# CA-II

# ADVANCED MACHINE LEARNING

Name: Shaikh Amaan Mohd Abdul Gaffar

Roll No: AI3113

PROJECT NAME: Product Recommendation on Amazon Data

Problem Statement: -

The project aims to build a search engine that allows users to search for products in the Amazon Product Dataset using a query. It uses natural language processing techniques such as tokenization, stemming, and TF-IDF vectorization to convert product titles and descriptions into a numerical format and calculate cosine similarity between the query and the products.

Dataset Link: https://www.kaggle.com/datasets/karkavelrajaj/amazon-sales-dataset

Python Code:

import numpy as np

import pandas as pd

import pandas as pd

import numpy as np

import nltk

nltk.download()

amzon\_df = pd.read\_csv('amazon\_product.csv')

amzon\_df.drop('id',axis=1)

from nltk.stem.snowball import SnowballStemmer

stemmer = SnowballStemmer("english")

def tokenize\_stem(text):

    tokens = nltk.word\_tokenize(text.lower())

    stem = [stemmer.stem(w) for w in tokens]

    return " ".join(stem)

amzon\_df['stemmed\_tokens'] = amzon\_df.apply(lambda row: tokenize\_stem(row['Title'] + ' ' + row['Description']), axis=1)

amzon\_df['stemmed\_tokens']

def search\_product(query):

    stemmed\_query = tokenize\_stem(query)

    #calcualting cosine similarity between query and stemmed tokens columns

    amzon\_df['similarity'] = amzon\_df['stemmed\_tokens'].apply(lambda x:cosine\_sim(stemmed\_query,x))

    res = amzon\_df.sort\_values(by=['similarity'],ascending=False).head(10)[['Title','Description','Category']]

    return res

search\_product(' PURELL ES8 Professional HEALTHY SOAP Foam Refill, Fresh Scent Fragrance, 1200 mL Soap Refill for PURELL ES8 Touch-Free Dispenser (Pack of 2) - 7777-02 ')

amzon\_df['Title'][10]

**Streamlit code :**

import pandas as pd  
import numpy as np  
import nltk  
from nltk.stem.snowball import SnowballStemmer  
from sklearn.feature\_extraction.text import TfidfVectorizer  
from sklearn.metrics.pairwise import cosine\_similarity  
import streamlit as st  
  
# Load the dataset  
data = pd.read\_csv('amazon\_product\_dataset.csv')  
  
# Remove unnecessary columns  
data = data.drop('id'**,** axis=**1**)  
  
# Define tokenizer and stemmer  
stemmer = SnowballStemmer('english')  
def tokenize\_and\_stem(text):  
 tokens = nltk.word\_tokenize(text)  
 stems = [stemmer.stem(t) for t in tokens]  
 return stems  
  
# Create stemmed tokens column  
data['stemmed\_tokens'] = data.apply(lambda row: tokenize\_and\_stem(row['Title'] + ' ' + row['Description'])**,** axis=**1**)  
  
# Define TF-IDF vectorizer and cosine similarity function  
tfidf\_vectorizer = TfidfVectorizer(tokenizer=tokenize\_and\_stem)  
def cosine\_sim(text1**,** text2):  
 tfidf\_matrix = tfidf\_vectorizer.fit\_transform([text1**,** text2])  
 return cosine\_similarity(tfidf\_matrix)[**0**][**1**]  
  
# Define search function  
def search\_products(query):  
 query\_stemmed = tokenize\_and\_stem(query)  
 data['similarity'] = data['stemmed\_tokens'].apply(lambda x: cosine\_sim(query\_stemmed**,** x))  
 results = data.sort\_values(by=['similarity']**,** ascending=False).head(**10**)[['Title'**,** 'Description'**,** 'Category']]  
 return results  
  
# Create Streamlit app  
st.title('Amazon Product Search')  
  
# Create search box and button  
query = st.text\_input('Enter a product name')  
search\_button = st.button('Search')  
  
# Perform search and display results  
if search\_button:  
 results = search\_products(query)  
 st.write(results)

Abstract of Project:

Amazon's success relies heavily on its product recommendation system. This system analyzes user behavior to predict their preferences and suggest relevant products. Research in this area explores various techniques to build such recommender systems using the vast amount of data available on Amazon, including product information, ratings, reviews, and purchase history. Common approaches include collaborative filtering, which recommends items similar to those a user has interacted with in the past, and content-based filtering, which recommends items with similar features to products the user has liked. Researchers are also investigating the use of deep learning techniques to create even more accurate and personalized recommendations.

Conclusion:

In conclusion, we successfully built a search engine that allows users to search for products in the Amazon Product Dataset using a query. The engine uses natural language processing techniques to convert product titles and descriptions into a numerical format and match them with user queries in a similar format. This enables the search engine to understand the semantic meaning behind the words, allowing for more natural and accurate searches. Users can now find relevant products even if they don't use the exact keywords present in the titles or descriptions. This improvement can significantly enhance the user experience and lead to faster product discovery on the Amazon Product Dataset.