



Fashion Recommender Engine
Using TF-IDF Language Processing and Open-
CV Face Detection

Submitted by: Rohan Kiran Gunjal

21ECB0A49

Department of Electronics and

Communication Engineering

National Institute of Technology, Warangal

November, 2022

Submission to: Dr K. Ravi Kishore

Software Used: KNIME Analytics & Jupyter Notebooks

Abstract: This project deals with the problem of recommendation system using cosine similarity and takes the useful input using open-cv face and gender detect. The dataset was downloaded from Kaggle and contained a list of products from Mytra.com.

Details of the product like brand name, description, etc. are then processed by NLP pre-processing nodes in KNIME. This is then fed into TF-IDF to get tag vectors of the products in the list, which will later be used to recommend products to the user.

From face data, it gets the gender data of the user and uses the suitable data to recommend. The model then recommends top 5 highest rated products and then lets the user select anyone of the product.

The model then, by using cosine similarity tells the user which products to recommend.

Implementation:

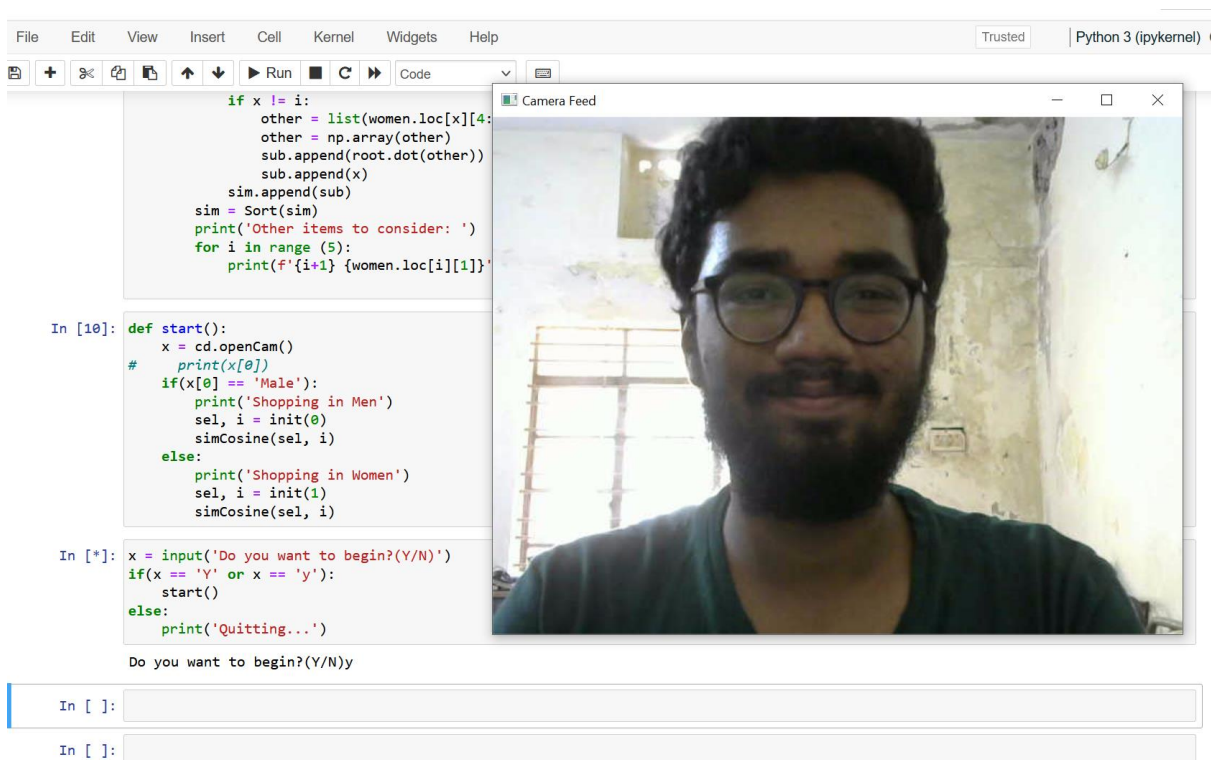
1) Begin:

```
In [*]: x = input('Do you want to begin?(Y/N)')
if(x == 'Y' or x == 'y'):
    start()
else:
    print('Quitting...')
```

Do you want to begin?(Y/N)

In []:

2) Detect Gender



The screenshot shows a Jupyter Notebook interface with a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The notebook contains several code cells. The first cell shows a loop for processing women's locations. The second cell defines a `start()` function that opens a camera, prints the first frame, and branches based on gender detection. The third cell is the same input prompt as in the first section. A 'Camera Feed' window is open, displaying a video of a man with glasses and a beard. Below the code cells are input fields for the notebook's execution state.

```
if x != i:
    other = list(women.loc[x][4:
    other = np.array(other)
    sub.append(root.dot(other))
    sub.append(x)
    sim.append(sub)
    sim = Sort(sim)
    print('Other items to consider: ')
    for i in range(5):
        print(f'{i+1} {women.loc[i][1]}')
```

```
In [10]: def start():
x = cd.openCam()
print(x[0])
#
if(x[0] == 'Male'):
    print('Shopping in Men')
    sel, i = init(0)
    simCosine(sel, i)
else:
    print('Shopping in Women')
    sel, i = init(1)
    simCosine(sel, i)
```

```
In [*]: x = input('Do you want to begin?(Y/N)')
if(x == 'Y' or x == 'y'):
    start()
else:
    print('Quitting...')
```

Do you want to begin?(Y/N)y

In []:

In []:

3) Initial input:

```

Do you want to begin?(Y/N)y
Shopping in Men
0 https://www.myntra.com/socks/hrx-by-hrithik-roshan/hrx-by-hrithik-roshan-men-quarter-length-pack-of-3-terry-socks/976816/buy
1 https://www.myntra.com/shirts/wrogn/wrogn-men-maroon--off-white-slim-fit-checked-casual-shirt/11361424/buy
2 https://www.myntra.com/shirts/roadster/roadster-men-navy-blue--red-striped-pure-cotton-sustainable-casual-shirt/10945090/buy
3 https://www.myntra.com/shirts/dennis-lingo/dennis-lingo-men-white-slim-fit-casual-shirt/14094532/buy
4 https://www.myntra.com/tshirts/wrogn/wrogn-men-black-printed-round-neck-t-shirt/11363104/buy

Enter the clothes you want to wear:

```

4) Final output:

```
print('Quitting...')
```

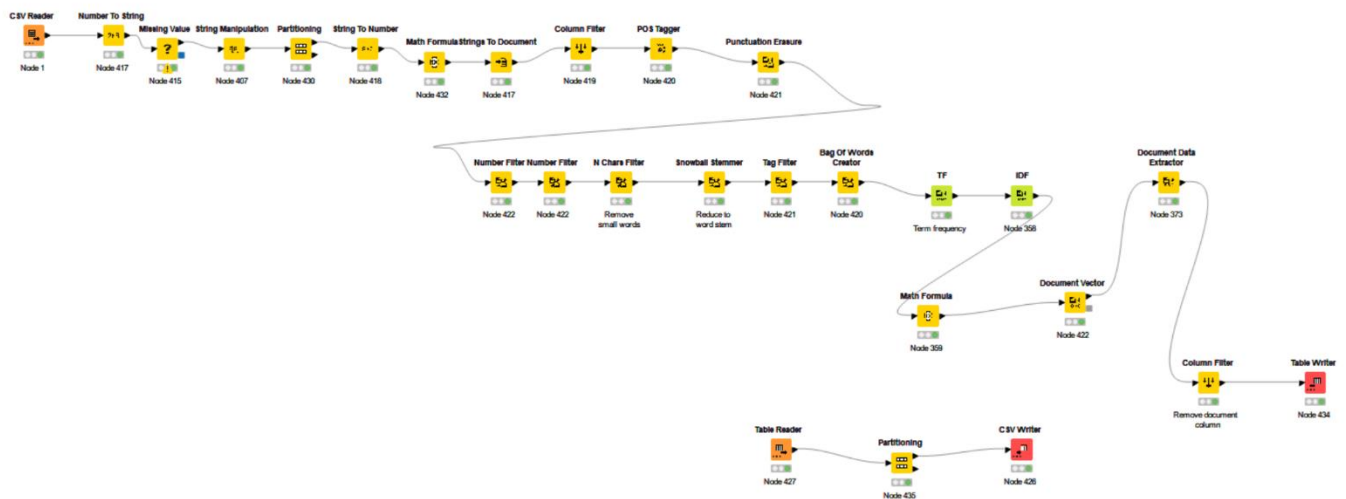
```

Do you want to begin?(Y/N)y
Shopping in Men
0 https://www.myntra.com/socks/hrx-by-hrithik-roshan/hrx-by-hrithik-roshan-men-quarter-length-pack-of-3-terry-socks/976816/buy
1 https://www.myntra.com/shirts/wrogn/wrogn-men-maroon--off-white-slim-fit-checked-casual-shirt/11361424/buy
2 https://www.myntra.com/shirts/roadster/roadster-men-navy-blue--red-striped-pure-cotton-sustainable-casual-shirt/10945090/buy
3 https://www.myntra.com/shirts/dennis-lingo/dennis-lingo-men-white-slim-fit-casual-shirt/14094532/buy
4 https://www.myntra.com/tshirts/wrogn/wrogn-men-black-printed-round-neck-t-shirt/11363104/buy

Enter the clothes you want to wear:2
Other item to consider:
https://www.myntra.com/shirts/wrogn/wrogn-men-maroon--off-white-slim-fit-checked-casual-shirt/11361424/buy

```

KNIME Dataset Creation:



Theory:

STEP 1:

The first step involved cleaning up the data, for e.g., the rows where the ratings and number of reviews are zero or where multiple values are missing. Next step involved joining the description, product name, and other details together for document vectorization. After removing punctuations and such, the concatenated column is converted to a document. Later the document terms are grouped according to nouns, etc. and captured in a bag of words. This is then fed to the TF-IDF unit and frequency is obtained.

The usage popularity is calculated by multiplying TF and IDF and the document is vectorized and made available to use in csv format.

STEP 2:

Next step involved creating a webcam video capture which returned frames, using OpenCV. The models used are pretrained IMDB face, age and gender detect models. This is applied on a frame which figures out the age and gender of the frame fed to the model.

This data is then used by the recommendation engine.

STEP 3:

The weights/ frequencies obtained from KNIME are then concatenated with the respective Product ID.

Cosine Similarity is then used to predict the products related to the one selected by the user.

Cosine Similarity is defined by:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Insights:

NLP:

The methods used to extract useful tags for describing the product details, many well developed ideas have been thoroughly tested and researched. Grouping of words based on nouns and adjectives, and deleting the words representing positional details help in simplifying the tag creation process.

Part of Speech (POS) helps in achieving the same. The bag of words model is then used to group the words for a given description. The frequencies of the terms used in document is calculated and normalized, known as the Term Frequency or TF. The frequency of certain letters like “a”, “the”, “of”, etc. provide no insight into the product details and are therefore somehow need to be eliminated. This is done by Inverse Document Frequency or IDF which is the negative log of the (number of unique terms)/ (number of times the term has occurred).

The $TF * IDF$ term gives the “popularity” of the term.

The cosine similarity then gives the similarity of a given product with another product. Higher the value, higher the similarity. Maximum out of these gives the product most similar to the given product.

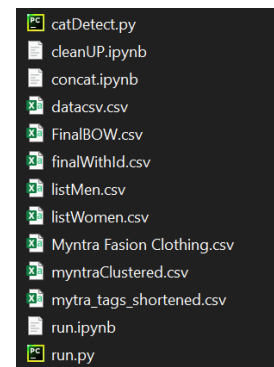
OPEN-CV:

A face frame is first obtained and converted to a blob which is used to apply the pre-trained models. This then gives the return array of [‘gender’, ‘age’].

Data Creation:

This was by far the hardest thing to do in the project, as it involved a ton of trials to get the data to work right.

A lot of work was required to get the data to work right.



Conclusion & Future

Aspects:

This project has a high potential for fashion retail industry, further improvements like face structure, race, age can be used to recommend products that better suit the user. This project utilizes all many aspects of machine learning from using pretrained models to data clean up.

If more work is put into research along with recommendations from fashion specialists will help us in recommending more satisfactory items, resulting in retaining the users.

Here's an example for the same:

