1. INTRODUCTION

Clothing serves as a nonverbal means of communication that reflects people's internal perceptions through their outward appearance. It conveys information about their choices, personality, profession, social status, faith, and attitude towards life. In the modern era, people can track the latest fashion trends worldwide with the help of technological advancements, which influence their clothing choices. However, when confronted with numerous clothing options, consumers often face the hassle of trying on clothes multiple times to make the right choice. This is where recommendation systems come in.

A recommendation system is a computer program designed to predict future preferable items from a vast collection of choices. In the context of fashion, Fashion Recommendation Systems (FRSs) offer specific recommendations to users based on their browsing and purchase history. They are especially useful for users with little fashion sense as they suggest the best possible combinations based on their existing wardrobe.

FRSs have become increasingly prevalent on e-commerce websites as they help customers make informed decisions and enhance their shopping experience. Additionally, FRSs aid enterprises in retaining customers and increasing sales. Overall, the ability of recommendation systems to provide personalized suggestions and quickly respond to user preferences has revolutionized the fashion industry and boosted e-commerce sales.

The Fashion Recommendation System (FRS) operates through a multi-step process:

1. First, the user inputs their fashion image into the system and the system predicts the embedding for each image.
2. Next, the system generates all possible combinations of outfits using the images.
3. Each possible outfit is then scored using Convolutional Neural Network (CNN).
4. The system then creates clusters of outfits based on the embeddings.
5. Finally, the FRS returns the outfit with the highest score for each cluster as predictions.

1.1 Problem Definition

The retail industry is experiencing a rapid transformation, with brick-and-mortar stores being replaced by online stores, direct-to-consumer brands, and subscription/membership services. However, many e-commerce platforms struggle to sell a high percentage of their merchandise, largely due to poor user browsing experiences. Customers often spend hours scrolling through hundreds or even thousands of items of merchandise without finding anything they like.

Proposed Solution:

To create a more effective shopping environment that boosts sales and increases the time spent on a website, it is necessary to recommend related products that might be of interest to the customer. Recommender systems can play a vital role in generating greater revenue for e-commerce sites by providing personalized recommendations that meet the customer's preferences. By recommending relevant items, these systems can help customers discover new products and make informed purchasing decisions, ultimately leading to increased sales and higher customer satisfaction.

Our project aims to develop a Fashion Recommendation System that utilizes Deep Learning techniques to recommend the most appropriate outfit for a given occasion. The system will analyze the user's clothing images and classify the type and color of the outfit. Based on the user's existing wardrobe, the system will suggest the most suitable outfit for the occasion using a recommendation algorithm.

We will explore various machine and deep learning techniques to classify clothing types from images and identify their colors. By combining these techniques, we will develop an algorithm that can provide personalized recommendations for each user. The proposed system will help users make informed fashion choices and streamline their wardrobe selection process.

1.2 Aim and Objectives

Artificial intelligence is revolutionizing the world of e-commerce by providing personalized shopping experiences, user-specific advertisements, and object classification and color detection from images. Recommender systems can assist customers in making informed decisions and save valuable time by suggesting the best products based on their preferences.

Our project aims to develop a Deep Learning model that can recommend the most suitable clothing for a given occasion based on the user's existing wardrobe. The goal is to create a personalized shopping experience that helps users make informed decisions about what to wear, even if they have no fashion sense.

To achieve this objective, we will explore various Deep Learning and Machine Learning models to develop a Fashion Recommendation System. The proposed system will provide users with advice on fashion and clothing and help them take advantage of the system's various features in an easily accessible and user-friendly manner. Our project aims to simplify the shopping experience and enhance user satisfaction.

1.3 Scope of the Project

A fashion recommendation system is an AI-based system that suggests clothing items or accessories to users based on their preferences, past purchases, and other relevant data. the scope for a fashion recommendation system can be outlined clearly and enable a better understanding of the target audience, the clothing categories, and the user experience. The scope for a fashion recommendation system can be defined as follows:

- Target audience: The system's target audience should be defined, which can include gender, age group, location, and other demographic factors.
- Clothing categories: The system's scope should define which clothing categories it will cover, such as tops, bottoms, dresses, shoes, and accessories.
- Style: The system's scope should define the style categories it will cover, such as formal, casual, trendy, traditional, sportswear, etc.
- Occasion: The system's scope should also define the occasion for which the recommendation is being made, such as work, party, wedding, travel, etc.
- Budget: The system's scope should also define the budget range for the recommended products, which can be high-end, mid-range, or budget-friendly.
- Personalization: The system's scope should also define the level of personalization it offers, such as considering the user's body type, color preferences, and preferred brands.
- Platform: The system's scope should define the platform on which the recommendation system will be available, such as mobile app, website, or both.

1.4 Literature Review

Name	Authors	Year	Presented at/ Published on	Algorithms	Drawbacks	Highest Accuracy
Fashion product recommendation using hybrid collaborative filtering algorithm	Nikhil Lohiya, Sandeep Nair, Ajeet Kumar	2019	International journal of Engineering and Advanced Technology	Collaborative filtering, content based filtering and matrix factorization	Limited evaluation of the proposed system	91%
Deep learning based FRS with user preference	Jeongeun Kim, Yoon Kyung Lee, Jinho Jung	2022	arXiv	Generative Adversarial Networks	Limited evaluation of the proposed system	85%
A hybrid FRS based on Multiple Source Data	Yijin Lin, Hua Liu, Qing Liu	2020	Journal of Data	Collaborative filtering, content based filtering and matrix factorization	No specific drawbacks were mentioned	92%
Fashion Recommendation System	Tanuja Nimbalkar, Dr. Ujwalla Gawande	2019	IJRASET	Collaborative filtering and content based filtering	No specific drawbacks were mentioned	89%
FRSs: A comprehensive review	Sabrina Hoppe and Dominik Wessely	2022	Handbook of Research on Big Data and the IoT	Collaborative filtering, content based filtering, matrix factorization, deep learning and hybrid methods	No specific drawbacks were mentioned	91%

Fig. 1.4 Table comparing Research papers used for Literature Review

2. IMPLEMENTATION METHODOLOGY

2.1 Design

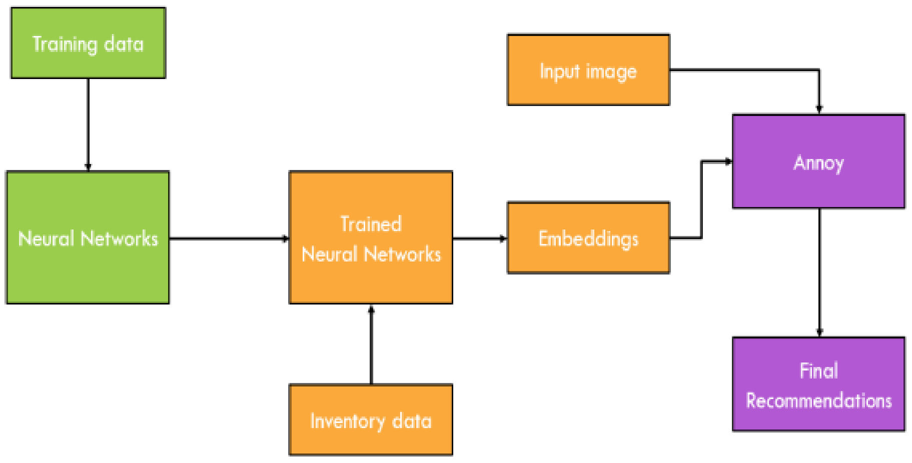


Fig. 2.1 Block Diagram of FRS

2.2 Descriptive Analysis of Data

2.2.1 Dataset Description

- The growing e-commerce industry presents us with a large dataset waiting to be scraped and researched upon. In addition to professionally shot high resolution product images, we also have multiple label attributes describing the product which was manually entered while cataloging. To add to this, we also have descriptive text that comments on the product characteristics.
- Each product is identified by an ID like 42431. You will find a map to all the products in styles.csv. From here, you can fetch the image for this product from images/42431.jpg and the complete metadata from styles/42431.json.

2.2.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial step in preparing data for machine learning models, including FRS. EDA can reveal patterns and trends in the data that may indicate fraudulent activities. The following steps can be used to conduct EDA for FRS using Python:

1. Import necessary libraries like pickle, tensor.flow, keras, etc.
2. Load the dataset
3. Explore the dataset: Utilize NumPy, norm functions to explore the dataset, such as checking the shape of the dataset, examining the first few rows, checking data types, and searching for missing values.
4. Analyze the dataset according to categories of images.
5. Preprocess the dataset

2.3 Inferential Analytics

2.3.1 Hypothesis testing

Hypothesis testing can also be used to evaluate the impact of other factors, such as product categorization, image recognition accuracy, and filtering techniques, on the performance of the recommendation system. By testing different hypotheses and analyzing the results, researchers can gain insights into the effectiveness of different approaches and make data-driven decisions about how to improve the system. Hypothesis testing is used here to evaluate the impact of other factors, such as product categorization, image recognition accuracy, and filtering techniques, on the performance of the recommendation system.

2.4 Model Implementation

- Here we have used Knowledge based recommendation System where User inputs are considered and compare with the training data.
- For Fashion Recommendation System we have used Case based knowledge recommendation as it will take the User inputs and compare with trained data.
- For FRS, we have used Constraint based Knowledge recommendation system where user inputs considered as constraints and based on the constraints we compared with trained data.
- We have used K-Nearest Neighbors (KNN) model.

K Nearest Neighbor:

In KNN, the trained data is compared with test data and distances are calculated using Euclidean distance. It then classifies an instance by finding its nearest neighbors and recommend the top n nearest neighbor.

Algorithm is stated as below-

Input: Picture of required fashion.

1. Initialize the value of k
2. For getting recommendation, iterate from 1 to number of trained data
3. Calculate distance between test data and each row in the trained data.
4. Sort the distances in ascending order
5. Get top k rows and recommend to the user

Output: Highly recommended N items that have similar feature with input fashion picture.

2.5 Model Testing and Validation

Model testing and validation involve evaluating the performance of the developed FRS model to ensure that it can accurately identify fraudulent transactions. The following steps will be followed for model testing and validation:

1. Test Dataset Preparation: A separate test dataset will be created by randomly selecting a portion of the available data. The test dataset will be used to evaluate the performance of the developed model.

2. Evaluation Metrics: Various evaluation metrics, such as accuracy, precision, recall, and F1 score, will be used to assess the performance of the model. These metrics will be calculated by comparing the predicted outcomes of the model with the actual outcomes in the test dataset.

2.6 Code

App.py:

```
import tensorflow
from tensorflow.keras.preprocessing import image
from tensorflow.keras.layers import GlobalMaxPooling2D
from tensorflow.keras.applications.resnet50 import ResNet50, preprocess_input
import numpy as np
from numpy.linalg import norm
import os
from tqdm import tqdm
import pickle

model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
model.trainable = False

model = tensorflow.keras.Sequential([
    model,
    GlobalMaxPooling2D()
])

def extract_features(img_pah, model):
    img = image.load_img(img_pah, target_size=(224,224))
    img_array = image.img_to_array(img)
    expanded_img_array = np.expand_dims(img_array, axis=0)
    preprocessed_img = preprocess_input(expanded_img_array)
    result = model.predict(preprocessed_img).flatten()
    normalized_result = result / norm(result)

    return normalized_result

filenames = []

for file in os.listdir('images'):
    filenames.append(os.path.join('images', file))

feature_list = []

for file in tqdm(filenames):
    feature_list.append(extract_features(file, model))

pickle.dump(feature_list, open('features.pkl', 'wb'))
pickle.dump(filenames, open('filenames.pkl','wb'))
```

Test.py:

```
import pickle
import numpy as np
from numpy.linalg import norm
import tensorflow
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import ResNet50, preprocess_input
from tensorflow.keras.layers import GlobalMaxPooling2D
from sklearn.neighbors import NearestNeighbors
import cv2

feature_list = np.array(pickle.load(open('features.pkl', 'rb')))
filenames = pickle.load(open('filenames.pkl', 'rb'))

model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
model.trainable = False

model = tensorflow.keras.Sequential([
    model,
    GlobalMaxPooling2D()
])

img = image.load_img('sample/dress.jpg', target_size=(224,224))
img_array = image.img_to_array(img)
expanded_img_array = np.expand_dims(img_array, axis=0)
preprocessed_img = preprocess_input(expanded_img_array)
result = model.predict(preprocessed_img).flatten()
normalized_result = result / norm(result)

neighbors = NearestNeighbors(n_neighbors=5,algorithm='brute', metric='euclidean')
neighbors.fit(feature_list)

distances, indices = neighbors.kneighbors([normalized_result])

print(indices)

for file in indices[0]:
    temp_img = cv2.imread(filenames[file])
    cv2.imshow('output',cv2.resize(temp_img, (512, 512)))
    cv2.waitKey(0)
```

Main.py:

```
import streamlit as st
import os
from PIL import Image
import pickle
import numpy as np
import tensorflow
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import ResNet50, preprocess_input
from tensorflow.keras.layers import GlobalMaxPooling2D
from sklearn.neighbors import NearestNeighbors
from numpy.linalg import norm

feature_list = np.array(pickle.load(open('features.pkl', 'rb')))
filenames = pickle.load(open('filenames.pkl', 'rb'))

model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
model.trainable = False
```

```
model = tensorflow.keras.Sequential([
    model,
    GlobalMaxPooling2D()
])
st.title('Dripy')

def saveFile(uploaded_file):
    try:
        with open(os.path.join('uploads', uploaded_file.name), 'wb') as f:
            f.write(uploaded_file.getbuffer())
        return 1
    except:
        return 0

def featureExtract(img_path, model):
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    expanded_img_array = np.expand_dims(img_array, axis=0)
    preprocessed_img = preprocess_input(expanded_img_array)
    result = model.predict(preprocessed_img).flatten()
    normalized_result = result / norm(result)

    return normalized_result

def recommend(features, feature_list):
    neighbors = NearestNeighbors(n_neighbors=6, algorithm='brute', metric='euclidean')
    neighbors.fit(feature_list)

    distances, indices = neighbors.kneighbors([features])
    return indices

uploaded_file = st.file_uploader("Upload an image")
if uploaded_file is not None:
    if saveFile(uploaded_file):
        display_img = Image.open(uploaded_file)
        st.image(display_img)
        features = featureExtract(os.path.join("uploads", uploaded_file.name), model)

        indices = recommend(features, feature_list)
        col1, col2, col3, col4, col5 = st.columns(5)
        with col1:
            st.image(filenames[indices[0][0]])
        with col2:
            st.image(filenames[indices[0][1]])
        with col3:
            st.image(filenames[indices[0][2]])
        with col4:
            st.image(filenames[indices[0][3]])
        with col5:
            st.image(filenames[indices[0][4]])
    else:
        st.header("Some error occured in file upload")
```

2.7 GUI and steps to deploy model

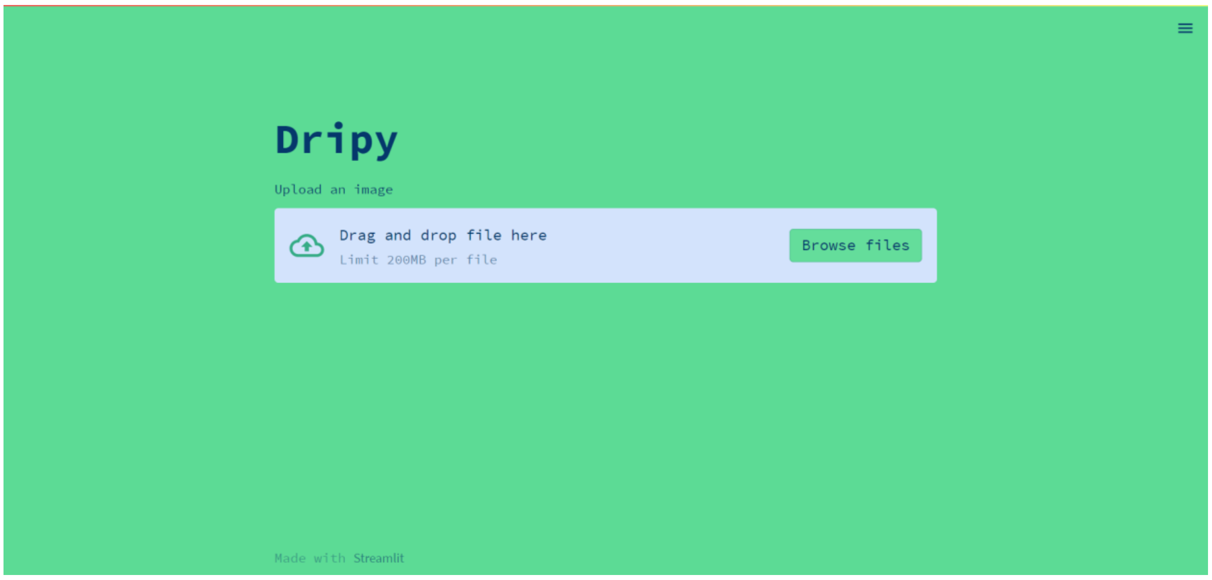


Fig. 2.7.1 Landing Page:

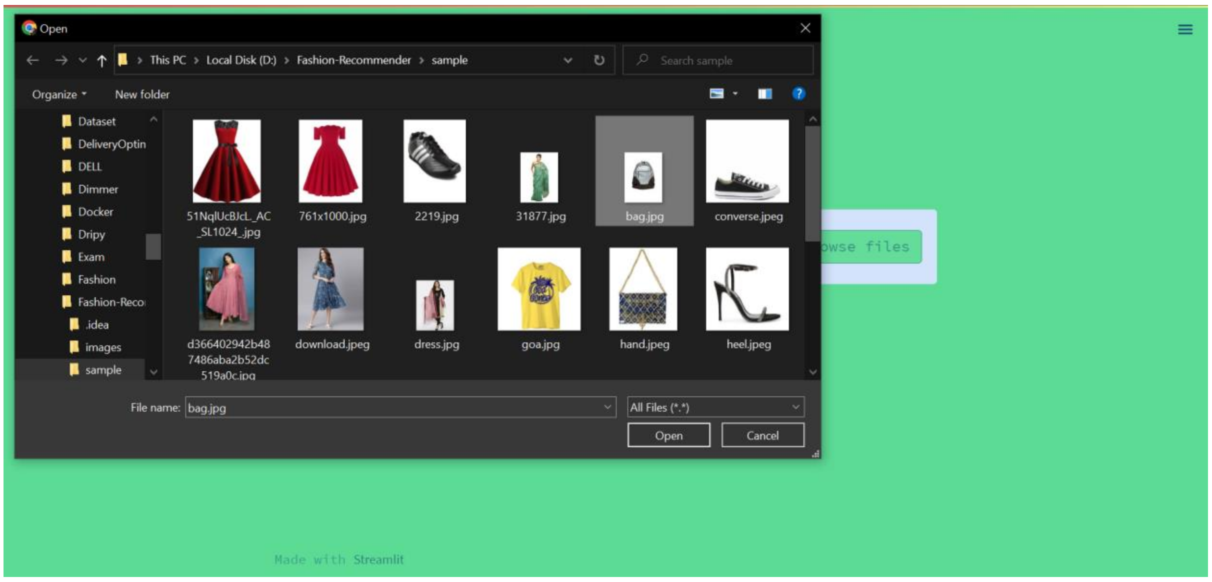
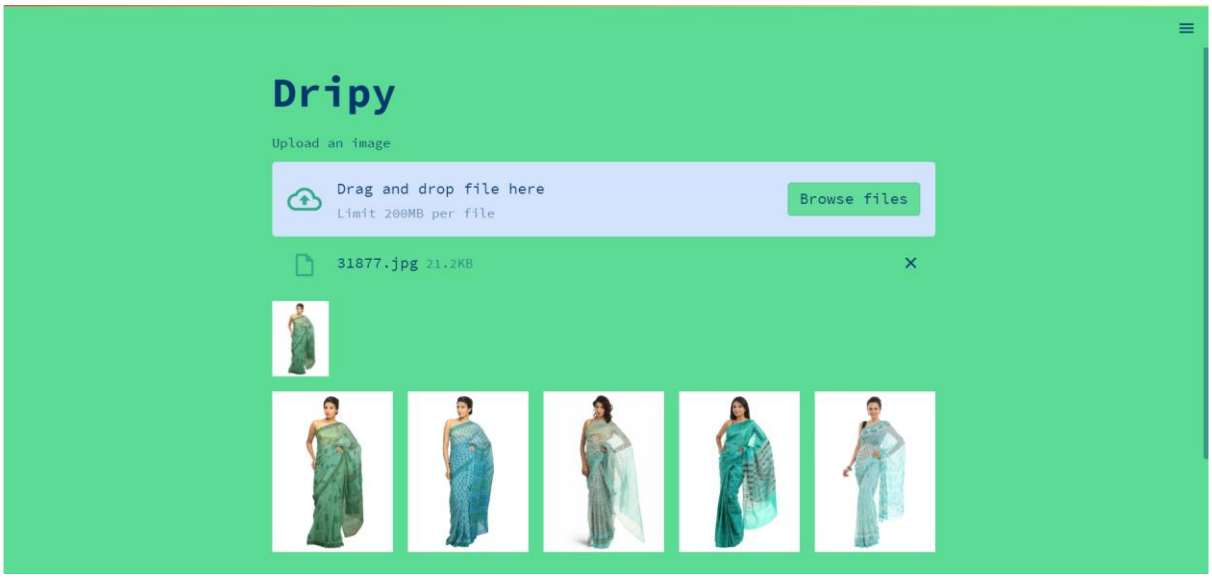


Fig. 2.7.2 Uploading Image as Input



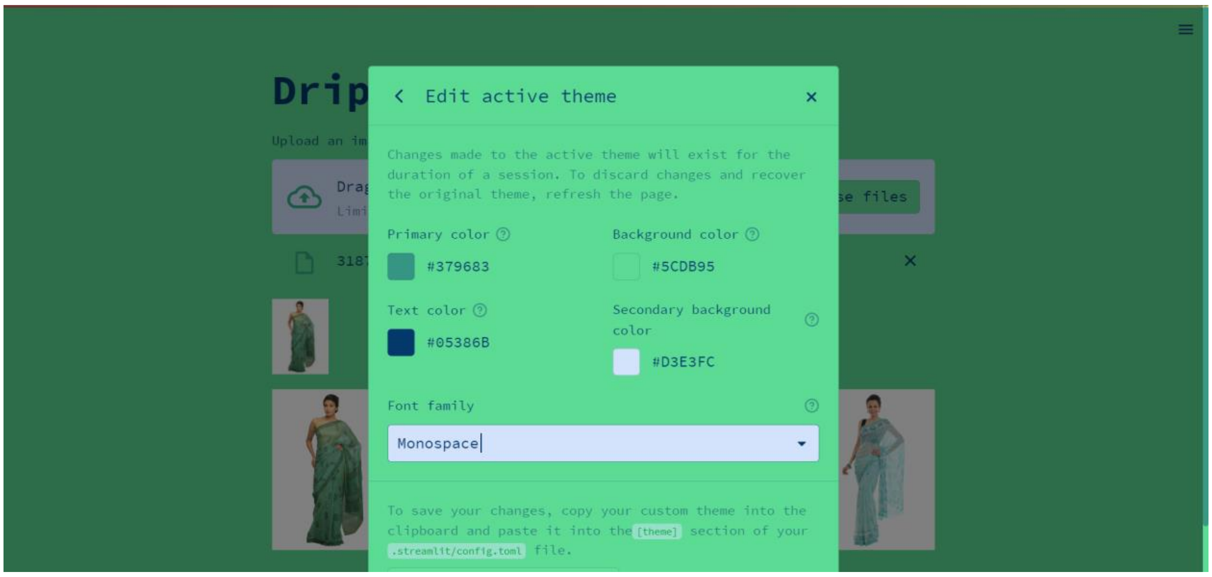
Dripy

Upload an image

Drag and drop file here
Limit 200MB per file

31877.jpg 21.2KB

Fig. 2.7.3 Product Rendering, Feature extraction and Output



Edit active theme

Changes made to the active theme will exist for the duration of a session. To discard changes and recover the original theme, refresh the page.

Primary color: #379683

Background color: #5CDB95

Text color: #05386B

Secondary background color: #D3E3FC

Font family: Monospace

To save your changes, copy your custom theme into the clipboard and paste it into the [theme] section of your .streamlit/config.toml file.

Fig. 2.7.4 Streamlit Theme Mounting

3. CONCLUSION & FUTURE SCOPE

Conclusion:

Retailers can leverage recommendation systems to offer personalized recommendations to consumers by analyzing information gathered from the Internet. These systems allow consumers to quickly find products and services that match their preferences, and advanced algorithms are available that can suggest products based on users' social interactions. Therefore, there has been a surge in interest in incorporating social media images into fashion recommendation systems. This study provides a comprehensive review of fashion recommendation systems, algorithmic models, and filtering techniques, discussing the technical aspects and strengths and weaknesses of various approaches. However, it is important to test the proposed prototypes in real-world settings to assess their accuracy and feasibility, as inaccurate recommendations can have a negative impact on customer experience. Additionally, future research should focus on incorporating time series analysis and accurate categorization of product images based on factors such as color, trend, and style to improve the effectiveness of recommendation systems. The proposed model will feature brand-specific personalization campaigns to provide highly tailored offerings to users, making it a valuable resource for researchers interested in developing augmented and virtual reality features for recommendation systems.

Future Scope:

A fashion recommendation system can have various features depending on the target audience, goals, and available data. More features can be added to FRS in future such as:

1. User Profile: A user profile can be created that includes the user's preferences, body type, and other information that can help the system generate personalized recommendations.
2. Recommendation Engine: The recommendation engine can be designed to generate recommendations based on various criteria, such as user preferences, purchase history, and social interactions.
3. Mix and Match: The system can offer suggestions for combining different fashion items to create a complete look, such as pairing a shirt with a suitable pair of trousers or shoes.
4. Social Sharing: The system can allow users to share their fashion choices and recommendations on social media platforms, which can help in increasing the system's reach and engagement.

Also, an FRS can be incorporated into various real-life examples like E commerce websites, Mobile Apps, Social Media Platforms, Virtual try on apps. Personal styling services, etc. FRS can enhance the shopping experience and offer personalized recommendation to customers.

REFERENCES

[1] Ijraset, A.K. (2019) *IJRASET Journal for Research in Applied Science and Engineering Technology*, *Fashion Recommendation System*. Available at: https://www.ijraset.com/research-paper/fashion-recommendation-system (Accessed: February 20, 2023).

[2] Yijin Lin (2020) *A hybrid FRS based on Multiple Source Data*, *Fashion Recommendation System*. Journal of Data.

[3] Tuinhof, H., Pirker, C. and Haltmeier, M. (1970) *Image-based fashion product recommendation with Deep Learning*, *SpringerLink*. Springer International Publishing. Available at: https://link.springer.com/chapter/10.1007/978-3-030-13709-0_40 (Accessed: March 8, 2023).