

PNEUMONIA AND LUNG OPACITY CLASSIFICATION



PRESENTED BY
AISHWARYA G CB.EN.P2AID19002
POOJA SRI R CB.EN.P2AID19022



Pneumonia Opacity Classification

RSNA PNEUMONIA DETECTION
CHALLENGE DATASET





INTRODUCTION

- The main idea of the project is to find the areas affected by pneumonia and classify the patients based on their number of infected regions ie, Lung Opacities.
- We had only the patient datas that had pneumonia, hence we formulated the project to classify the patients based on the lung opacity regions present in their infected lungs.
- This implies that the classification problem is modeled to classes with acute pneumonia and severe pneumonia based on the area of the affected region.



FEATURE EXTRACTION

As suggested by the competition host RSNA and MD.ai, the feature extraction using Mask RCNN was carried out and the results was submitted to kaggle to check accuracy.

The method that involved general CNN didn't provide great accuracy, while the model that used Mask RCNN along with Gaussian Blur for Image Augmentation provided greater accuracy. That was verified using Kaggle Submissions provided in the screenshots given in next slide

CNN Segmentation + connected components 320x320

← → ↺

🔒 kaggle.com/aishuganesh22/cnn-segmentation-connected-components-320x320

☆ 🏠 📌 a ⋮

☰ kaggle

🔍 Search

🔔 🦅

🏠 Home

🏆 Compete

📁 Data

🔗 Notebooks

💬 Discuss

🎓 Courses

⌵ More

Recently Viewed

🦅 CNN Segmentation + c...

🦅 Mask-RCNN with sub...

RSNA Pneumonia Dete...

👤 Diabetic Retinopathy u...

👤 Retinopathy Classificat...

🦅

CNN Segmentation + connected components 320x320

Python notebook using data from [RSNA Pneumonia Detection Challenge](#) · 12 views · 21h ago · 🖨️ gpu

✎ Edit tags

⬆️ 0

🔒 Sharing

Best Submission

✓ Successful

Submitted by aishug 18 minutes ago

Private Score

0.10014

Public Score

0.07465

Version 1 of 1

forked from CNN Segmentation + connected components 320x320

Notebook

Approach

Network

Load Pneumonia Locations

Load Filenames

Input (1)

Approach

- Firstly a convolutional neural network is used to segment the image, using the bounding boxes directly as a mask.
- Secondly connected components is used to separate multiple areas of predicted pneumonia.
- Finally a bounding box is simply drawn around every connected component.

Network

cnn-segmentati...ipynb

results.zip

Show all

×

Type here to search

🔍

📁

🛒

e

📁

📧

📧

🔥

🌐

15:10

05-06-2020

ENG

🔊

🔌

⬆️

🗨️

Mask-RCNN with submission

← → ↺ 🏠

🔒 https://www.kaggle.com/aishwaryatanyaa/mask-rcnn-with-submission/output

⋮ 🛡️ ⭐

⬇️ 📄 📷 📺

⋮

☰ kaggle

🔍 Search

🔔 🦉

🔗 🏆 📅 <> 💬 🎓 ⌵

🦉

Mask-RCNN with submission

Python notebook using data from [RSNA Pneumonia Detection Challenge](#) · 10 views · 5h ago · 🖨️ gpu 🛠️ Edit tags

⬆️ 0

🔒 Sharing

Best Submission

✔️ Successful

Submitted by Aishwarya Ganesh 6 minutes ago

Private Score

0.00396

Public Score

0.00000

Version 2 of 2

forked from Mask-RCNN with submission

Notebook

⬆️ ⬇️

Input (1)

Output

Execution Info

Log

Comments (0)

Mask-RCNN Sample Starter Model for the RSNA Pneumonia Detection Challenge

MD.ai. The dataset for this challenge, created on the MD.ai platform in collaboration with the Radiological Society of North America (RSNA), the Society of Thoracic Radiology (STR), the US National Institutes of Health (NIH), and Kaggle. This notebook covers the basics of parsing the competition dataset, training using a detector based on the [Mask-RCNN algorithm](#) for object detection and instance segmentation.

Note that the Mask-RCNN detector configuration parameters have been selected to reduce training time for demonstration purposes, they are not optimal.

This is based on our deep learning for medical imaging lessons:

- Lesson 1. Classification of chest vs. adominal X-rays using TensorFlow/Keras [Github Annotator](#)
- Lesson 2. Lung X-Rays Semantic Segmentation using UNets. [Github Annotator](#)

https://www.kaggle.com/aishwaryatanyaa

🖥️ 🔍 Type here to search

🔍 📁 📧 📧 📧 📧 📧 📧

15:10 05-06-2020

⬆️ 📶 🔊 ENG

Home

Compete

Data

Notebooks

Discuss

Courses

More

stage_2_detailed_cla...

stage_2_sample_sub...

stage_2_train_labels.c...

Recently Viewed

RSNA Pneumonia Dete...

Mask-RCNN with sub...

Creating Kaggle Accou...

multiple accounts

RSNA Stage 2 Anchor ...

Output

2.04 GB

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

pneumonia20200605T045...

submission.csv

submission.csv (113.34 KB)

Submit

1482acb5-2d2d-48e7-a953-4f1cd72fd37a

21ef32bf-62fb-4868-b7a4-f29b21a5ede4

26cb0d8a-d50a-4de8-92af-78d46355725c

241008d0-9269-4c06-b905-074c6ad6c84c

04b196dc-81cf-45f4-9180-e6766614599f

0.98 196.0 444.0 248.0 312.0

23e99d2f-b0c5-4191-a5fe-8bcece4edd5b

24df3a5d-14be-4d97-bb7c-96c274141e6c

305693d4-6acb-4bf0-90be-4ff2a218689f

21c8c4af-d6cc-45d8-adf2-0f975d79108d

Version 2 of 2

forked from Mask-RCNN with submission

Notebook

Input (1)

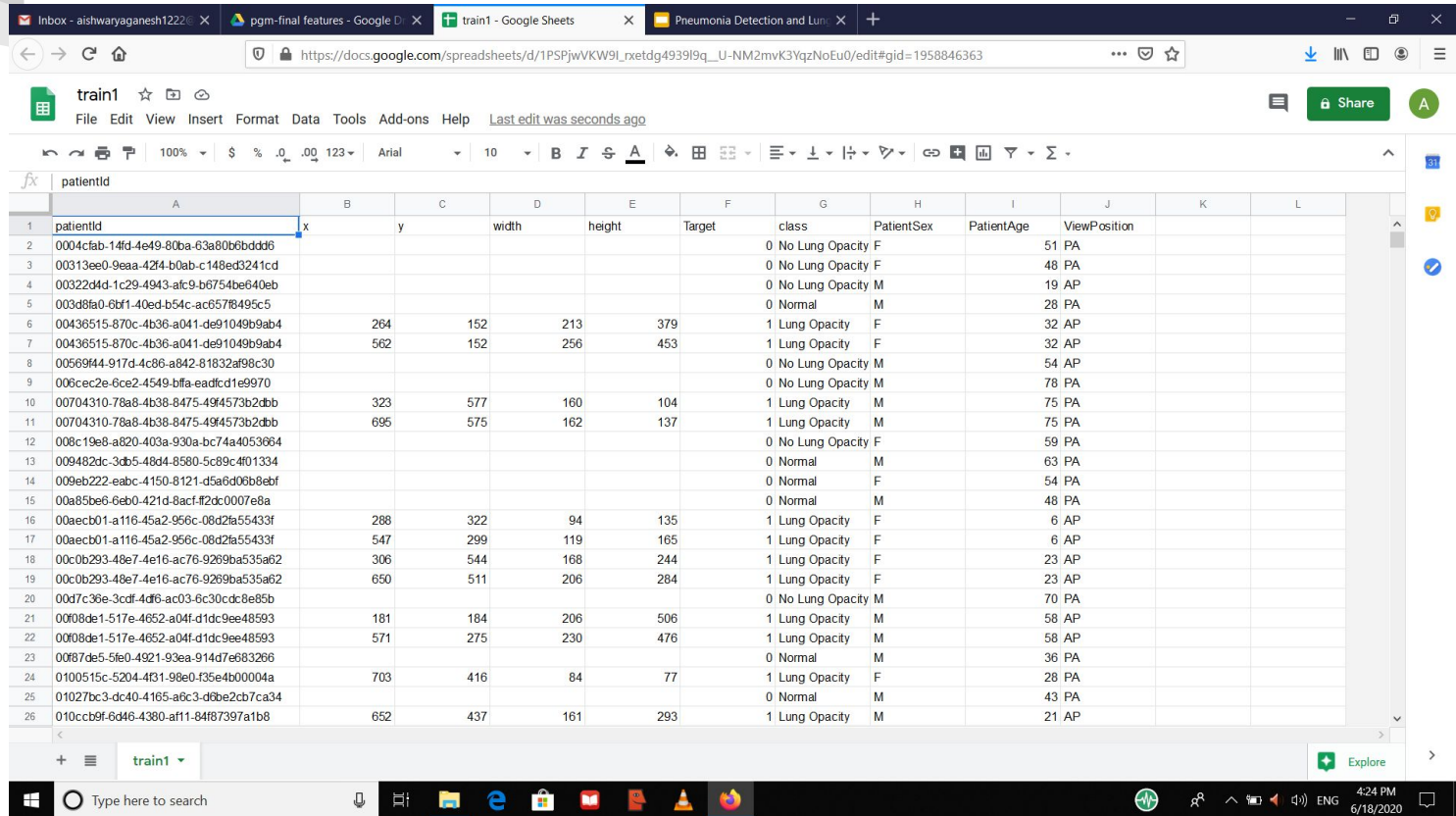
Output

Execution Info

Log

Comments (0)

Initial features(Given) + Features from MetaData(Sex, Age,VP included)



patientId	A	B	C	D	E	F	G	H	I	J	K	L
1	patientId	x	y	width	height	Target	class	PatientSex	PatientAge	ViewPosition		
2	0004cfab-14fd-4e49-80ba-63a80b6bddd6						0 No Lung Opacity	F	51	PA		
3	00313ee0-9eaa-42f4-b0ab-c148ed3241cd						0 No Lung Opacity	F	48	PA		
4	00322d4d-1c29-4943-afc9-b6754be640eb						0 No Lung Opacity	M	19	AP		
5	003d8a0-6bf1-40ed-b54c-ac6578495c5						0 Normal	M	28	PA		
6	00436515-870c-4b36-a041-de91049b9ab4	264	152	213	379	1 Lung Opacity	F	32	AP			
7	00436515-870c-4b36-a041-de91049b9ab4	562	152	256	453	1 Lung Opacity	F	32	AP			
8	00569M44-917d-4c86-a842-81832af98c30					0 No Lung Opacity	M	54	AP			
9	006cec2e-6ce2-4549-bffa-ea4dcd1e9970					0 No Lung Opacity	M	78	PA			
10	00704310-78a8-4b38-8475-49f4573b2dbb	323	577	160	104	1 Lung Opacity	M	75	PA			
11	00704310-78a8-4b38-8475-49f4573b2dbb	695	575	162	137	1 Lung Opacity	M	75	PA			
12	008c19e8-a820-403a-930a-bc74a053664					0 No Lung Opacity	F	59	PA			
13	009482dc-3db5-48d4-8580-5c89c4f01334					0 Normal	M	63	PA			
14	009eb222-eabc-4150-8121-d5a6d00b8ebf					0 Normal	F	54	PA			
15	00a85be6-6eb0-421d-8acf-#2dc0007e8a					0 Normal	M	48	PA			
16	00aebc01-a116-45a2-956c-08d2fa55433f	288	322	94	135	1 Lung Opacity	F	6	AP			
17	00aebc01-a116-45a2-956c-08d2fa55433f	547	299	119	165	1 Lung Opacity	F	6	AP			
18	00c0b293-48e7-4e16-ac76-9269ba535a62	306	544	168	244	1 Lung Opacity	F	23	AP			
19	00c0b293-48e7-4e16-ac76-9269ba535a62	650	511	206	284	1 Lung Opacity	F	23	AP			
20	00d7c36e-3cdf-4df6-ac03-6c30cdc8e85b					0 No Lung Opacity	M	70	PA			
21	00f08de1-517e-4652-a04f-d1dc9ee48593	181	184	206	506	1 Lung Opacity	M	58	AP			
22	00f08de1-517e-4652-a04f-d1dc9ee48593	571	275	230	476	1 Lung Opacity	M	58	AP			
23	00f87de5-5fe0-4921-93ea-914d7e683266					0 Normal	M	36	PA			
24	0100515c-5204-4f31-99e0-f35e4b00004a	703	416	84	77	1 Lung Opacity	F	28	PA			
25	01027bc3-dc40-4165-a6c3-d8be2cb7ca34					0 Normal	M	43	PA			
26	010ccb9f-6d46-4380-af11-84f87397a1b8	652	437	161	293	1 Lung Opacity	M	21	AP			



Dataset Observation

As we can see from the given dataset, the problem is to identify the lung opacity regions and not to classify the data into pneumonia and normal classes.

Hence that part of classification is done using another dataset Chest X-ray from Kaggle(explained later).

Now for this RSNA dataset, the problem is formulated to find the number of lung opacity regions present in a patient x ray. A bayesian approach is provided for that obtained features by grouping the number of opacity regions as Region of Interest as 0 if less opacity regions and 1 if there are more than 2 opacity region .

Windows taskbar showing search bar, taskbar icons (File Explorer, Edge, Mail, etc.), and system tray (clock, date, volume, network).

After processing and grouping the values in features

pyagram-bn-classifier

File Edit View Insert Format Data Tools Add-ons Help Last edit was made 4 minutes ago by anonymous

	A	B	C	D	E	F	G	H	I	J	K
1		num_rois	rois_area_avg	age	gender	has_outliers	high_black_pixel	high_white_pixel	ViewPosition		
2	0	1	1	adult		0	0	0	0		
3	1	1	1	adult		0	0	0	0		
4	2	1	1	adult		1	0	0	0		
5	3	1	1	adult		1	0	0	0		
6	4	1	1	kid		0	0	0	0		
7	5	1	1	kid		0	0	0	0		
8	6	1	1	adult		0	0	0	0		
9	7	1	1	adult		0	0	0	0		
10	8	1	1	adult		1	0	0	0		
11	9	1	1	adult		1	0	0	0		
12	10	0	1	adult		0	0	0	0		
13	11	1	1	adult		1	0	0	0		
14	12	1	1	adult		1	0	0	0		
15	13	1	1	adult		1	0	0	0		
16	14	1	1	adult		1	0	0	0		
17	15	1	1	adult		1	0	0	0		
18	16	1	1	adult		1	0	0	0		
19	17	1	1	adult		0	0	0	0		
20	18	1	1	adult		0	0	0	0		
21	19	1	1	adult		0	0	0	0		
22	20	1	1	adult		0	0	0	0		
23	21	1	1	adult		0	0	0	0		
24	22	1	1	adult		0	0	0	0		
25	23	0	1	adult		0	0	0	0		
26	24	1	1	adult		1	0	0	0		

pyagram-bn-classifier

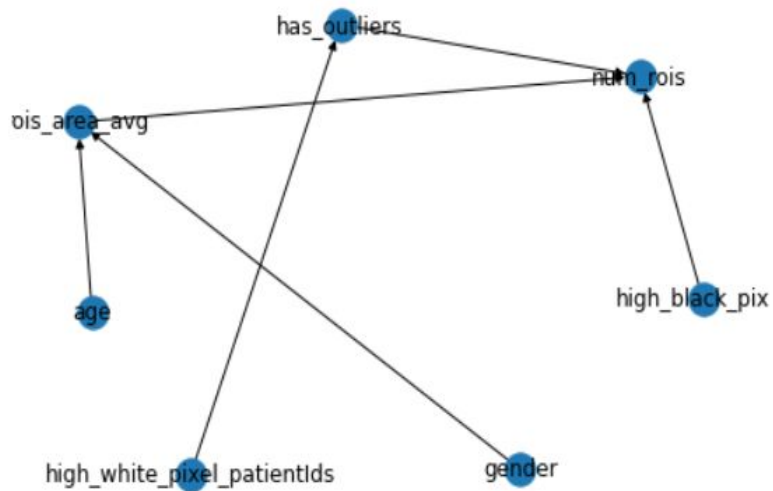
Explore

Type here to search

4:41 PM 6/18/2020



BAYESIAN MODELLING USING PYGMMPY



```
from pgmpy.inference import VariableElimination
LungDisease_infer = VariableElimination(model)

q = LungDisease_infer.query(variables=['age'], evidence={'num_rois': 0})
print(q)
```

```
Finding Elimination Order: : 100%|██████████| 5/5 [00:00<00:00, 2519.40it/s]
Eliminating: has_outliers: 100%|██████████| 5/5 [00:00<00:00, 431.28it/s] +-----+
| age          | phi(age) |
+=====+
| age(adult)   | 0.8903  |
+-----+
| age(kid)     | 0.0280  |
+-----+
| age(old)     | 0.0125  |
+-----+
| age(teen)    | 0.0649  |
+-----+
| age(toddler) | 0.0042  |
+-----+
```

File Edit View Insert Runtime Tools Help

+ Code + Text

[] model.fit(LungDisease)

[] y_pred = model.predict(predict_data)

100% | 48/48 [00:00<00:00, 236.99it/s]

▶ y_pred.head(10)

	num_rois
0	1
1	0
2	0
3	0
4	0
5	0
6	0
7	1
8	1
9	0

[] #print('Accuracy is'+ str(counter/LungDisease.shape[0]*100))

[] from sklearn.metrics import accuracy_score, confusion_matrix

[] print ('Accuracy Score :', accuracy_score(actual, y_pred)*100)

Accuracy Score : 62.95655932569699

[] print(confusion_matrix(actual, y_pred))

[5270 1671]
[3471 3469]

[]

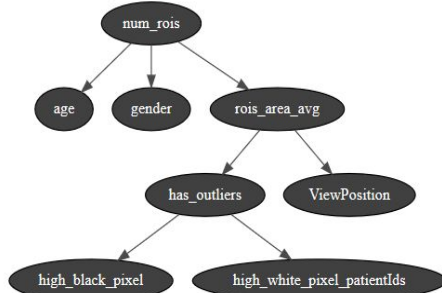
BAYESIAN MODELLING USING PYAGRUM - Model Created

Create Bayesian Network

```
bn = gum.BayesNet("Pneumonia Detection")
bn = gum.fastBN("Age(baby|toddler|kid|teen|adult|old)<-num_rois{True|False}>->gender{0|1};high_black_pixel{0|1}<-has_outliers{0|1}>->high_white_pixel_patientIds{0|1};has_outliers{0|1}<-rois_area_avg{0|1};ViewPosition{0|1}<-rois_area_avg{0|1}")
print(bn.variable("num_rois"))
print(bn.variable("age"))
print(bn.variable("gender"))
print(bn.variable("rois_area_avg"))
print(bn.variable("has_outliers"))
print(bn.variable("high_black_pixel"))
print(bn.variable("high_white_pixel_patientIds"))
print(bn.variable("ViewPosition"))

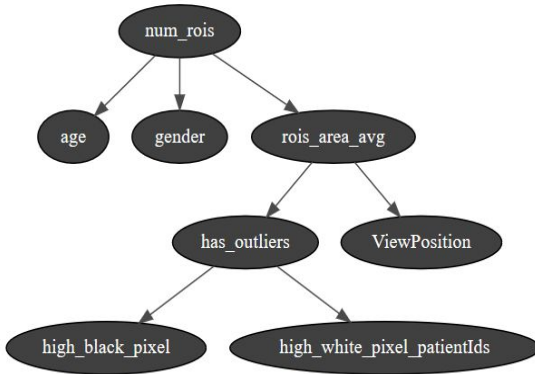
bn
```

```
num_rois<True,False>
age<baby,toddler,kid,teen,adult,old>
gender<0,1>
rois_area_avg<0,1>
has_outliers<0,1>
high_black_pixel<0,1>
high_white_pixel_patientIds<0,1>
ViewPosition<0,1>
```

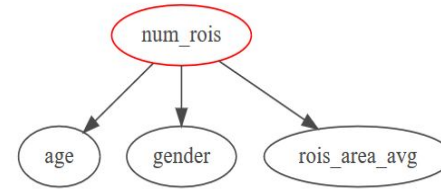


Markov Blanket of Created Network

```
gnb.sideBySide(bn, gum.MarkovBlanket(bn, 'num_rois'), captions=["Learned Bayesian Network", "Markov blanket of 'num_rois'"])
```

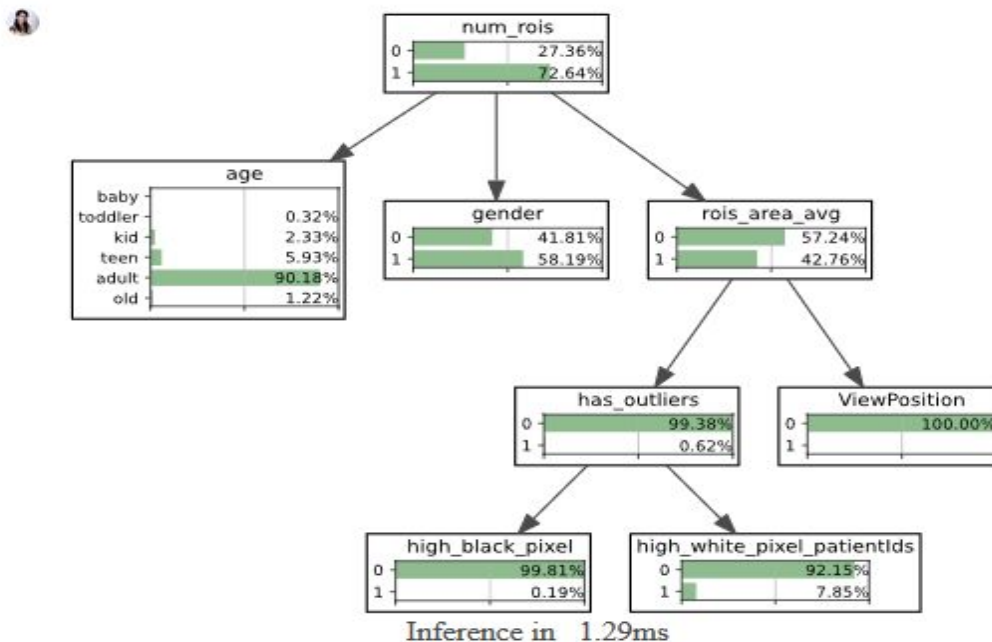


Learned Bayesian Network



Markov blanket of 'num_rois'

Inference when View Position is AP for our designed network



Accuracy: 72.15%

Bayesian PyAgrum.ipynb - Colab

```
[ ] def update_beliefs(engine, bayesNet, row):
    # Update beliefs from a given row less the Survived variable
    for var in bayesNet.names():
        if var == "num_roids":
            continue
        try:
            label = str(row.to_dict()[var])
            idx = bayesNet.variable(var).index(str(row.to_dict()[var]))
            engine.chgEvidence(var, idx)
        except gum.NotFound:
            # this can happen when value is missing in the test base.
            pass
    engine.makeInference()

def is_well_predicted(engine, bayesNet, auc, row):
    update_beliefs(engine, bayesNet, row)
    marginal = engine.posterior('num_roids')
    outcome = row.to_dict()['num_roids']
    if outcome == 1: # Did not survived
        if marginal.toarray()[1] > auc:
            return "True Positive"
        else:
            return "False Negative"
    else: # Survived
        if marginal.toarray()[1] <= auc:
            return "True Negative"
        else:
            return "False Positive"

ie = gum.LazyPropagation(bn)
init_belief(ie)
ie.addTarget('num_roids')
result = traindf.apply(lambda x: is_well_predicted(ie, bn, 0.62, x), axis=1)
result.value_counts(True)

True Positive    0.717530
False Positive   0.269597
False Negative   0.008896
True Negative    0.003977
dtype: float64

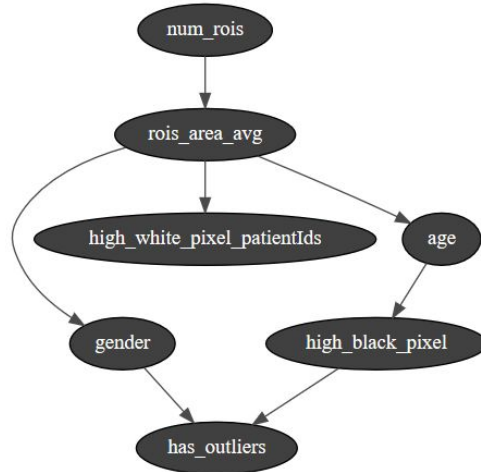
[ ]

positives = sum(result.map(lambda x: 1 if x.startswith("True") else 0))
total = result.count()
print("{0:.2f}% good predictions".format(positives/total*100))

72.15% good predictions
```

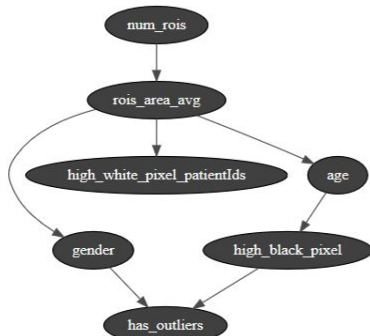
Model the package suggests (Observation: It uses Hill Climbing Algorithm)

```
] traindf.to_csv("/content/drive/My Drive/pgm-final features/pyagrum-bn-classifier.csv")  
  
] file = "/content/drive/My Drive/pgm-final features/pyagrum-bn-classifier.csv"  
  learner = gum.BNLearner(file, template)  
  bn = learner.learnBN()  
  bn
```

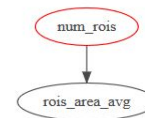


Markov Blanket of Suggested Network

```
mb.sideBySide(bn, gum.MarkovBlanket(bn, 'num_rois'), captions=["Learned Bayesian Network", "Markov blanket of 'Number of ROI's'"])
```

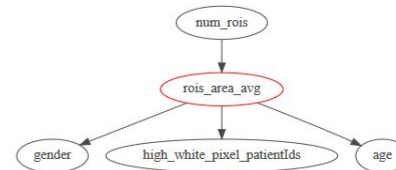
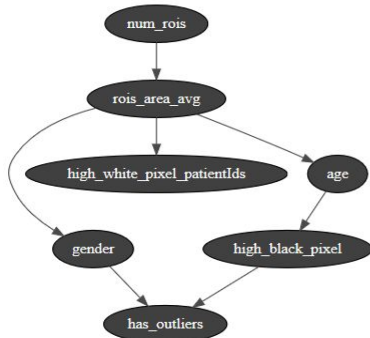


Learned Bayesian Network



Markov blanket of 'Number of ROI's'

```
mb.sideBySide(bn, gum.MarkovBlanket(bn, 'rois_area_avg'), captions=["Learned Bayesian Network", "Markov blanket of 'Average of ROI'"])
```



```
Using BN Classifier

[ ] import pyAgrum.lib.classifier as BNClassifier

[ ] from sklearn.model_selection import train_test_split

[ ] targetColumn = 'num_rois'

X = traindf.drop(targetColumn, axis=1)
y = traindf[targetColumn]

[ ] x_train_df,x_test_df,y_train_df,y_test_df=train_test_split(X,y,test_size=0.3,random_state=10)

[ ] bn = BNClassifier.BNClassifier()
bn.fit(x_train_df, y_train_df)
y_test_pred = bn.predict(x_test_df)

print("{0:.2f}% good predictions".format(accuracy_score(y_test_df, y_test_pred)*100))

57.94% good predictions

[ ] bn_prior = BNClassifier.BNClassifier(prior='laplace')
bn_prior.fit(x_train_df, y_train_df)
y_test_pred_prior = bn_prior.predict(x_test_df)

print("{0:.2f}% good predictions".format(accuracy_score(y_test_df, y_test_pred_prior)*100))

59.78% good predictions
```

Pneumonia prediction in Chest X ray images





INTRODUCTION:

- The model proposed is used to classify whether the person has pneumonia or normal lung.
- The steps involved in this work is
 - Principal component analysis
 - Histogram classifier
 - Bayesian classifier
- PCA is used for dimensionality reduction in the image.
- On the dimension reduced image we perform histogram and bayesian classification.



DATASET DESCRIPTION:

- Dataset that we used is a kaggle challenge dataset.
- It consists of 5216 images.
- Out of that 1341 are normal lung images and 3875 are pneumonia lung images.
- The image format is jpeg.



BAYESIAN CLASSIFIER:

- A Bayesian classifier is based on the idea that the role of a class is to predict the values of features for members of that class.
- In a Bayesian classifier, the learning agent builds a probabilistic model of the features and uses that model to predict the classification of a new example.
- The simplest case is the naive Bayesian classifier, which makes the independence assumption that the input features are conditionally independent of each other given the classification.



PARAMETERS USED IN BAYESIAN MODEL:

- `mun` -mean vector of normal
- `mup` -mean vector of pneumonia
- `cn`-convolution matrix of normal
- `cp`-convolution matrix of pneumonia
- `labeln`-unique value i.e zero
- `labelp`-unique value i.e one(pneumonia)
- `Nn`-0 existed time
- `Np`-1 existed time

```
def Bayesian2DClassifier(queries, Nn, Np, mun, mup, cn, cp, labeln, labelp):
    w1 = 1
    w2 = 1
    A = w1*w2
    N = np.alen(queries)
    [countn, countp] = np.zeros((2, N))
    factorn = Nn * A * (1 / (2 * np.pi * np.sqrt(np.linalg.det(cn))))
    factorp = Np * A * (1 / (2 * np.pi * np.sqrt(np.linalg.det(cp))))
    icn = np.linalg.inv(cn)
    icp = np.linalg.inv(cp)
    for i, q in enumerate(queries):
        countn[i] = factorn*np.exp(-0.5 * np.dot(np.dot(q - mun, icn), q - mun))
        countp[i] = factorp*np.exp(-0.5 * np.dot(np.dot(q - mup, icp), q - mup))
    resultlabel = np.full(N, 999, dtype = int)
    indicesn = countn > countp
    indicesp = countp > countn
    resultlabel[indicesn] = labeln
    resultlabel[indicesp] = labelp
    resultprob = countn/(countn+countp)
    resultprob[indicesp] = 1 - resultprob[indicesp]
    return resultlabel, resultprob
```



BAYESIAN CLASSIFIER ACCURACY:

- As a result we got 74.29% accuracy rate with the help of bayesian model.

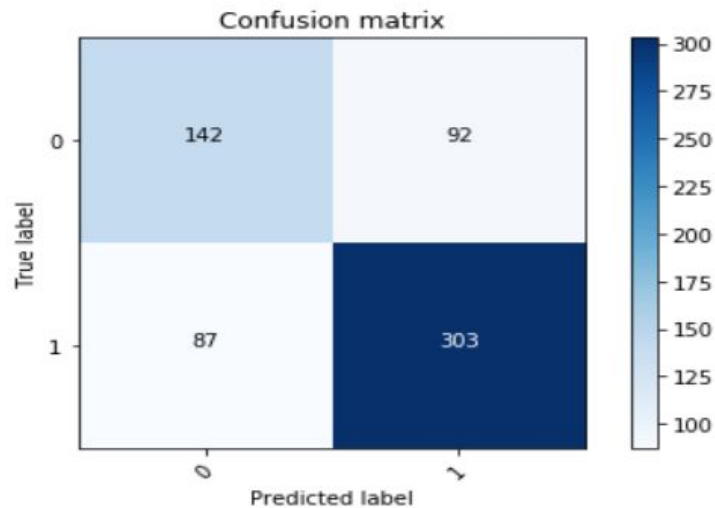
```
[ ] accuracyB = accuracy(T1, Blabels)*100  
accuracyB
```



74.29064417177914



CONFUSION MATRIX:





THANK YOU