```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)



import nltk
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()

!pip install -U scikit-learn scipy matplotlib
```

```
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.4.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (1.13.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.8.4)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.25.2)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.4)
```

```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))



# Clone the repository containing the dataset
!git clone https://github.com/imharshitaa/ML-Model.git

# Assuming the dataset file is in CSV format, you can use pandas to load it
medicines = pd.read_csv("ML-Model/medicine.csv")

# Drop the 'Index' column
medicines.drop(columns=['index'], inplace=True)

# Replace NaN values with 0
medicines.fillna(0, inplace=True)

# Now you can work with the modified dataset
print(medicines.head())

medicines.head()

medicines.shape
```

```
Cloning into 'ML-Model'...
remote: Enumerating objects: 1977, done.
remote: Counting objects: 100% (1977/1977), done.
remote: Compressing objects: 100% (1839/1839), done.
remote: Total 1977 (delta 128), reused 1957 (delta 123), pack-reused 0
Receiving objects: 100% (1977/1977), 13.41 MiB | 19.48 MiB/s, done.
Resolving deltas: 100% (128/128), done.
                                        Drug_Name Reason  \
0              A CN Gel(Topical) 20gmA CN Soap 75gm   Acne
```

```
1  A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0...   Acne
2                            ACGEL CL NANO Gel 15gm   Acne
3                               ACGEL NANO Gel 15gm   Acne
4                             Acleen 1% Lotion 25ml   Acne

                                        Description
0                      Mild to moderate acne (spots)
1  A RET 0.025% is a prescription medicine that i...
2  It is used to treat acne vulgaris in people 12...
3  It is used to treat acne vulgaris in people 12...
4  treat the most severe form of acne (nodular ac...
(9720, 3)
```

```
medicines.isnull().sum()
```

```
Drug_Name      0
Reason         0
Description    0
dtype: int64
```

```
medicines.dropna(inplace=True)
```

```
medicines.duplicated().sum()
```

```
94
```

```
medicines['Description']
```

```
0                          Mild to moderate acne (spots)
1       A RET 0.025% is a prescription medicine that i...
2       It is used to treat acne vulgaris in people 12...
3       It is used to treat acne vulgaris in people 12...
4       treat the most severe form of acne (nodular ac...
                              ...
9715                           used for treating warts
9716                       used to soften the skin cells
9717                                     used for scars
9718                                    used for wounds
9719    used to treat and remove raised warts (usually...
Name: Description, Length: 9720, dtype: object
```

```
medicines['Description'].apply(lambda x:x.split())
```

```
0                     [Mild, to, moderate, acne, (spots)]
1       [A, RET, 0.025%, is, a, prescription, medicine...
2       [It, is, used, to, treat, acne, vulgaris, in, ...
3       [It, is, used, to, treat, acne, vulgaris, in, ...
4       [treat, the, most, severe, form, of, acne, (no...
                              ...
9715                        [used, for, treating, warts]
9716                [used, to, soften, the, skin, cells]
9717                                  [used, for, scars]
9718                                 [used, for, wounds]
9719    [used, to, treat, and, remove, raised, warts, ...
Name: Description, Length: 9720, dtype: object
```

```
medicines['Description']
```

```
medicines['Description'].apply(lambda x:x.split())
```

```
medicines['Reason'] = medicines['Reason'].apply(lambda x:x.split())
```

```
medicines['Description'] = medicines['Description'].apply(lambda x:x.split())
```

```
medicines['Description'] = medicines['Description'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
medicines['Description'] = medicines['Description'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
medicines['tags'] = medicines['Description'] + medicines['Reason']
```

```
# Print column names
print(medicines.columns)

new_df = medicines[['Drug_Name','Reason','tags']]
new_df
```

```
new_df['tags'].apply(lambda x:" ".join(x))
new_df

new_df['tags'] = new_df['tags'].apply(lambda x:" ".join(x))
new_df

new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())
new_df
```

```
Index(['Drug_Name', 'Reason', 'Description', 'tags'], dtype='object')
<ipython-input-13-f39e365d7dce>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  new_df['tags'] = new_df['tags'].apply(lambda x:" ".join(x))
<ipython-input-13-f39e365d7dce>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())
```

7476 to 7500 of 9720 entries Filter

index	Drug_Name	Reason	tags
7475	Bmdrise Nasal Spray 6ml	Osteoporosis	â prevent the loss of bone that occurs in women after they have been through the menopause osteoporosis
7476	BON K2 Tablet 10'S	Osteoporosis	prevent loss of bone in men or people taking steroids, such as prednisolone and methyl prednisolone osteoporosis
7477	Supracal OS Tablet 10'S	Osteoporosis	treats bone disease in people with kidney problems (renal osteodystrophy) osteoporosis
7478	T Score Kit	Osteoporosis	balance bone deficiency osteoporosis
7479	Tricium Pth Tp3 250mcg Injection 1'S	Osteoporosis	â treat weakening of the bones in women after the menopause (change of life) osteoporosis
7480	Triple A Cal OS Tablet 10'S	Osteoporosis	â prevent the loss of bone that occurs in women after they have been through the menopause osteoporosis
7481	UNICALCIN 50iu Injection 1'sUnicalcin 100IU Injection 1ml	Osteoporosis	â prevent the loss of bone that occurs in women after they have been through the menopause osteoporosis
7482	Unicalcin NS Nasal Spray 3.7ml	Osteoporosis	prevent loss of bone in men or people taking steroids, such as prednisolone and methyl prednisolone osteoporosis
7483	Vebalone 150mg Tablet	Osteoporosis	treats bone disease in people with kidney problems (renal osteodystrophy) osteoporosis
7484	Xtracal CT Tablet 10'S	Osteoporosis	balance bone deficiency osteoporosis
7485	Zendrone 25mg Injection 1mlZendrone 50mg Injection 1ml	Osteoporosis	â treat weakening of the bones in women after the menopause (change of life) osteoporosis
7486	Zestabolin 50mg Injection 1ml	Osteoporosis	â prevent the loss of bone that occurs in women after they have been through the menopause osteoporosis
7487	ZOLDRO 4mg Injection 1's	Osteoporosis	â prevent the loss of bone that occurs in women after they have been through the menopause osteoporosis
	3D Injection 3ml3D 1% Gel 30gm3D		

Next steps: View recommended plots

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(stop_words='english',max_features=5000)
def stem(text):
  y = []

  for i in text.split():
    y.append(ps.stem(i))

  return " ".join(y)



new_df['tags'] = new_df['tags'].apply(stem)

cv.fit_transform(new_df['tags']).toarray().shape
vectors = cv.fit_transform(new_df['tags']).toarray()
feature_names = cv.get_feature_names_out()
```

```
<ipython-input-15-e5869b9c8592>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  new_df['tags'] = new_df['tags'].apply(stem)
```

```
# Now you can access the feature names
print(feature_names)

from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(vectors)
similarity = cosine_similarity(vectors)
similarity[1]

def recommend(medicine):
    medicine_index = new_df[new_df['Drug_Name'] == medicine].index[0]
    distances = similarity[medicine_index]
    medicines_list = sorted(list(enumerate(distances)),reverse=True,key=lambda x:x[1])[1:6]

    for i in medicines_list:
        print(new_df.iloc[i[0]].Drug_Name)

print("---------------------------------")

recommend("Paracetamol 125mg Syrup 60mlParacetamol 500mg Tablet 10'S")


print("---------------------------------")
import pickle
pickle.dump(new_df.to_dict(),open('medicine_dict.pkl','wb'))
pickle.dump(similarity,open('similarity.pkl','wb'))
```

```
          whilst   white   widen   withdrawn   womb   women   work   worsen
         'wound' 'wrinkl' 'wrinkles' 'year' 'younger' 'zinc' 'œpancreat']
        --------------------------------
        Oxypamol D Tablet 10'S
        Pacimol MF Tablet 10'S
        Painil Plus 100/500mg Tablet 10'S
        Pamagin Plus Gel 30gm
        Paracetamol 125mg Syrup 60mlParacetamol 500mg Tablet 10'S
        --------------------------------
```

```
print("--------------------------------")

recommend("AMTERICIN 50mg Injection 50ml")

print("--------------------------------")

recommend("Baraclude 1mg Tablet 10'S")

print("--------------------------------")

recommend("Dolentia AQ Injection 1ml")

print("--------------------------------")

recommend("Dolo Cold Tablet 10'S")

print("--------------------------------")

recommend("Placentrex V Gel 20gm")

print("--------------------------------")
```

```
--------------------------------
AF K Lotion 60ml
Albol 200mg Suspension 10ml
Amfocin Cream 10gmAmfocin Cream 30gm
Amoron Cream 30gm
Ampholyn 50mg Injection 1'S
--------------------------------
Baraclude 1mg Tablet 10'S
Cymgal 450mg Tablet 10'S
Duovir Tablet 60'SDuovir N Tablet 30'SDuovir Tablet 10'S
Virson Gel 5gm
Zidovir 300mg Tablet 10'SZidovir 300mg Capsule 60'SZidovir 100mg Capsule 100'SZidovir 100mg Capsule 10'SZidovir 50mg Solution 100mlz
--------------------------------
Abmef 100Mg Oral Suspension 60ml
Acceclowoc TH 8mg Tablet 10'S
Accewodol P Tablet 10'S
ACE Q Para Tablet 10'S
ACECARE SP Tablet 10's
--------------------------------
AC 375mg Tablet 6'SAC 100mg Tablet 10'S
Accemol Tablet 10'S
ACE Proxyvon Gel 30gm
Acebloc P Tablet 10'S
Aceclo Plus Tablet 15'S
--------------------------------
Betadine Antiseptic Ointment 20gm
Burnheal Dusting Powder 10gm
Cetrilak Strong Solution 100ml
Drez 5Cm Tulle
Hexilak Gel 20gmHexilak Gel 10gm
--------------------------------
```

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import pickle
```

```
# Load the data
medicines = pd.read_csv("ML-Model/medicine.csv")

# Drop unnecessary columns and handle missing values
medicines.drop(columns=['index'], inplace=True)
medicines.fillna(0, inplace=True)

# Combine the 'Description' and 'Reason' columns to create 'tags'
medicines['tags'] = medicines['Description'] + medicines['Reason']

# Convert 'tags' to lowercase and apply stemming
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
medicines['tags'] = medicines['tags'].apply(lambda x: ' '.join([ps.stem(word) for word in x.lower().split()]))



# Load the similarity matrix and medicine dictionary
similarity = pickle.load(open('similarity.pkl', 'rb'))
medicine_dict = pickle.load(open('medicine_dict.pkl', 'rb'))

def recommend_medicines(reason):
    # Stem and lowercase the reason
    stemmed_reason = ' '.join([ps.stem(word) for word in reason.lower().split()])

    # Find the index of the reason in the medicine dictionary
    reason_index = None
    for idx, tags in medicine_dict['tags'].items():
        if stemmed_reason in tags:
            reason_index = idx
            break

    # If the reason is found, recommend medicines based on similarity
    if reason_index is not None:
        distances = similarity[reason_index]
        medicines_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]

        recommended_medicines = []
        for i in medicines_list:
            recommended_medicines.append(medicine_dict['Drug_Name'][i[0]])

        return recommended_medicines
    else:
        return "No medicines found for the given reason."



# Example usage
reason = "headache"
recommended_medicines = recommend_medicines(reason)
print("Recommended Medicines for", reason + ":")
for medicine in recommended_medicines:
    print(medicine)
```

```
Recommended Medicines for headache:
Aedon 10mg Tablet 14'SAedon 5mg Tablet 14'S
Alam 0.50mg Tablet 10'SAlam 0.25mg Tablet 10'S
Alark 0.25mg Tablet 10'SAlark 0.5mg Tablet 10'S
Alarm 0.50mg Tablet 10'S
Albium 10mg Tablet 10'S
```

```
# Display all unique reasons
unique_reasons = medicines['Reason'].unique()
print("List of reasons for medication recommendation:")
for i, reason in enumerate(unique_reasons, start=1):
    print(f"{i}. {reason}")
```

```
List of reasons for medication recommendation:
1. Acne
2. Adhd
3. Allergies
4. Alzheimer
5. Amoebiasis
6. Anaemia
7. Angina
8. Anxiety
9. Appetite
10. Arrhythmiasis
11. Arthritis
12. Cleanser
```

```
13. Constipation
14. Contraception
15. Dandruff
16. Depression
17. Diabetes
18. Diarrhoea
19. Digestion
20. Fever
21. Fungal
22. General
23. Glaucoma
24. Gout
25. Haematopoiesis
26. Haemorrhoid
27. Hyperpigmentation
28. Hypertension
29. Hyperthyroidism
30. Hypnosis
31. Hypotension
32. Hypothyroidism
33. Infection
34. Malarial
35. Migraine
36. Mydriasis
37. Osteoporosis
38. Pain
39. Parkinson
40. Psychosis
41. Pyrexia
42. Scabies
43. Schizophrenia
44. Smoking
45. Supplement
46. Thrombolysis
47. Vaccines
48. Vertigo
49. Viral
50. Wound
```

```
# Function to recommend medicines based on the chosen reason
def recommend_medicines_for_chosen_reason(chosen_reason):
    selected_reason = unique_reasons[chosen_reason - 1]
    recommended_medicines = recommend_medicines(selected_reason)
    print("\nRecommended Medicines for", selected_reason + ":")
    for medicine in recommended_medicines:
        print(medicine)



# Function to recommend medicines based on a given reason
def recommend_medicines_for_reason():
    reason = input("Enter the reason for medication recommendation: ")
    recommended_medicines = recommend_medicines(reason)
    print("\nRecommended Medicines for", reason + ":")
    for medicine in recommended_medicines:
        print(medicine)

# Call the function to recommend medicines based on user input
recommend_medicines_for_reason()
```

```
Enter the reason for medication recommendation: pain

Recommended Medicines for pain:
Adapen Gel 15gm
Benzer 5% Gel 20gmBenzer 2.5% Gel 20gm
CLENCHIN Gel(Topical) 20gm
Clindakem A Gel 20gm
ENCLINA Gel(Topical) 20gm
```

```
# Prompt the user to choose a reason
try:
    chosen_reason = int(input("\nEnter the number corresponding to the reason for medication recommendation: "))
    if 1 <= chosen_reason <= len(unique_reasons):
        recommend_medicines_for_chosen_reason(chosen_reason)
    else:
        print("Invalid input. Please enter a valid number.")
except ValueError:
    print("Invalid input. Please enter a valid number.")
```

```
Enter the number corresponding to the reason for medication recommendation: 11

Recommended Medicines for Arthritis:
```

```
Adalirel 40mg Injection 1'S
Arava 20mg Tablet 30'SArava 10mg Tablet 30'S
Arthrella Ointment 30gm
Artilage Tablet 10'S
Bioquin 200mg Tablet 10'S


# Function to recommend medicines based on a given reason
def recommend_medicines_for_reason():
    while True:
        reason = input("\nEnter the reason for medication recommendation (or 'exit' to quit): ")
        if reason.lower() == 'exit':
            print("Exiting...")
            break

        recommended_medicines = recommend_medicines(reason)
        print("\nRecommended Medicines for", reason + ":")
        for medicine in recommended_medicines:
            print(medicine)

# Call the function to recommend medicines based on user input
recommend_medicines_for_reason()
```

```
Enter the reason for medication recommendation (or 'exit' to quit): pain

Recommended Medicines for pain:
Adapen Gel 15gm
Benzer 5% Gel 20gmBenzer 2.5% Gel 20gm
CLENCHIN Gel(Topical) 20gm
Clindakem A Gel 20gm
ENCLINA Gel(Topical) 20gm

Enter the reason for medication recommendation (or 'exit' to quit): headache

Recommended Medicines for headache:
Aedon 10mg Tablet 14'SAedon 5mg Tablet 14'S
Alam 0.50mg Tablet 10'SAlam 0.25mg Tablet 10'S
Alark 0.25mg Tablet 10'SAlark 0.5mg Tablet 10'S
Alarm 0.50mg Tablet 10'S
Albium 10mg Tablet 10'S

Enter the reason for medication recommendation (or 'exit' to quit): depression

Recommended Medicines for depression:
Agodep 25mg Tablet 10'S
Alamflu Tablet 10'S
Ambulax Ad 5Mg Tablet 10'sAmbulax Ad 10Mg Tablet 10's
Amitar 10Mg Tablet 10'sAmitar 25Mg Tablet 10's
Amitril 25mg Tablet 10'S

Enter the reason for medication recommendation (or 'exit' to quit): anxiety

Recommended Medicines for anxiety:
Aedon 10mg Tablet 14'SAedon 5mg Tablet 14'S
Alam 0.50mg Tablet 10'SAlam 0.25mg Tablet 10'S
Alark 0.25mg Tablet 10'SAlark 0.5mg Tablet 10'S
Alarm 0.50mg Tablet 10'S
Albium 10mg Tablet 10'S

Enter the reason for medication recommendation (or 'exit' to quit): viral

Recommended Medicines for viral:
ACIV 800mg Tablet 10'sACIV 200mg Tablet 10'sACIV 400mg Tablet 10's
Alltera Tablet 120'S
Anzavir R Tablet 30'S
Cmvee 350mg Tablet 2'SCmvee 450mg Tablet 2'S
Daclahep 60mg Tablet 28'S

Enter the reason for medication recommendation (or 'exit' to quit): cough

Recommended Medicines for cough:
Alex P Syrup 60ml
Asthalin DX Syrup 100ml
Cetaphil Daylong Kids SPF 50+ Lotion 150mlCetaphil Daylong SPF 30 Gel 30ml
Cosvate GM Cream 25gm
Cutiwash Soft Foaming Face Wash 60ml

Enter the reason for medication recommendation (or 'exit' to quit): exit
Exiting...
```