```
#Loading Eda Pakages
import pandas as pd
import numpy as np

#Loading data visualization pakages
import seaborn as sns
import matplotlib.pyplot as plt
```

```
!pip install neattext
```

```
Collecting neattext
  Downloading neattext-0.1.2-py3-none-any.whl (114 kB)
     |████████████████████████████████| 114 kB 5.1 MB/s
Installing collected packages: neattext
Successfully installed neattext-0.1.2
```

```
import neattext.functions as nfx
#These are text cleaning packages
```

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

```
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (1.21.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (1.4.1)
Collecting scipy
  Downloading scipy-1.7.3-cp37-cp37m-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (38
     |████████████████████████████████| 38.1 MB 1.4 MB/s
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-packages (1
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (1
Installing collected packages: scipy
  Attempting uninstall: scipy
    Found existing installation: scipy 1.4.1
    Uninstalling scipy-1.4.1:
      Successfully uninstalled scipy-1.4.1
ERROR: pip's dependency resolver does not currently take into account all the packages t
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 which is
```

```
#Loading ML pakages
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.pipeline import Pipeline
```

```
#Load ML Estimators
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.tree import DecisionTreeClassifier
```

```
1 from google.colab import files
2 uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in browser session. Please rerun this cell to enable.

Saving netflix_titles.csv to netflix_titles.csv

```
1 #Loading Dataset netflix
2 import pandas as pd
3 df = pd.read_csv("netflix_titles.csv")
4 df.head()
5 # This is will show 1st few entries
```

	show_id	type	title	director	cast	country	date_added	release_year	rat
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban... Sami Bouajila	South Africa	September 24, 2021	2021	TV

Product number 1 : recommendation system

we will be picking title and via cosine similarity, will build a recommendation system

EDA

```
1 # Let check the dimensions
2 df.shape
```

```
(8807, 12)
```

```
1 df.dtypes
2 #All objects none are strings
```

```
show_id         object
type            object
title           object
```

```
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
```

```
# Let check the number of coloumns
df.columns
```

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

Data analysis is a process of inspecting, cleansing, transforming, and modelling data with the goal of discovering useful information, informing conclusions, and supporting decision-making.

So as per the second step lets cleans the data

```
#Attribute/Methods
import neattext.functions as nfx
dir(nfx)
```

```
['BTC_ADDRESS_REGEX',
 'CURRENCY_REGEX',
 'CURRENCY_SYMB_REGEX',
 'Counter',
 'DATE_REGEX',
 'EMAIL_REGEX',
 'EMOJI_REGEX',
 'HASTAG_REGEX',
 'MASTERCard_REGEX',
 'MD5_SHA_REGEX',
 'MOST_COMMON_PUNCT_REGEX',
 'NUMBERS_REGEX',
 'PHONE_REGEX',
 'PoBOX_REGEX',
 'SPECIAL_CHARACTERS_REGEX',
 'STOPWORDS',
 'STOPWORDS_de',
 'STOPWORDS_en',
 'STOPWORDS_es',
 'STOPWORDS_fr',
 'STOPWORDS_ru',
 'STOPWORDS_yo',
 'STREET_ADDRESS_REGEX',
 'TextFrame',
 'URL_PATTERN',
```

```
 'USER_HANDLES_REGEX',
 'VISACard_REGEX',
 '__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__generate_text',
 '__loader__',
 '__name__',
 '__numbers_dict',
 '__package__',
 '__spec__',
 '_lex_richness_herdan',
 '_lex_richness_maas_ttr',
 'clean_text',
 'defaultdict',
 'digit2words',
 'extract_btc_address',
 'extract_currencies',
 'extract_currency_symbols',
 'extract_dates',
 'extract_emails',
 'extract_emojis',
 'extract_hashtags',
 'extract_html_tags',
 'extract_mastercard_addr',
 'extract_md5sha',
 'extract_numbers',
 'extract_pattern',
 'extract_phone_numbers',
 'extract_postoffice_box',
 'extract_shortwords',
 'extract_special_characters'
```

```
1 df['title'].str.lower()
```

```
0        dick johnson is dead
1               blood & water
2                   ganglands
3       jailbirds new orleans
4                kota factory
                ...
8802                   zodiac
8803              zombie dumb
8804               zombieland
8805                     zoom
8806                   zubaan
Name: title, Length: 8807, dtype: object
```

```
1 #Removing special characteristics
2 df["movie_title_clean"] = df["title"].str.lower().apply(lambda x:nfx.remove_special_charac
3 df["movie_title_clean"]
```

```
0        dick johnson is dead
1                blood  water
```

```
2                  ganglands
3       jailbirds new orleans
4                kota factory
                ...
8802                   zodiac
8803              zombie dumb
8804               zombieland
8805                     zoom
8806                   zubaan
Name: movie_title_clean, Length: 8807, dtype: object
```

```
df["movie_title_clean"] = df["title"].str.lower().apply(lambda x:nfx.remove_stopwords(x))
df["movie_title_clean"]
```

```
0           dick johnson dead
1               blood & water
2                  ganglands
3       jailbirds new orleans
4                kota factory
                ...
8802                   zodiac
8803              zombie dumb
8804               zombieland
8805                     zoom
8806                   zubaan
Name: movie_title_clean, Length: 8807, dtype: object
```

```
#Let's check the difference
df[['title' , 'movie_title_clean']]
```

	title	movie_title_clean
0	Dick Johnson Is Dead	dick johnson dead
1	Blood & Water	blood & water
2	Ganglands	ganglands
3	Jailbirds New Orleans	jailbirds new orleans
4	Kota Factory	kota factory
...	...	...
8802	Zodiac	zodiac
8803	Zombie Dumb	zombie dumb
8804	Zombieland	zombieland
8805	Zoom	zoom
8806	Zubaan	zubaan

8807 rows × 2 columns

Recommendation system using Movie Titles

```
#Convert text to Vectors Features
#to vectorise entire stuff
cv = CountVectorizer()
vectorized_text = cv.fit_transform(df["movie_title_clean"])
vectorized_text.toarray()
```

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

```
# Vocabulary
cv.vocabulary_
```

```
{'dick': 2184,
 'johnson': 4047,
 'dead': 2023,
 'blood': 1006,
 'water': 8455,
 'ganglands': 3022,
 'jailbirds': 3947,
 'new': 5507,
 'orleans': 5731,
 'kota': 4383,
 'factory': 2656,
 'midnight': 5142,
 'mass': 4979,
 'little': 4622,
 'pony': 6132,
 'generation': 3069,
 'sankofa': 6866,
 'great': 3244,
 'british': 1165,
 'baking': 682,
 'starling': 7468,
 'vendetta': 8295,
 'truth': 8074,
 'lies': 4578,
 'mafia': 4796,
 'bangkok': 717,
 'breaking': 1138,
 'je': 3981,
 'suis': 7583,
 'karl': 4184,
 'confessions': 1731,
 'invisible': 3869,
```

```
 'girl': 3129,
 'crime': 1862,
 'stories': 7515,
 'india': 3782,
 'detectives': 2136,
 'dear': 2032,
 'white': 8520,
 'people': 5974,
 'europe': 2584,
 'dangerous': 1973,
 'man': 4868,
 'otto': 5747,
 'skorzeny': 7264,
 'spain': 7383,
 'falsa': 2673,
 'identidad': 3720,
 'intrusion': 3859,
 'jaguar': 3943,
 'monsters': 5282,
 'inside': 3828,
 '24': 68,
 'faces': 2651,
 'billy': 929,
 'milligan': 5160,
 'resurrection': 6548,
 'ertugrul': 2555,
```

```
1 cv.get_feature_names()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning
  warnings.warn(msg, category=FutureWarning)
['000',
 '009',
 '01',
 '09',
 '10',
 '100',
 '1000',
 '100kg',
 '101',
 '11',
 '12',
 '122',
 '123',
 '12th',
 '13',
 '13th',
 '14',
 '15',
 '16',
 '1666',
 '17',
 '18',
 '187',
 '1897',
 '1898',
```

```
 '19',
 '1914',
 '1918',
 '1920',
 '1922',
 '1939',
 '1945',
 '1976',
 '1978',
 '1982',
 '1983',
 '1984',
 '1988',
 '1989',
 '1990',
 '1991',
 '1992',
 '1993',
 '1994',
 '1996',
 '1997',
 '1br',
 '1st',
 '20',
 '2000s',
 '2009',
 '2011',
 '2012',
 '2015',
 '2016',
```

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.metrics.pairwise import cosine_similarity
```

```
# Cosine Matrix
cosine_mat = cosine_similarity(vectorized_text)
cosine_mat
```

```
array([[1., 0., 0., ..., 0., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 0., 1., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 1., 0., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 0., 1.]])
```

```
# the indexes are not understandable hence we 1st build index of Movie Title
movie_indices = pd.Series(df.index, index = df['movie_title_clean']).drop_duplicates()
movie_indices
```

```
movie_title_clean
dick johnson dead           0
blood & water               1
ganglands                   2
jailbirds new orleans       3
kota factory                4
                          ...
zodiac                   8802
zombie dumb              8803
zombieland               8804
zoom                     8805
zubaan                   8806
Length: 8807, dtype: int64
```

```
# Geting indexes of the movie from indices
idx = movie_indices['kota factory']
idx
```

```
4
```

```
# Look inside cosine_matrix for the index
cosine_mat[idx]
```

```
array([0., 0., 0., ..., 0., 0., 0.])
```

```
# Look inside cosine_matrix for the index
sim_scores = list(enumerate(cosine_mat[idx]))
sim_scores
```

```
[(0, 0.0),
 (1, 0.0),
 (2, 0.0),
 (3, 0.0),
 (4, 0.9999999999999998),
 (5, 0.0),
 (6, 0.0),
 (7, 0.0),
 (8, 0.0),
 (9, 0.0),
 (10, 0.0),
 (11, 0.0),
 (12, 0.0),
 (13, 0.0),
 (14, 0.0),
 (15, 0.0),
 (16, 0.0),
 (17, 0.0),
 (18, 0.0),
 (19, 0.0),
 (20, 0.0),
 (21, 0.0),
 (22, 0.0),
 (23, 0.0),
```

```
 (24, 0.0),
 (25, 0.0),
 (26, 0.0),
 (27, 0.0),
 (28, 0.0),
 (29, 0.0),
 (30, 0.0),
 (31, 0.0),
 (32, 0.0),
 (33, 0.0),
 (34, 0.0),
 (35, 0.0),
 (36, 0.0),
 (37, 0.0),
 (38, 0.0),
 (39, 0.0),
 (40, 0.0),
 (41, 0.0),
 (42, 0.0),
 (43, 0.0),
 (44, 0.0),
 (45, 0.0),
 (46, 0.0),
 (47, 0.0),
 (48, 0.0),
 (49, 0.0),
 (50, 0.0),
 (51, 0.0),
 (52, 0.0),
 (53, 0.0),
 (54, 0.0),
 (55, 0.0),
 (56, 0.0),
 (57, 0.0),
```

```
1 sim_scores_sorted = sorted(sim_scores, key = lambda x: x[1], reverse = True)
2 sim_scores_sorted
```

```
[(4, 0.9999999999999998),
 (3567, 0.4999999999999999),
 (4569, 0.4999999999999999),
 (6446, 0.408248290463863),
 (7275, 0.408248290463863),
 (3535, 0.35355339059327373),
 (8745, 0.35355339059327373),
 (0, 0.0),
 (1, 0.0),
 (2, 0.0),
 (3, 0.0),
 (5, 0.0),
 (6, 0.0),
 (7, 0.0),
 (8, 0.0),
 (9, 0.0),
 (10, 0.0),
```

```
 (11, 0.0),
 (12, 0.0),
 (13, 0.0),
 (14, 0.0),
 (15, 0.0),
 (16, 0.0),
 (17, 0.0),
 (18, 0.0),
 (19, 0.0),
 (20, 0.0),
 (21, 0.0),
 (22, 0.0),
 (23, 0.0),
 (24, 0.0),
 (25, 0.0),
 (26, 0.0),
 (27, 0.0),
 (28, 0.0),
 (29, 0.0),
 (30, 0.0),
 (31, 0.0),
 (32, 0.0),
 (33, 0.0),
 (34, 0.0),
 (35, 0.0),
 (36, 0.0),
 (37, 0.0),
 (38, 0.0),
 (39, 0.0),
 (40, 0.0),
 (41, 0.0),
 (42, 0.0),
 (43, 0.0),
 (44, 0.0),
 (45, 0.0),
 (46, 0.0),
 (47, 0.0),
 (48, 0.0),
 (49, 0.0),
 (50, 0.0),
```

```
selected_movies_indices = [i[0] for i in sim_scores_sorted[1:]]
selected_movies_scores = [i[1] for i in sim_scores_sorted[1:]]
# df['similarity_scores'] = selected_movies_indices
```

```
result_df = df.iloc[selected_movies_indices]
result_df['similarity_scores'] = selected_movies_scores
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: 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_
```

```
# movie has been recomended
result_df[['title' , 'similarity_scores']]
# result_df
```

	title	similarity_scores
3567	American Factory	0.500000
4569	Harishchandrachi Factory	0.500000
6446	Charlie and the Chocolate Factory	0.408248
7275	LeapFrog: Letter Factory	0.408248
3535	American Factory: A Conversation with the Obamas	0.353553
...	...	...
8802	Zodiac	0.000000
8803	Zombie Dumb	0.000000
8804	Zombieland	0.000000
8805	Zoom	0.000000
8806	Zubaan	0.000000

8806 rows × 2 columns

```
def get_recommendation(title , cosine_sim_mat, num_of_rec = 10):
    #indices of the course
    movie_indices = pd.Series(df.index, index = df['movie_title_clean']).drop_duplicates()
    # Index of course
    idx = movie_indices[title]

    #Looking into cosine mat for that index
    sim_scores = list(enumerate(cosine_sim_mat[idx]))
    sim_scores = sorted(sim_scores, key = lambda x: x[1], reverse = True)
    selected_course_indices = [i[0] for i in sim_scores[1:]]
    selected_movies_scores = [i[1] for i in sim_scores[1:]]

    #Get the dataframe and title
    result_df = df.iloc[selected_course_indices]
    result_df['similarity_scores'] = selected_movies_scores
    final_recommended_courses = result_df[['title' , 'similarity_scores' , 'movie_title_cl
    return final_recommended_courses.head(num_of_rec)
```

```
df.iloc[20]
```

```
show_id                                                    s21
```

```
type                                                          TV Show
title                 Monsters Inside: The 24 Faces of Billy Milligan
director                                                Olivier Megaton
cast                                                              NaN
country                                                           NaN
date_added                                          September 22, 2021
release_year                                                     2021
rating                                                          TV-14
duration                                                    1 Season
listed_in           Crime TV Shows, Docuseries, International TV S...
description         In the late 1970s, an accused serial rapist cl...
movie_title_clean             monsters inside: 24 faces billy milligan
Name: 20, dtype: object
```

```
df.iloc[20].title
```

```
'Monsters Inside: The 24 Faces of Billy Milligan'
```

```
#function to clean text
def preprocess_text(docx):
    results = nfx.remove_stopwords(nfx.remove_special_characters(docx.lower()))
    return results
```

```
ex1 = preprocess_text(df.iloc[130].title)
```

```
ex1
```

```
'barbie big city big dreams'
```

```
# Make recommendations
get_recommendation(ex1 , cosine_mat)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:15: 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_
  from ipykernel import kernelapp as app
```

	title	similarity_scores	movie_title_clean	cast	description
1572	The Big Show Show	0.755929	big	Paul Wight, Allison Munn, Reylynn Caster, Lily...	Former WWE wrestler the Big Show is out of the...
6304	Big Dreams, Small Spaces	0.566947	big dreams, small spaces	Monty Don	Writer and presenter Monty Don helps England's...
549	Big Timber	0.534522	big timber	NaN	A no-nonsense logger and his loyal crew battle...
1093	The Big Day	0.534522	big day	NaN	For six engaged couples, happily ever after be...
				Nick Kroll, John	Teenage friends