  8  L6  L7  25  26

0  2  3  2  1   1   2   2   1   2   1   3   3   2   3   4   2   2   2   2   2

1  3  3  2  2   2   2   2   1   2   2   1   3   2   4   3   3   2   2   2   2

2  3  3  1  1   1   1   2   1   2   1   1   3   4   4   4   3   1   2   6   6

3  2  3  1  1   1   1   1   1   1   1   2   3   3   5   3   3   2   2   2   1

4  2  2  1  1   1   1   1   1   2   1   3   3   3   8   3   1   2   2   2   1

...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...

142  3  3  2  1   1   2   2   1   2   1   2   2   2   5   2   2   2   2   4   4

143  2  2  1  1   1   1   1   1   1   1   3   1   1   3   1   1   2   2   1   1

144  3  2  2  1   1   2   2   1   2   1   3   3   3   5   3   2   2   2   2   2

145  2  2  1  1   1   1   1   1   2   1   2   2   2   2   2   2   2   2   1   1

146  2  2  2  1   1   2   2   1   2   1   3   3   3   5   3   2   2   2   2   6

147 rows × 19 columns
```

In [4]:
```
# In above data set column names are not given,we are trying to add column names to dataset
data.columns=['class', 'lymphatics', 'block of affere', 'bl. of lymph. c', 'bl. of lymph. s', 'by pass',
'extravasates', 'regeneration of', 'early uptake in', 'lym.nodes dimin', 'lym.nodes enlar',
'changes in lym.', 'defect in node', 'changes in node', 'changes in stru', 'special forms',
'dislocation of', 'exclusion of no.', 'no. of nodes in']
```

Out[5]:

	class	lymphatics	block of affere	bl. of lymph. c	bl. of lymph. s	by pass	extravasates	regeneration of	early uptake in	lym.nodes dimin	lym.nodes enlar	changes in lym.	defect in node	changes in node	changes in stru	special forms	dislocation of	exclusion of no.	no. of nodes in
0	2	3	2	1	1	2	2	1	2	1	3	3	2	3	4	2	2	2	2
1	3	3	2	2	2	2	2	1	2	2	1	4	3	4	3	3	2	2	2
2	3	3	1	1	1	1	2	1	2	1	1	3	4	4	4	3	1	2	6
3	2	3	1	1	1	1	1	1	1	1	2	3	3	5	3	3	2	2	1
4	2	2	1	1	1	1	1	1	2	1	3	3	3	8	3	1	2	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
142	3	3	2	1	1	2	2	1	2	1	2	2	2	5	2	2	2	2	4
143	2	2	1	1	1	1	1	1	1	1	3	1	1	3	1	1	2	2	1
144	3	2	2	1	1	2	2	1	2	1	3	3	3	5	3	2	2	2	2
145	2	2	1	1	1	1	1	1	2	1	2	2	2	2	2	2	2	2	1
146	2	2	2	1	1	2	2	1	2	1	3	3	3	5	3	2	2	2	6

147 rows × 19 columns

In [6]:
```
# In above data set column names are not given,we are trying to add column names to dataset
data.columns=['class', 'lymphatics', 'block of affere', 'bl. of lymph. c', 'bl. of lymph. s', 'by pass',
'extravasates', 'regeneration of', 'early uptake in', 'lym.nodes dimin', 'lym.nodes enlar',
'changes in lym.', 'defect in node', 'changes in node', 'changes in stru', 'special forms',
'dislocation of', 'exclusion of no.', 'no. of nodes in']
```

In [7]:
```
data
```

Out[7]:

	class	lymphatics	block of affere	bl. of lymph. c	bl. of lymph. s	by pass	extravasates	regeneration of	early uptake in	lym.nodes dimin	lym.nodes enlar	changes in lym.	defect in node	changes in node	changes in stru	special forms	dislocation of	exclusion of no.	no. of nodes in
0	2	3	2	1	1	2	2	1	2	1	3	3	2	3	4	2	2	2	2
1	3	3	2	2	2	2	2	1	2	2	1	4	3	4	3	3	2	2	2
2	3	3	1	1	1	1	2	1	2	1	1	3	4	4	4	3	1	2	6
3	2	3	1	1	1	1	1	1	1	1	2	3	3	5	3	3	2	2	1
4	2	2	1	1	1	1	1	1	2	1	3	3	3	8	3	1	2	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
142	3	3	2	1	1	2	2	1	2	1	2	2	2	5	2	2	2	2	4
143	2	2	1	1	1	1	1	1	1	1	3	1	1	3	1	1	2	2	1
144	3	2	2	1	1	2	2	1	2	1	3	3	3	5	3	2	2	2	2
145	2	2	1	1	1	1	1	1	2	1	2	2	2	2	2	2	2	2	1
146	2	2	2	1	1	2	2	1	2	1	3	3	3	5	3	2	2	2	6

147 rows × 19 columns

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

Out[8]:

	class	lymphatics	block of affere	bl. of lymph. c	bl. of lymph. s	by pass	extravasates	regeneration of	early uptake in	lym.nodes dimin	lym.nodes enlar	changes in lym.	defect in node	changes in node	changes in stru	special forms	dislocation of	exclusion of no.	no. of nodes in
0	2	3	2	1	1	2	2	1	2	1	3	3	2	3	4	2	2	2	2
1	3	3	2	2	2	2	2	1	2	2	1	4	3	4	3	3	2	2	2
2	3	3	1	1	1	1	2	1	2	1	1	3	4	4	4	3	1	2	6
3	2	3	1	1	1	1	1	1	1	1	2	3	3	5	3	3	2	2	1
4	2	2	1	1	1	1	1	1	2	1	3	3	3	8	3	1	2	2	1

In [9]:
```
data.head()
```

Out[9]:

	class	lymphatics	block of affere	bl. of lymph. c	bl. of lymph. s	by pass	extravasates	regeneration of	early uptake in	lym.nodes dimin	lym.nodes enlar	changes in lym.	defect in node	changes in node	changes in stru	special forms	dislocation of	exclusion of no.	no. of nodes in
0	2	3	2	1	1	2	2	1	2	1	3	3	2	3	4	2	2	2	2
1	3	3	2	2	2	2	2	1	2	2	1	4	3	4	3	3	2	2	2
2	3	3	1	1	1	1	2	1	2	1	1	3	4	4	4	3	1	2	6
3	2	3	1	1	1	1	1	1	1	1	2	3	3	5	3	3	2	2	1
4	2	2	1	1	1	1	1	1	2	1	3	3	3	8	3	1	2	2	1

In [10]:
```
data.info()
```

Out[10]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 147 entries, 0 to 146
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   class           147 non-null    int64
 1   lymphatics      147 non-null    int64
 2   block of affere 147 non-null    int64
 3   bl. of lymph. c 147 non-null    int64
 4   bl. of lymph. s 147 non-null    int64
 5   by pass         147 non-null    int64
 6   extravasates    147 non-null    int64
 7   regeneration of 147 non-null    int64
 8   early uptake in 147 non-null    int64
 9   lym.nodes dimin 147 non-null    int64
 10  lym.nodes enlar 147 non-null    int64
 11  changes in lym. 147 non-null    int64
 12  defect in node  147 non-null    int64
 13  changes in node 147 non-null    int64
 14  changes in stru 147 non-null    int64
 15  special forms   147 non-null    int64
 16  dislocation of  147 non-null    int64
 17  exclusion of no 147 non-null    int64
 18  no. of nodes in 147 non-null    int64
dtypes: int64(19)
memory usage: 21.9 KB
```

In [11]:
```
data.describe()
```

Out[11]:

	class	lymphatics	block of affere	bl. of lymph. c	bl. of lymph. s	by pass	extravasates	regeneration of	early uptake in	lym.nodes dimin	lym.nodes enlar	changes in lym.	defect in node	changes in node	changes in stru	s
count	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.000000	147.00
mean	2.448980	2.734694	1.551020	1.170071	1.047619	1.244898	1.510204	1.060027	1.700680	1.061224	2.476190	2.401361	2.972789	2.790816	3.197279	2.3
std	0.679572	0.913916	0.499091	0.383054	0.213867	0.431497	0.501605	0.252003	0.459526	0.240496	0.600866	0.590355	0.807071	0.707960	2.166455	0.7
min	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00
25%	2.000000	2.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	2.000000	2.000000	2.000000	2.000000	4.000000	2.0
50%	2.000000	3.000000	2.000000	1.000000	1.000000	2.000000	2.000000	1.000000	2.000000	1.000000	2.000000	2.000000	3.000000	3.000000	5.000000	3.0
75%	3.000000	3.000000	2.000000	1.000000	1.000000	1.000000	2.000000	1.000000	2.000000	1.000000	3.000000	3.000000	4.000000	3.000000	6.000000	3.0
max	4.000000	4.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	4.000000	3.000000	4.000000	4.000000	8.000000	6.0

In [12]:
```
data['class'].value_counts()
```

Out[12]:
```
2    81
3    49
4    14
1     3
Name: class, dtype: int64
```

In [13]:
```
def count_plot(col):
    sns.countplot(x=data[col])
    plt.title('class count')
    plt.show()
count_plot("class")
```

Out[13]:

In [14]:
```
data.isnull().sum()
```

Out[14]:
```
class             0
lymphatics        0
block of affere   0
bl. of lymph. c   0
bl. of lymph. s   0
by pass           0
extravasates      0
regeneration of   0
early uptake in   0
lym.nodes dimin   0
lym.nodes enlar   0
changes in lym.   0
defect in node    0
changes in node   0
changes in stru   0
special forms     0
dislocation of    0
exclusion of no   0
no. of nodes in   0
dtype: int64
```

In [15]:
```
sns.set_style('whitegrid')
sns.distplot(data['class'],kde=True,color='red')
plt.show()
```

Out[15]:
```
C:\Users\User\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future ve
rsion. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histogram
s).
  warnings.warn(msg, FutureWarning)
```

Imblearn SMOTH algorith to balance the classes

In [16]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
```

In [17]:
```
x = data.drop(columns=['class'])
y = data['class']

x_train,x_test,y_train,y_test = train_test_split(x,y,random_state=100,test_size=0.3,stratify=y)
```

In [18]:
```
model = LogisticRegression()
model.fit(x_train,y_train)
pred = model.predict(x_test)
print('Accuracy',accuracy_score(y_test,pred))
print(classification_report(y_test,pred))
sns.heatmap(confusion_matrix(y_test,pred),cmap,fmt='.2g')
```

Out[18]:
```
Accuracy 0.9111111111111111

              precision    recall  f1-score   support

           1       1.00      1.00      1.00         1
           2       0.92      0.92      0.92        24
           3       0.89      0.89      0.89        18
           4       1.00      1.00      1.00         3

    accuracy                           0.91        45
   macro avg       0.95      0.95      0.95        45
weighted avg       0.91      0.91      0.91        45

C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
<ExecutableDict>
```

Out[18]:

In [19]:
```
np.bincount(y_train)
```

Out[19]:
```
array([0, 1, 56, 42, 3], dtype=int64)
```

In [20]:
```
!pip install imblearn
```

Out[20]:
```
Collecting imblearn
  Downloading imblearn-0.0.py3-none-any.whl (1.9 kB)
Collecting imbalanced-learn
  Downloading imbalanced_learn-0.9.1.py3-none-any.whl (199 kB)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\user\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (2.2.0)
Requirement already satisfied: scikit-learn>=1.1.0 in c:\users\user\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.2.0)
Requirement already satisfied: numpy>=1.17.3 in c:\users\user\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.20.3)
Requirement already satisfied: scipy>=1.8.0 in c:\users\user\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.7.1)
Requirement already satisfied: joblib>=1.0.1 in c:\users\user\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.1.1)
Installing collected packages: scikit-learn, imbalanced-learn, imblearn
  Attempting Uninstall: scikit-learn
    Found existing installation: scikit-learn 0.24.2
    Uninstalling scikit-learn-0.24.2:
      Successfully uninstalled scikit-learn-0.24.2
ERROR: Could not install packages due to an OSError: [WinError 5] Access is denied: 'C:\Users\user\anaconda3\Lib\site-packages\~sklearn\linear_model\_c
ache_coordinate.pyd'
Consider using the `--user` option or check the permissions.
```

In [21]:
```
from tensorflow.keras.layers import Dense ##
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
```

In [22]:
```
features = data.drop(columns=['class'])
target = data['class']
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(features, target)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train)
```

In [23]:
```
model = Sequential()

model.add(Dense(100, input_shape=(features.shape[1],))
model.add(Dense(30, activation='relu'))
# model.add(Dense(45, activation='relu'))
# model.add(Dense(20, activation='relu'))
# model.add(Dense(15, activation='relu'))
model.add(Dense(3, activation='softmax'))
```

In [24]:
```
model.summary()
```

Out[24]:
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               1900

 dense_1 (Dense)             (None, 30)                3030

 dense_2 (Dense)             (None, 3)                 93

=================================================================
Total params: 5,023
Trainable params: 5,023
Non-trainable params: 0
_________________________________________________________________
```

In [25]:
```
import tensorflow
```

In [26]:
```
model.compile(optimizer='sgd',
              loss=tensorflow.keras.losses.CategoricalCrossentropy(),
              metrics=['accuracy'])
```

In [27]:
```
X_train.shape
```

Out[27]:
```
(92, 18)
```

In [28]:
```
y_train.shape
```

Out[28]:
```
(92,)
```

In [29]:
```
y_pred = model.predict(X_test)
```

Out[29]:
```
2/2 [==============================] - 3s 38ms/step
```

In [30]:
```
y_pred
```

Out[30]:
```
array([[0.294396 , 0.351614 , 0.327932 ],
       [0.295395 , 0.350622 , 0.353983 ],
       [0.295395 , 0.350622 , 0.353983 ],
       [0.137869 , 0.358461 , 0.503668 ],
       [0.317248 , 0.52495 , 0.157552 ],
       [0.327139 , 0.567513 , 0.104967 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.212589 , 0.546288 , 0.241098 ],
       [0.096373 , 0.594091 , 0.309454 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.293881 , 0.451356 , 0.252628 ],
       [0.647123 , 0.092925 , 0.262002 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.210803 , 0.55135 , 0.241777 ],
       [0.159165 , 0.575776 , 0.265114 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.147367 , 0.565206 , 0.287427 ],
       [0.172746 , 0.461149 , 0.364104 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.159165 , 0.575776 , 0.265114 ],
       [0.163908 , 0.331738 , 0.504145 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.152638 , 0.552963 , 0.294399 ],
       [0.096463 , 0.503076 , 0.376355 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.162543 , 0.749905 , 0.156026 ],
       [0.142721 , 0.597346 , 0.259814 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.169619 , 0.525036 , 0.305345 ],
       [0.163631 , 0.585257 , 0.207741 ],
       [0.140196 , 0.657765 , 0.202039 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.137128 , 0.573017 , 0.289618 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.152245 , 0.597431 , 0.250324 ],
       [0.189149 , 0.652748 , 0.158102 ],
       [0.294396 , 0.351614 , 0.327932 ],
       [0.096578 , 0.771954 , 0.131468 ]], dtype=float32)
```

Tune the Model using GridSearchCv

In [31]:
```
!pip install pipeline
```

Out[31]:
```
Collecting pipeline
  Downloading pipeline-0.1.0.py3-none-any.whl (2.6 kB)
Installing collected packages: pipeline
Successfully installed pipeline-0.1.0
```

In [32]:
```
from sklearn.model_selection import GridSearchCV
```

In [33]:
```
parameters={'learning_rate':[0.1,0.15,0.20,0.25,0.3],
            'gamma':[0,0.1,0.2,0.3,0.4]}
```

In [34]:
```
# mr=classifier(estimator=model, param_grid=parameters, scoring='neg_log_loss',cv=10, n_jobs=-1)
# mr=search.fit(x_train,y_train)
```

Plot the Accuracy and Loss graph

In [35]:
```
X = data.drop(columns=['class'])
y = data['class']
model.compile(loss='sparse',
              optimizer='adam',
              metrics=['accuracy'])

history = model.fit(X,y,steps_per_epoch=5,
                    validation_split=0.2)
print(history.history.keys())
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

Out[35]:
```
Epoch 1/3
5/5 [==============================] - 1s 100ms/step - loss: 4.6272 - accuracy: 0.1088
Epoch 2/3
5/5 [==============================] - 0s 9ms/step - loss: 4.6224 - accuracy: 0.0816
Epoch 3/3
5/5 [==============================] - 0s 9ms/step - loss: 4.6147 - accuracy: 0.1073
dict_keys(['loss', 'accuracy'])
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<AppdataLocal\Temp\ipykernel_1876/1915254886.py in <module>
     19 print(history.history.keys())
---> 20 plt.plot(history.history['accuracy'])
     21 plt.plot(history.history['val_accuracy'])
     22 plt.title('model accuracy')
KeyError: 'val_accuracy'
```