## Group 27

**Team Members:**

**1.P.Sai Charan Yadav -S20200010166**
**2.S.Rohit Kumar -S20200010201**

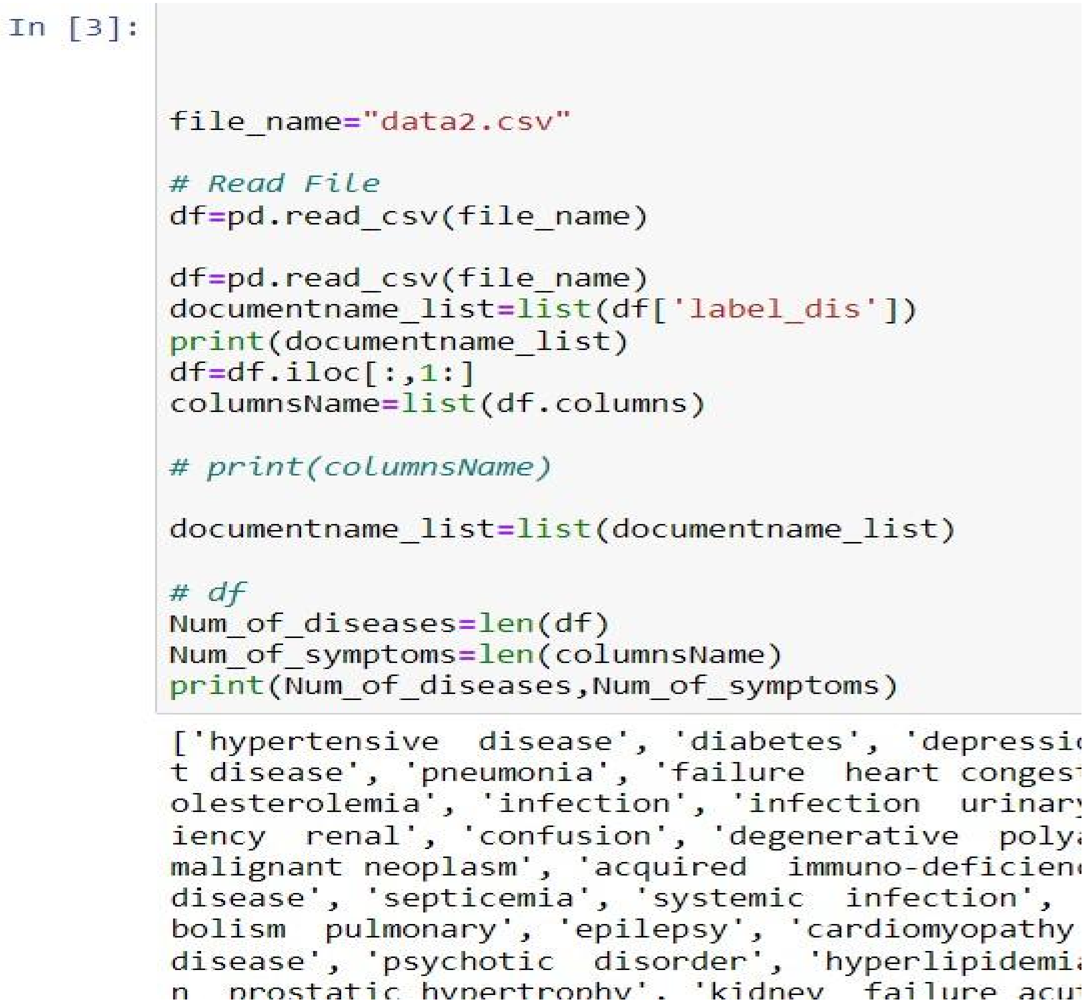
# Topic Name:

“Disease Symptom Prediction and its Data Analytics ”

## Involved Components:

1)Indexing
2)Searching Component
3)Refining
4)Feedback
5)Assessment Components

# 1)Indexing:

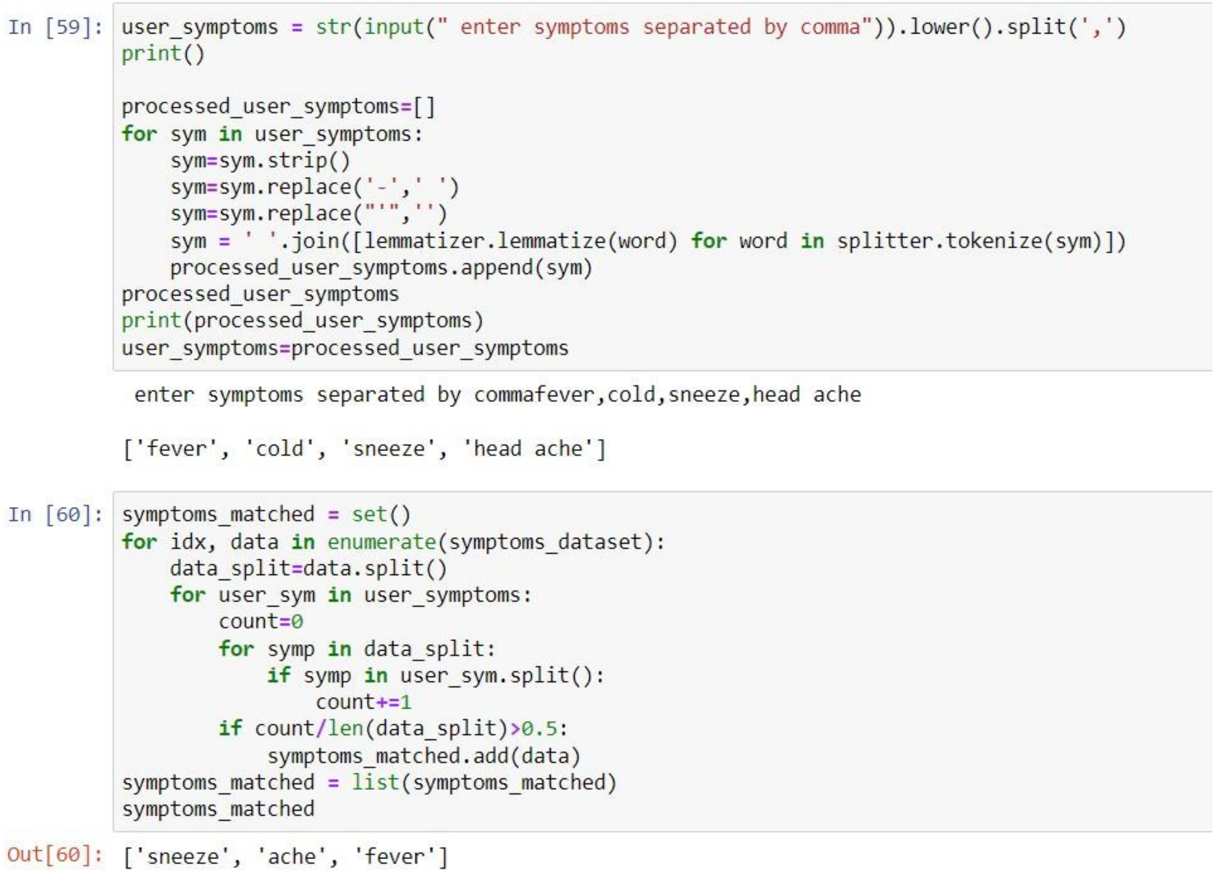


```
file_name="data2.csv"

# Read File
df=pd.read_csv(file_name)

df=pd.read_csv(file_name)
documentname_list=list(df['label_dis'])
print(documentname_list)
df=df.iloc[:,1:]
columnsName=list(df.columns)

# print(columnsName)

documentname_list=list(documentname_list)

# df
Num_of_diseases=len(df)
Num_of_symptoms=len(columnsName)
print(Num_of_diseases,Num_of_symptoms)
```

```
['hypertensive  disease', 'diabetes', 'depressi
t disease', 'pneumonia', 'failure  heart conges
olesterolemia', 'infection', 'infection  urinar
iency  renal', 'confusion', 'degenerative  poly
malignant neoplasm', 'acquired  immuno-deficien
disease', 'septicemia', 'systemic  infection',
bolism  pulmonary', 'epilepsy', 'cardiomyopathy
disease', 'psychotic  disorder', 'hyperlipidemi
n  prostatic hypertrophy', 'kidney  failure acu
```

We took Indexed data online and found the TF, IDF, and TF_IDF score.

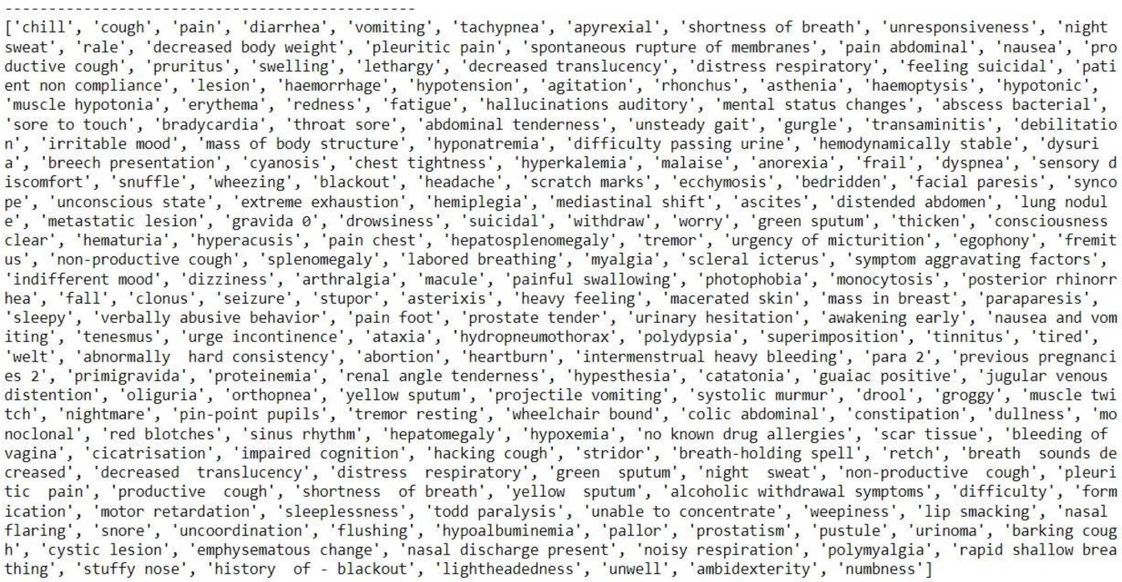
# 2)Searching Component



```
In [59]: user_symptoms = str(input(" enter symptoms separated by comma")).lower().split(',')
         print()

         processed_user_symptoms=[]
         for sym in user_symptoms:
             sym=sym.strip()
             sym=sym.replace('-',' ')
             sym=sym.replace("'",'')
             sym = ' '.join([lemmatizer.lemmatize(word) for word in splitter.tokenize(sym)])
             processed_user_symptoms.append(sym)
         processed_user_symptoms
         print(processed_user_symptoms)
         user_symptoms=processed_user_symptoms

          enter symptoms separated by commafever,cold,sneeze,head ache

         ['fever', 'cold', 'sneeze', 'head ache']

In [60]: symptoms_matched = set()
         for idx, data in enumerate(symptoms_dataset):
             data_split=data.split()
             for user_sym in user_symptoms:
                 count=0
                 for symp in data_split:
                     if symp in user_sym.split():
                         count+=1
                 if count/len(data_split)>0.5:
                     symptoms_matched.add(data)
         symptoms_matched = list(symptoms_matched)
         symptoms_matched

Out[60]: ['sneeze', 'ache', 'fever']
```

Enter the query in comma-separated values,then the list gets matched with the symptoms dataset and returns the symptoms_matched list

# 3)Refining Searches

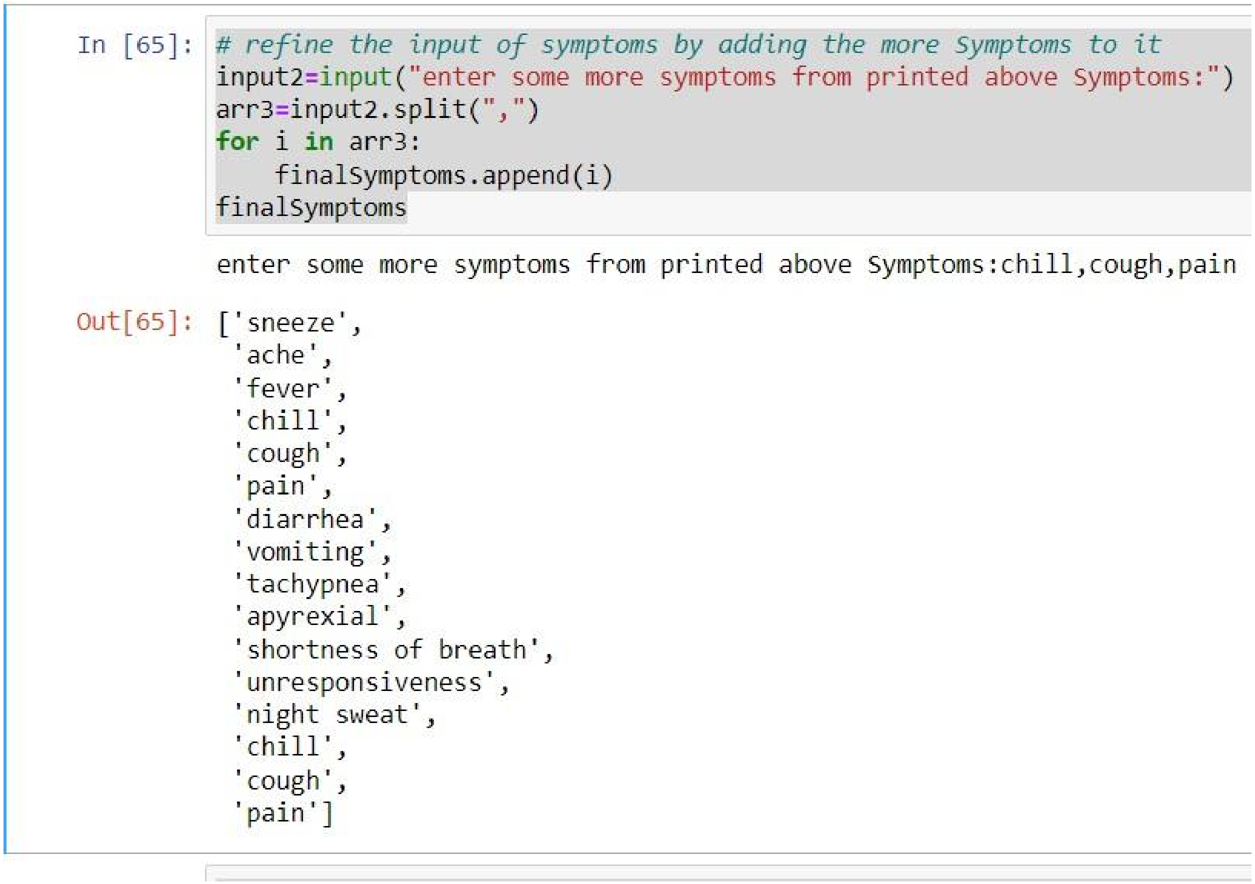


```
-----------------------------------------------
['chill', 'cough', 'pain', 'diarrhea', 'vomiting', 'tachypnea', 'apyrexial', 'shortness of breath', 'unresponsiveness', 'night
sweat', 'rale', 'decreased body weight', 'pleuritic pain', 'spontaneous rupture of membranes', 'pain abdominal', 'nausea', 'pro
ductive cough', 'pruritus', 'swelling', 'lethargy', 'decreased translucency', 'distress respiratory', 'feeling suicidal', 'pati
ent non compliance', 'lesion', 'haemorrhage', 'hypotension', 'agitation', 'rhonchus', 'asthenia', 'haemoptysis', 'hypotonic',
'muscle hypotonia', 'erythema', 'redness', 'fatigue', 'hallucinations auditory', 'mental status changes', 'abscess bacterial',
'sore to touch', 'bradycardia', 'throat sore', 'abdominal tenderness', 'unsteady gait', 'gurgle', 'transaminitis', 'debilitatio
n', 'irritable mood', 'mass of body structure', 'hyponatremia', 'difficulty passing urine', 'hemodynamically stable', 'dysuri
a', 'breech presentation', 'cyanosis', 'chest tightness', 'hyperkalemia', 'malaise', 'anorexia', 'frail', 'dyspnea', 'sensory d
iscomfort', 'snuffle', 'wheezing', 'blackout', 'headache', 'scratch marks', 'ecchymosis', 'bedridden', 'facial paresis', 'synco
pe', 'unconscious state', 'extreme exhaustion', 'hemiplegia', 'mediastinal shift', 'ascites', 'distended abdomen', 'lung nodul
e', 'metastatic lesion', 'gravida 0', 'drowsiness', 'suicidal', 'withdraw', 'worry', 'green sputum', 'thicken', 'consciousness
clear', 'hematuria', 'hyperacusis', 'pain chest', 'hepatosplenomegaly', 'tremor', 'urgency of micturition', 'egophony', 'fremit
us', 'non-productive cough', 'splenomegaly', 'labored breathing', 'myalgia', 'scleral icterus', 'symptom aggravating factors',
'indifferent mood', 'dizziness', 'arthralgia', 'macule', 'painful swallowing', 'photophobia', 'monocytosis', 'posterior rhinorr
hea', 'fall', 'clonus', 'seizure', 'stupor', 'asterixis', 'heavy feeling', 'macerated skin', 'mass in breast', 'paraparesis',
'sleepy', 'verbally abusive behavior', 'pain foot', 'prostate tender', 'urinary hesitation', 'awakening early', 'nausea and vom
iting', 'tenesmus', 'urge incontinence', 'ataxia', 'hydropneumothorax', 'polydypsia', 'superimposition', 'tinnitus', 'tired',
'welt', 'abnormally  hard consistency', 'abortion', 'heartburn', 'intermenstrual heavy bleeding', 'para 2', 'previous pregnanci
es 2', 'primigravida', 'proteinemia', 'renal angle tenderness', 'hypesthesia', 'catatonia', 'guaiac positive', 'jugular venous
distention', 'oliguria', 'orthopnea', 'yellow sputum', 'projectile vomiting', 'systolic murmur', 'drool', 'groggy', 'muscle twi
tch', 'nightmare', 'pin-point pupils', 'tremor resting', 'wheelchair bound', 'colic abdominal', 'constipation', 'dullness', 'mo
noclonal', 'red blotches', 'sinus rhythm', 'hepatomegaly', 'hypoxemia', 'no known drug allergies', 'scar tissue', 'bleeding of
vagina', 'cicatrisation', 'impaired cognition', 'hacking cough', 'stridor', 'breath-holding spell', 'retch', 'breath  sounds de
creased', 'decreased  translucency', 'distress  respiratory', 'green  sputum', 'night  sweat', 'non-productive  cough', 'pleuri
tic  pain', 'productive  cough', 'shortness  of breath', 'yellow  sputum', 'alcoholic withdrawal symptoms', 'difficulty', 'form
ication', 'motor retardation', 'sleeplessness', 'todd paralysis', 'unable to concentrate', 'weepiness', 'lip smacking', 'nasal
flaring', 'snore', 'uncoordination', 'flushing', 'hypoalbuminemia', 'pallor', 'prostatism', 'pustule', 'urinoma', 'barking coug
h', 'cystic lesion', 'emphysematous change', 'nasal discharge present', 'noisy respiration', 'polymyalgia', 'rapid shallow brea
thing', 'stuffy nose', 'history  of - blackout', 'lightheadedness', 'unwell', 'ambidexterity', 'numbness']
```

All Symptoms List got from First Query.

The whole list of symptoms of the “diseases having at least one symptom in symptoms_matched list”.

## Actual Refining:



```
# refine the input of symptoms by adding the more Symptoms to it
input2=input("enter some more symptoms from printed above Symptoms:")
arr3=input2.split(",")
for i in arr3:
    finalSymptoms.append(i)
finalSymptoms
```

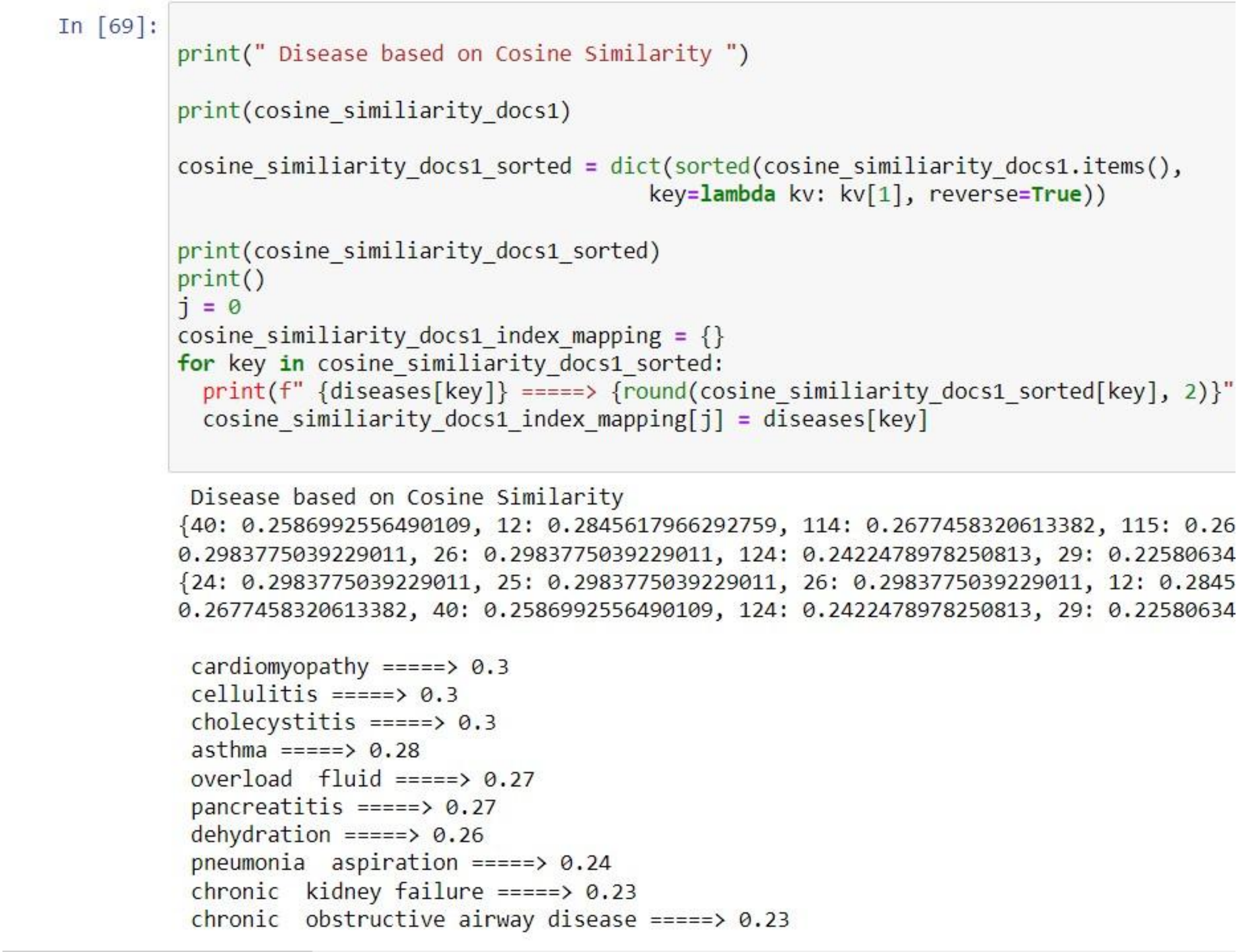
```
enter some more symptoms from printed above Symptoms:chill,cough,pain
```

```
['sneeze',
 'ache',
 'fever',
 'chill',
 'cough',
 'pain',
 'diarrhea',
 'vomiting',
 'tachypnea',
 'apyrexial',
 'shortness of breath',
 'unresponsiveness',
 'night sweat',
 'chill',
 'cough',
 'pain']
```

Enter some more symptoms (query terms)matching your need those get added to the final symptoms list.

# 4) Feedback

## Before Feedback:



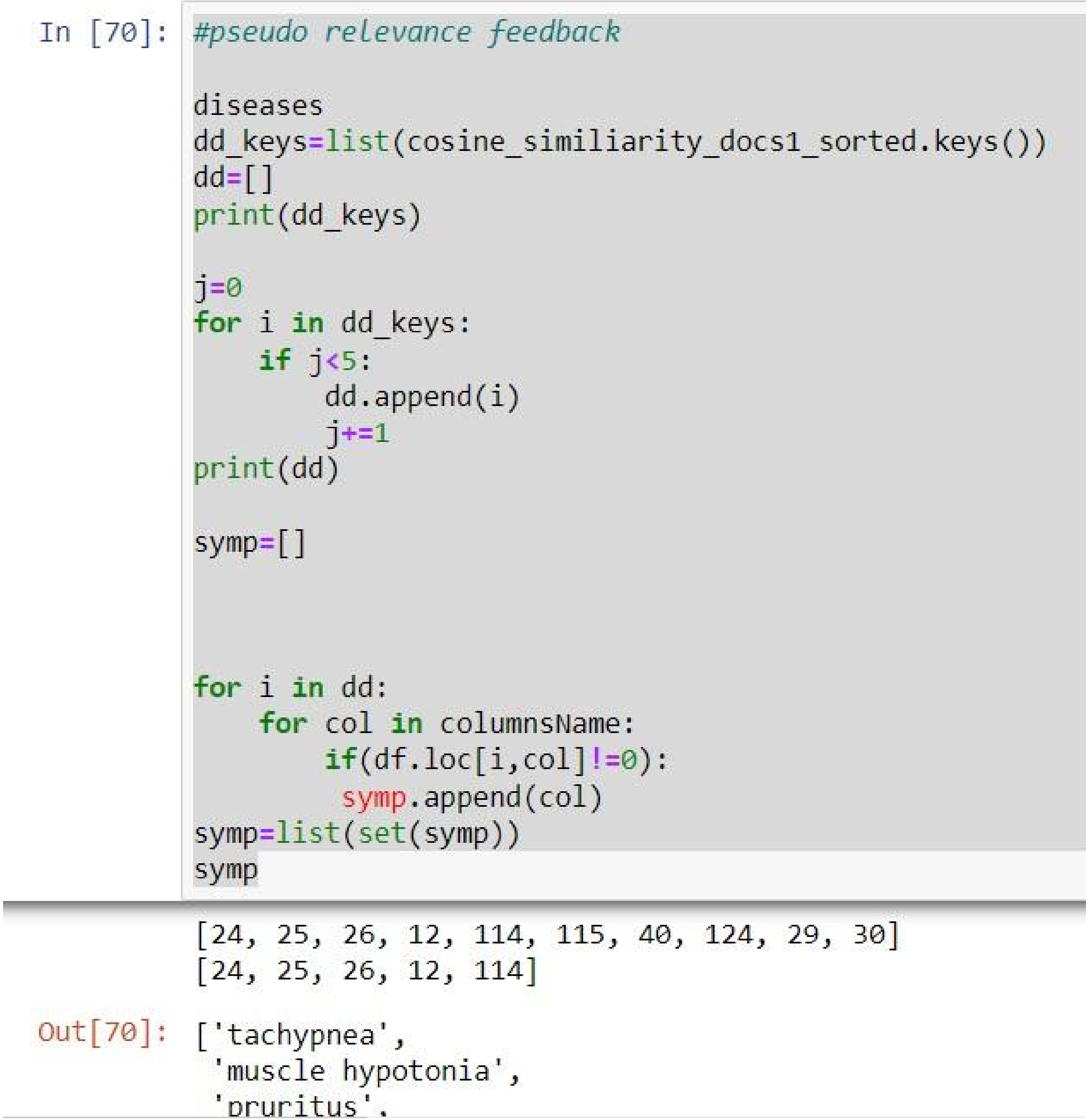
In [69]:

```
print(" Disease based on Cosine Similarity ")

print(cosine_similiarity_docs1)

cosine_similiarity_docs1_sorted = dict(sorted(cosine_similiarity_docs1.items(),
                                   key=lambda kv: kv[1], reverse=True))

print(cosine_similiarity_docs1_sorted)
print()
j = 0
cosine_similiarity_docs1_index_mapping = {}
for key in cosine_similiarity_docs1_sorted:
  print(f" {diseases[key]} =====> {round(cosine_similiarity_docs1_sorted[key], 2)}"
  cosine_similiarity_docs1_index_mapping[j] = diseases[key]
```

```
 Disease based on Cosine Similarity
{40: 0.2586992556490109, 12: 0.2845617966292759, 114: 0.2677458320613382, 115: 0.26
0.2983775039229011, 26: 0.2983775039229011, 124: 0.2422478978250813, 29: 0.22580634
{24: 0.2983775039229011, 25: 0.2983775039229011, 26: 0.2983775039229011, 12: 0.2845
0.2677458320613382, 40: 0.2586992556490109, 124: 0.2422478978250813, 29: 0.22580634

 cardiomyopathy =====> 0.3
 cellulitis =====> 0.3
 cholecystitis =====> 0.3
 asthma =====> 0.28
 overload  fluid =====> 0.27
 pancreatitis =====> 0.27
 dehydration =====> 0.26
 pneumonia  aspiration =====> 0.24
 chronic  kidney failure =====> 0.23
 chronic  obstructive airway disease =====> 0.23
```

Document retrieval(“Disease Name”) based on cosine similarity score before the pseudo relevance feedback.

## Feedback:

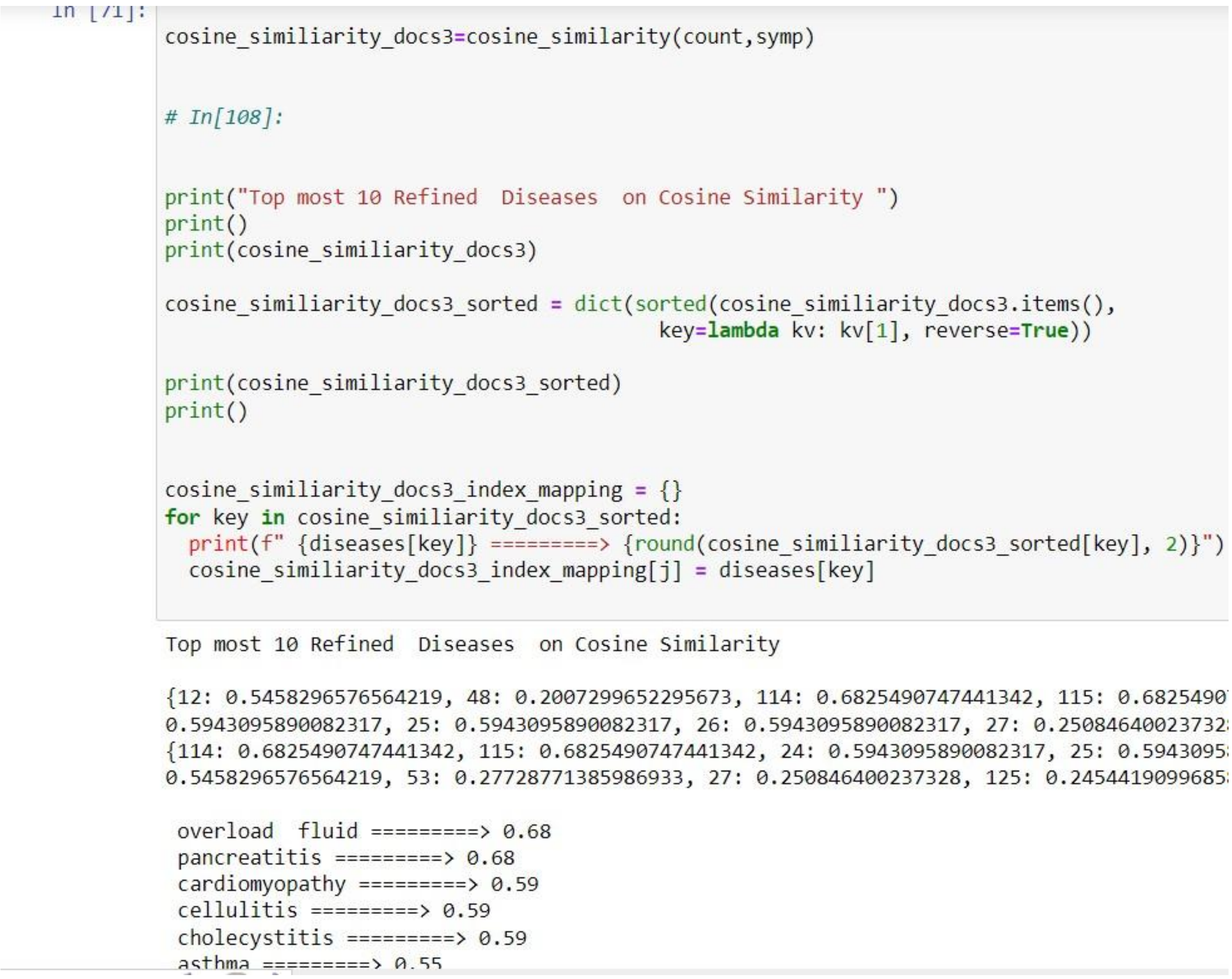


```
In [70]: #pseudo relevance feedback

         diseases
         dd_keys=list(cosine_similiarity_docs1_sorted.keys())
         dd=[]
         print(dd_keys)

         j=0
         for i in dd_keys:
             if j<5:
                 dd.append(i)
                 j+=1
         print(dd)

         symp=[]



         for i in dd:
             for col in columnsName:
                 if(df.loc[i,col]!=0):
                  symp.append(col)
         symp=list(set(symp))
         symp
```

```
         [24, 25, 26, 12, 114, 115, 40, 124, 29, 30]
         [24, 25, 26, 12, 114]

Out[70]: ['tachypnea',
          'muscle hypotonia',
          'pruritus',
```

Feedback considering Top 5 as relevant documents. And computing the Cosine similarity between query(final symptoms list) and symptoms in Top5 diseases.

## After FeedBack:



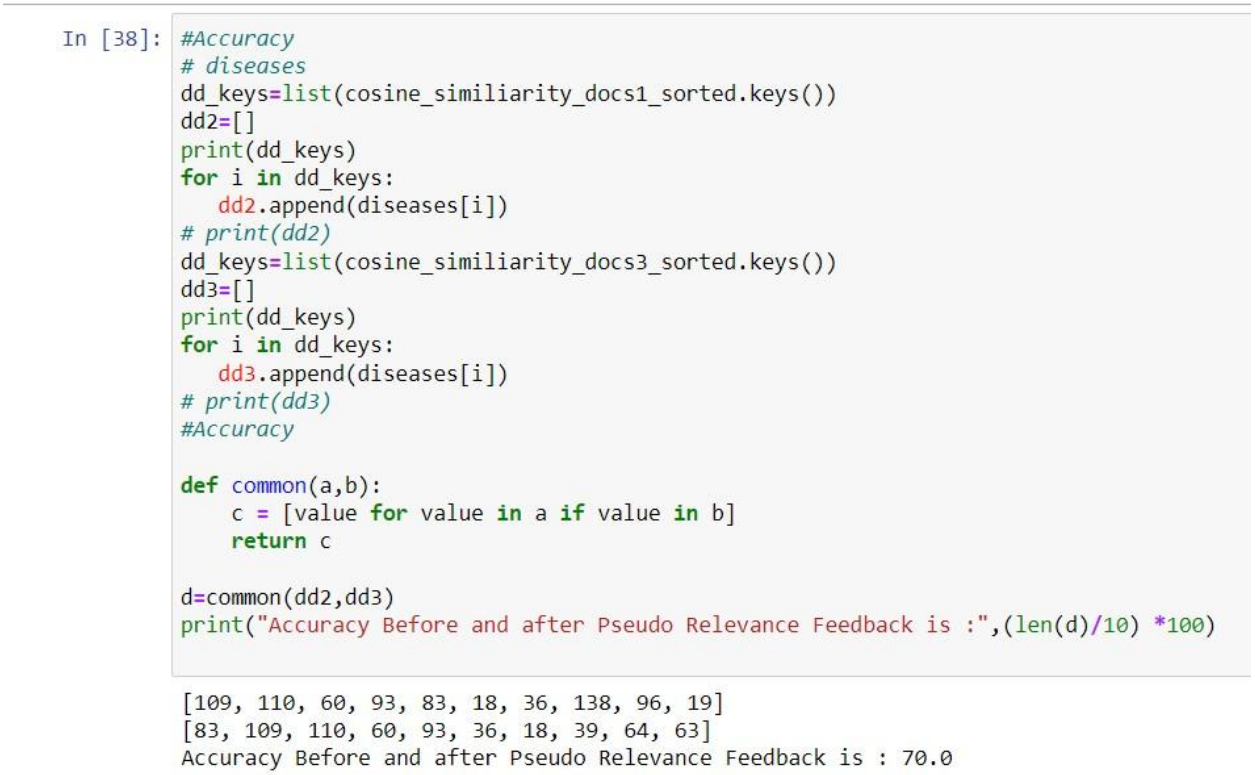
```
In [71]:
cosine_similiarity_docs3=cosine_similarity(count,symp)


# In[108]:


print("Top most 10 Refined  Diseases  on Cosine Similarity ")
print()
print(cosine_similiarity_docs3)

cosine_similiarity_docs3_sorted = dict(sorted(cosine_similiarity_docs3.items(),
                                   key=lambda kv: kv[1], reverse=True))

print(cosine_similiarity_docs3_sorted)
print()


cosine_similiarity_docs3_index_mapping = {}
for key in cosine_similiarity_docs3_sorted:
  print(f" {diseases[key]} =========> {round(cosine_similiarity_docs3_sorted[key], 2)}")
  cosine_similiarity_docs3_index_mapping[j] = diseases[key]
```

```
Top most 10 Refined  Diseases  on Cosine Similarity

{12: 0.5458296576564219, 48: 0.2007299652295673, 114: 0.6825490747441342, 115: 0.6825490
0.5943095890082317, 25: 0.5943095890082317, 26: 0.5943095890082317, 27: 0.2508464002373
{114: 0.6825490747441342, 115: 0.6825490747441342, 24: 0.5943095890082317, 25: 0.5943095
0.5458296576564219, 53: 0.27728771385986933, 27: 0.250846400237328, 125: 0.2454419099685

 overload  fluid =========> 0.68
 pancreatitis =========> 0.68
 cardiomyopathy =========> 0.59
 cellulitis =========> 0.59
 cholecystitis =========> 0.59
 asthma =========> 0.55
```

Document retrieval("Disease Name") based on cosine similarity score after the pseudo relevance feedback.

# 5)Assessment Components

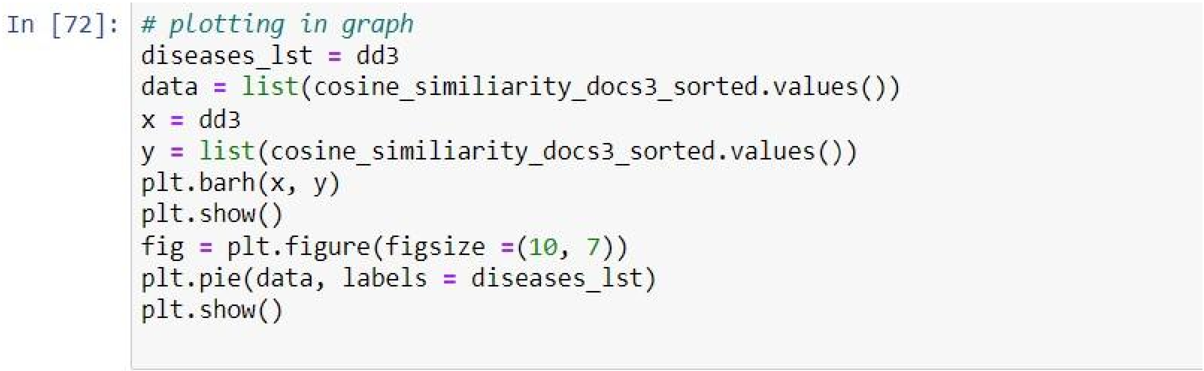
## Accuracy:



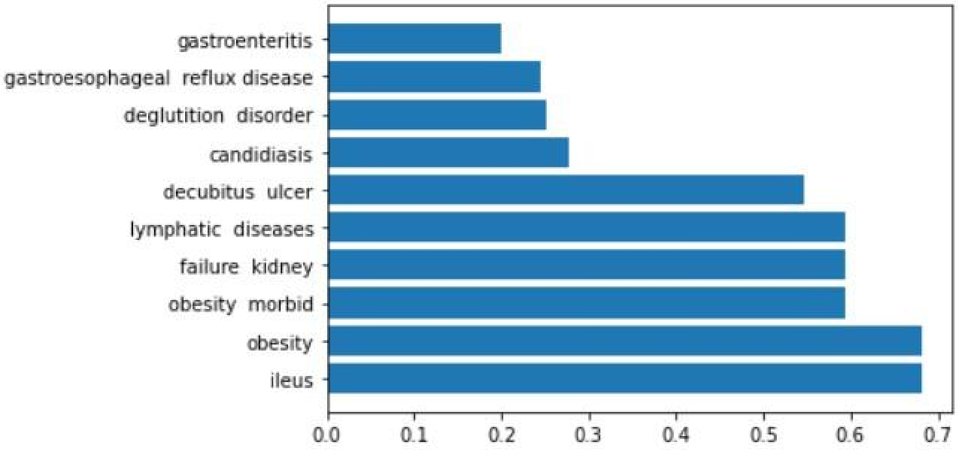
```
In [38]: #Accuracy
         # diseases
         dd_keys=list(cosine_similiarity_docs1_sorted.keys())
         dd2=[]
         print(dd_keys)
         for i in dd_keys:
            dd2.append(diseases[i])
         # print(dd2)
         dd_keys=list(cosine_similiarity_docs3_sorted.keys())
         dd3=[]
         print(dd_keys)
         for i in dd_keys:
            dd3.append(diseases[i])
         # print(dd3)
         #Accuracy

         def common(a,b):
             c = [value for value in a if value in b]
             return c

         d=common(dd2,dd3)
         print("Accuracy Before and after Pseudo Relevance Feedback is :",(len(d)/10) *100)

         [109, 110, 60, 93, 83, 18, 36, 138, 96, 19]
         [83, 109, 110, 60, 93, 36, 18, 39, 64, 63]
         Accuracy Before and after Pseudo Relevance Feedback is : 70.0
```

Finding accuracy between the documents retrieved before and after the relevance feedback.

## Plotting:



```
In [72]: # plotting in graph
         diseases_lst = dd3
         data = list(cosine_similiarity_docs3_sorted.values())
         x = dd3
         y = list(cosine_similiarity_docs3_sorted.values())
         plt.barh(x, y)
         plt.show()
         fig = plt.figure(figsize =(10, 7))
         plt.pie(data, labels = diseases_lst)
         plt.show()
```



Plotting the diseases in a Bar Graph with respective cosine similarity scores