India's No.1 E-School Challenge is back!

Flipkart

GRiD 6.0

SHAPING INDIA'S TECHSCAPE

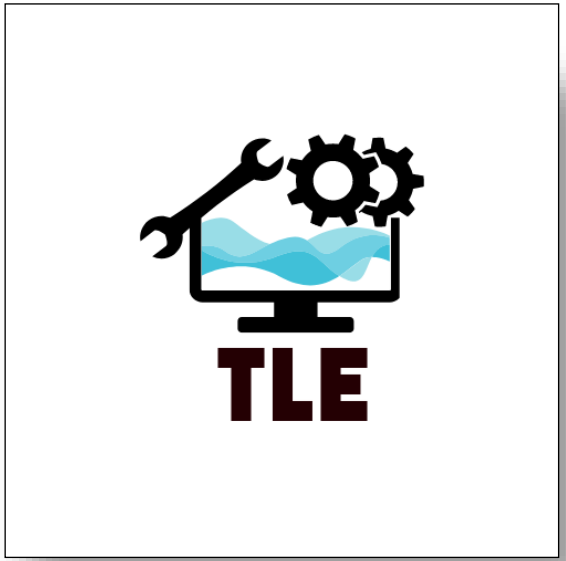
Team Introduction

Title:

AI-Powered Size Chart Generator for Apparel Sellers

Team Name:

TLE



Team Members:

Vaibhav Shukla (Team Leader)

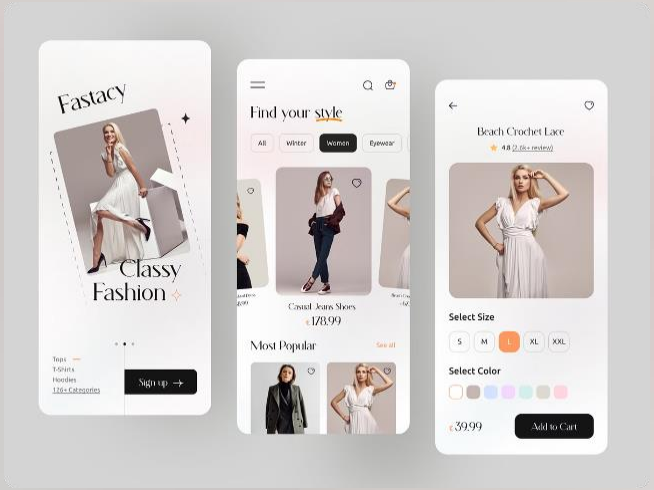
Juhi Dwivedi (Team Member)

College/University:

Amity University

Date:

25-08-2024



AI-Powered Size Chart Generator for Apparel Sellers

Our solution outlines an innovative AI-powered size chart generator designed to designed to revolutionize the apparel shopping experience on Flipkart's e-e-commerce platform. The generator provides accurate size recommendations, recommendations, enhances customer satisfaction, and streamlines the return return process.

Overview of Flipkart's e-commerce platform

Flipkart is India's leading e-commerce platform, offering a vast selection of products across various categories, including apparel. The platform boasts a user-friendly interface, seamless checkout process, and a wide range of payment options.

Product Categories

Flipkart offers a diverse range of products, from electronics and home home appliances to fashion and beauty. beauty.

Customer Base

Flipkart has a large and growing customer customer base, with millions of users users making purchases on the platform. platform.

Seller Network

The platform hosts a vast network of of sellers, offering a wide selection of of products and competitive pricing. pricing.

Executive Summary:

Problem Statement :
High return rates due to incorrect sizing are a significant issue for e-commerce platforms like Flipkart, leading to customer dissatisfaction and increased operational costs.
Current size charts are generic and don’t account for individual body variations.

Proposed Solution:
Implementing an AI-powered size chart generator that provides personalized size recommendations based on user body measurements, previous purchase history, and clustering analysis.

Key Objectives:
Reduce size-related returns.
Improve customer satisfaction.
Enhance the shopping experience by offering tailored recommendations.

Expected Impact:
Lower operational costs, increased customer loyalty, and strengthened competitive edge for Flipkart.

Improved Customer Experience with Tailored Recommendations

Accurate size recommendations are crucial for customer satisfaction in the online apparel market. When customers receive the correct size, they are more likely to be happy with their purchase and return for more.

1

Customer Profile

The generator collects customer data, such as height, weight, and measurements.

2

Style Preferences

The AI model analyzes customer purchase history and preferences to recommend sizes that fit their style.

3

Size Recommendation

The generator provides accurate size recommendations based on the collected data and and algorithms.

Technical Approach:

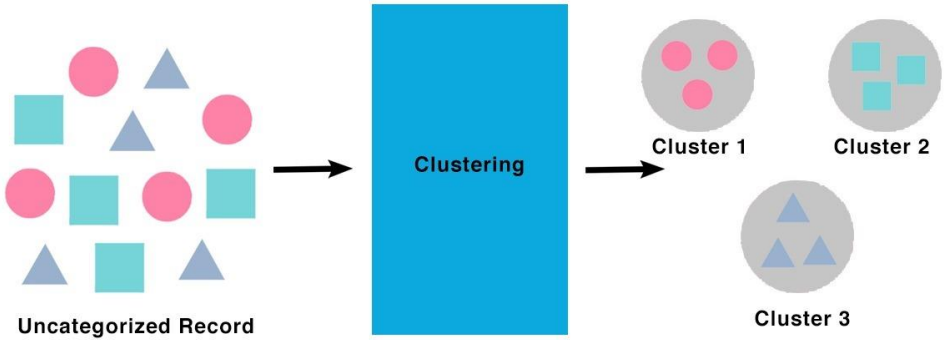
Methodology:

- Clustering Algorithm: Utilizes Agglomerative Clustering to group users with similar body measurements and successful purchase histories.
- Data Integration: Incorporates data from user measurements, purchase history, and return/exchange information to refine clusters and improve size accuracy.

Tools and Technologies:

Model Development

- Python: Core language for model development.
- Pandas: Data manipulation and preprocessing.
- Scikit-Learn: Clustering, model training, and evaluation.
- Joblib: Model saving and loading.
- Matplotlib & Seaborn: Data visualization.
- NumPy: Numerical computing for data processing.



Technical Approach:

Web App Integration
Flask/Django: Backend development and API creation.
React.js: Dynamic and responsive frontend.
Docker: Containerization for consistent deployment.
Gunicorn & Nginx: Production deployment and reverse proxy

Algorithm/Logic:
Prediction Process Overview
Measurements: [70, 150, 38, 32, 40] (height, weight, bust, waist, hips).
Step 1: Data is formatted into a list.
Step 2: The list is passed into the trained model.
Step 3: The model processes the data, comparing it to learned clusters.
Step 4: A cluster is predicted (e.g., Cluster 3).
Step 5: The cluster determines the recommended size.

Why It Works:
Data-Driven: Recommendations are based on real-world data patterns.
Holistic Approach: Considers multiple measurements for accurate sizing.
Scalability: Model can be retrained with new data for continued accuracy.

Technical Approach:

- This code file performs Agglomerative Clustering on a dataset of body measurements to group similar individuals into clusters.
- It first loads the processed data from a CSV file, then applies the clustering algorithm using key measurements like height, weight, and bust/chest size.
- The resulting cluster labels are added to the data, which is then saved to a new CSV file.
- Additionally, the code includes an optional visualization step to plot the clusters based on height and weight.

size-chart-prediction > scripts > clustering.py > ...

```python
import pandas as pd
from sklearn.cluster import AgglomerativeClustering
import matplotlib.pyplot as plt
import os

def load_processed_data(file_path='/Users/juhidwivedi/Desktop/GRiD SDE/size-chart-prediction/data/processed/processed_data.csv'):
    """
    Load the processed data from the CSV file.
    """
    data = pd.read_csv(file_path)
    print(f"Processed data loaded successfully with {data.shape[0]} rows and {data.shape[1]} columns.")
    return data

def perform_agglomerative_clustering(data, n_clusters=5):
    """
    Perform Agglomerative Clustering on the data.
    """
    agg_clustering = AgglomerativeClustering(n_clusters=n_clusters)
    data['Cluster'] = agg_clustering.fit_predict(data[['Height_in', 'Weight', 'Bust/Chest', 'Waist', 'Hips']])
    print("Agglomerative Clustering completed.")
    return data, agg_clustering

def save_clustered_data(data, file_path='/Users/juhidwivedi/Desktop/GRiD SDE/size-chart-prediction/data/processed/clustered_data.csv'):
    """
    Save the clustered data to a CSV file.
    - Create the directory if it doesn't exist
    """
    # Ensure the directory exists
    os.makedirs(os.path.dirname(file_path), exist_ok=True)

    # Save the clustered data
    data.to_csv(file_path, index=False)
    print(f"Clustered data saved successfully at {file_path}")

if __name__ == "__main__":
    # Load the processed data
    df = load_processed_data()

    # Perform Agglomerative Clustering
    df_clustered, clustering_model = perform_agglomerative_clustering(df)

    # Save the clustered data
    save_clustered_data(df_clustered)

    # Optional: Visualize the clusters
    plt.scatter(df_clustered['Height_in'], df_clustered['Weight'], c=df_clustered['Cluster'])
    plt.xlabel('Height (inches)')
    plt.ylabel('Weight')
    plt.title('Agglomerative Clustering')
    plt.show()
```

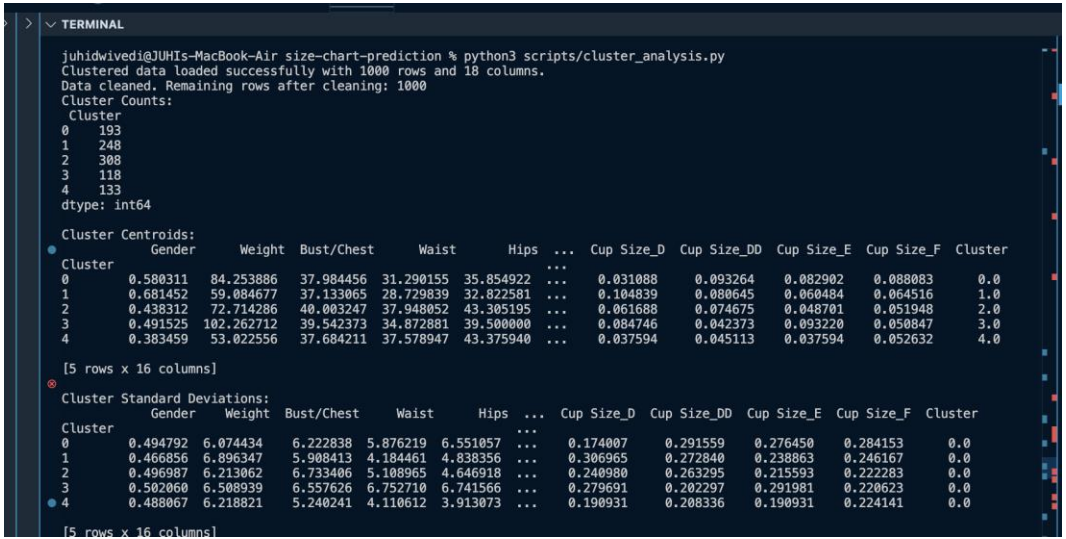
Here, we are loading the data from data_preprocessing.py. It handles the missing values in the given dataset, the outliers, and carries out data cleaning to fit our model. Then the processed data is saved as a csv file in the "data" folder. This function extracts that csv file and uses it for model building.

This function, imports the model from our sklearn library.

After creating clusters on the processed dataset, we now save the clustered output as a csv file in the data/ processed folder. Also, we create visualization for the clusters created.

Technical Approach:

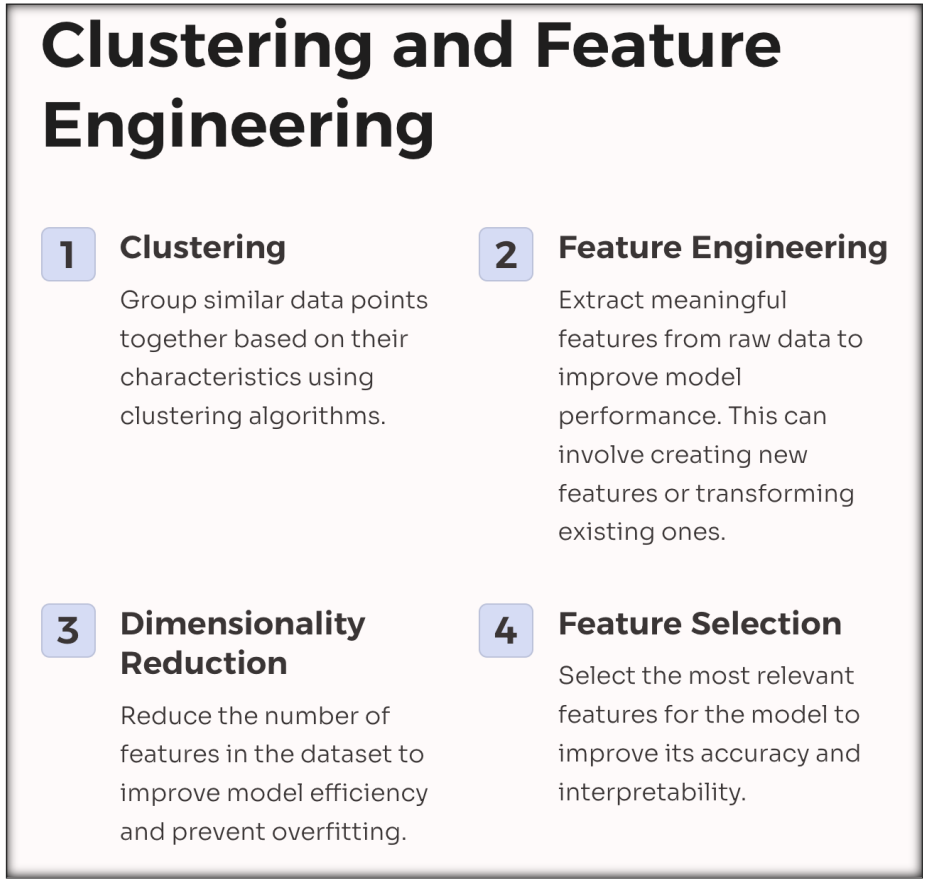
- The cluster_analysis.py file performs statistical analysis and visualization of the clustered data obtained from the Agglomerative Clustering process.
- It loads the clustered data, calculates key statistics such as cluster centroids and standard deviations, and generates visualizations to help understand the distribution and characteristics of each cluster.
- This analysis provides insights into how different body measurements are grouped, aiding in the creation of personalized size charts and enhancing the overall understanding of the dataset.



```
TERMINAL

juhidwivedi@JUHIs-MacBook-Air size-chart-prediction % python3 scripts/cluster_analysis.py
Clustered data loaded successfully with 1000 rows and 18 columns.
Data cleaned. Remaining rows after cleaning: 1000
Cluster Counts:
 Cluster
0    193
1    248
2    308
3    118
4    133
dtype: int64

Cluster Centroids:
          Gender      Weight  Bust/Chest      Waist       Hips  ...  Cup Size_D  Cup Size_DD  Cup Size_E  Cup Size_F  Cluster
Cluster                                                         ...
0       0.580311   84.253886   37.984456  31.290155  35.854922  ...    0.031088     0.093264    0.082902    0.088083      0.0
1       0.681452   59.084677   37.133065  28.729839  32.822581  ...    0.104839     0.080645    0.060484    0.064516      1.0
2       0.438312   72.714286   40.003247  37.948052  43.305195  ...    0.061688     0.074675    0.048701    0.051948      2.0
3       0.491525  102.262712   39.542373  34.872881  39.500000  ...    0.084746     0.042373    0.093220    0.050847      3.0
4       0.383459   53.022556   37.684211  37.578947  43.375940  ...    0.037594     0.045113    0.037594    0.052632      4.0

[5 rows x 16 columns]

Cluster Standard Deviations:
          Gender    Weight  Bust/Chest     Waist      Hips  ...  Cup Size_D  Cup Size_DD  Cup Size_E  Cup Size_F  Cluster
Cluster                                                     ...
0       0.494792  6.074434    6.222838  5.876219  6.551057  ...    0.174007     0.291559    0.276450    0.284153      0.0
1       0.466856  6.896347    5.908413  4.184461  4.838356  ...    0.306965     0.272840    0.238863    0.246167      0.0
2       0.496987  6.213062    6.733406  5.108965  4.646918  ...    0.240980     0.263295    0.215593    0.222283      0.0
3       0.502060  6.508939    6.557626  6.752710  6.741566  ...    0.279691     0.202297    0.291981    0.220623      0.0
4       0.488067  6.218821    5.240241  4.110612  3.913073  ...    0.190931     0.208336    0.190931    0.224141      0.0

[5 rows x 16 columns]
```

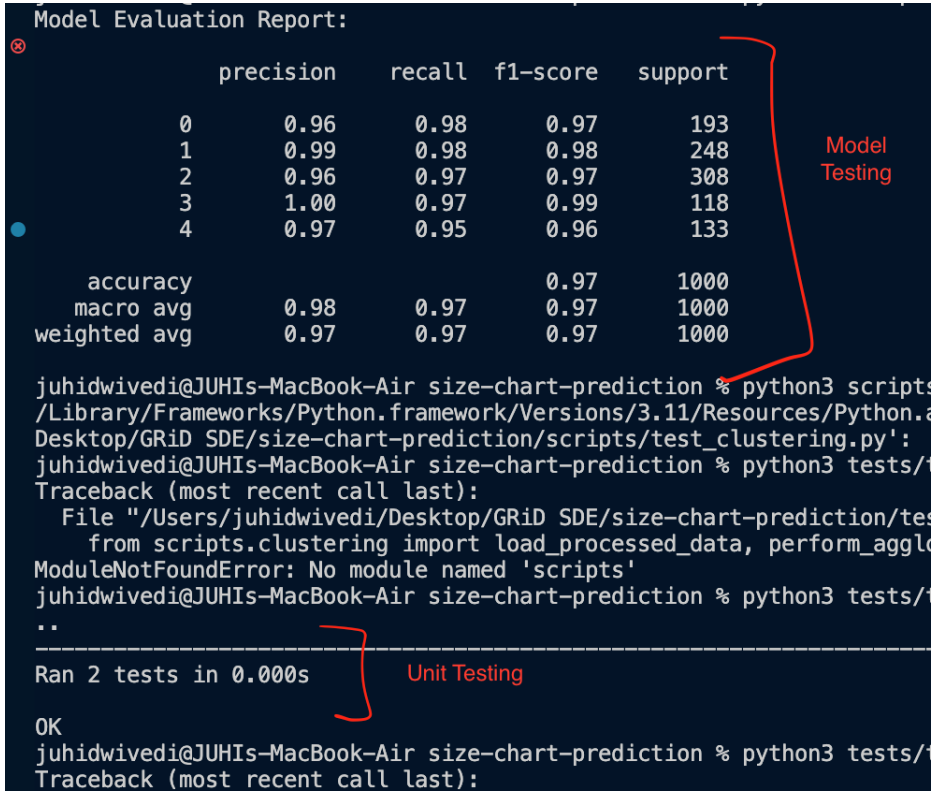
Technical Approach:

- In the context of the AI-Powered Size Chart Generator, clustering is essential for identifying patterns and natural groupings within customer body measurements (such as height, weight, bust/chest size, waist size, and hips size).
- By applying clustering techniques, specifically Agglomerative Clustering, we can group individuals with similar body measurements into distinct clusters.
- These clusters represent different size categories or body shapes, which are crucial for generating personalized size charts.



Technical Approach: Testing Approach

- The testing approach for the AI Powered Size Chart Generator involved a combination of unit testing and validation to ensure the accuracy, reliability, and robustness of the various components in the project.
- The primary goal was to validate that each module and function performed as expected, without introducing errors or inconsistencies.
- The testing files are organized within the tests/ directory in the project structure. Each file in this directory corresponds to a specific aspect of the system, ensuring that all major functionalities are tested.



Reduction in Size-Related Returns Returns

By providing accurate size recommendations, the AI-powered size chart generator aims to aims to reduce the number of size-related returns, resulting in cost savings for both sellers and sellers and customers.

Reduced Shipping Costs

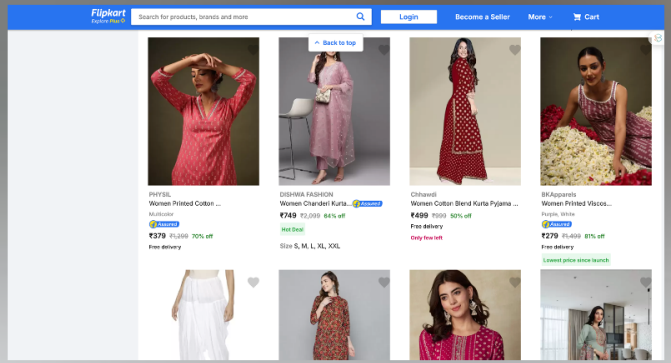
Fewer returns mean fewer shipments, shipments, leading to lower shipping shipping costs for sellers and customers.

Improved Inventory Management

With fewer returns, sellers can better better manage their inventory levels, levels, ensuring that they have the right right sizes in stock.

Enhanced Customer Satisfaction

Customers who receive the correct size are more likely to be satisfied with their purchase, purchase, leading to repeat business and positive reviews.



Integrating the Size Chart Generator into Flipkart's Platform

Integrating the AI-powered size chart generator into Flipkart's platform will require seamless integration with existing systems and user interfaces to provide a smooth and intuitive experience for customers.

1

API Integration

The AI model will be integrated via APIs to access customer data and provide size recommendations.

2

User Interface Design

The generator's interface will be designed to be user-friendly and visually appealing, ensuring a seamless integration with Flipkart's platform.

3

Testing and Deployment

Thorough testing will be conducted to ensure the generator functions correctly and meets the desired objectives.

Key Objectives of the AI-Powered Size Chart Generator

The AI-powered size chart generator aims to address the challenges faced by apparel sellers on e-commerce platforms by providing accurate size recommendations, improving customer satisfaction, and reducing return rates.

Reduce Returns

The generator aims to minimize size-related returns by providing accurate size recommendations.

Enhance Customer Satisfaction

By offering tailored size recommendations, the generator strives to enhance the shopping experience and increase customer satisfaction.

Improve Shopping Experience

The generator provides personalized size suggestions, making the shopping process more convenient and efficient for customers.

Data-Driven Insights

The AI model analyzes data to identify patterns and predict customer preferences for accurate size recommendations.

How the Size Chart Generator Works

The AI-powered size chart generator uses machine learning algorithms to analyze vast amounts of data, including customer measurements, clothing styles, and feedback from previous purchases.

1

Data Collection

The AI model gathers data from customer measurements, garment dimensions, and past purchase history.

2

Algorithm Training

Machine learning algorithms are trained on the collected data to identify patterns and predict accurate size recommendations.

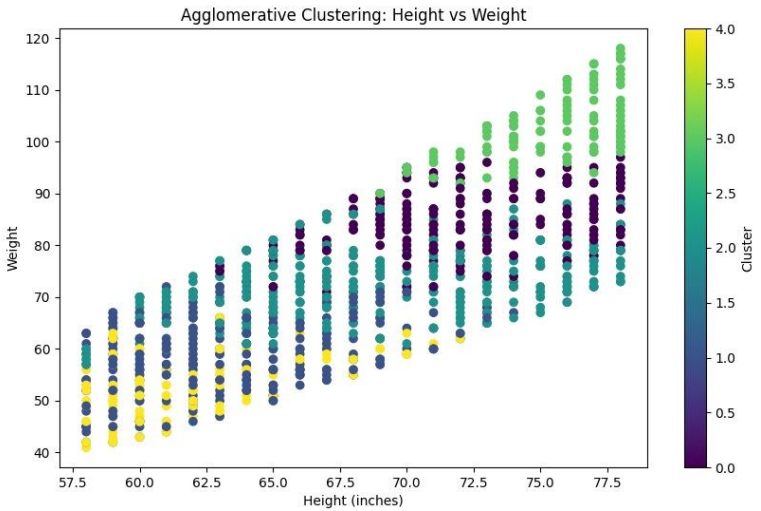
3

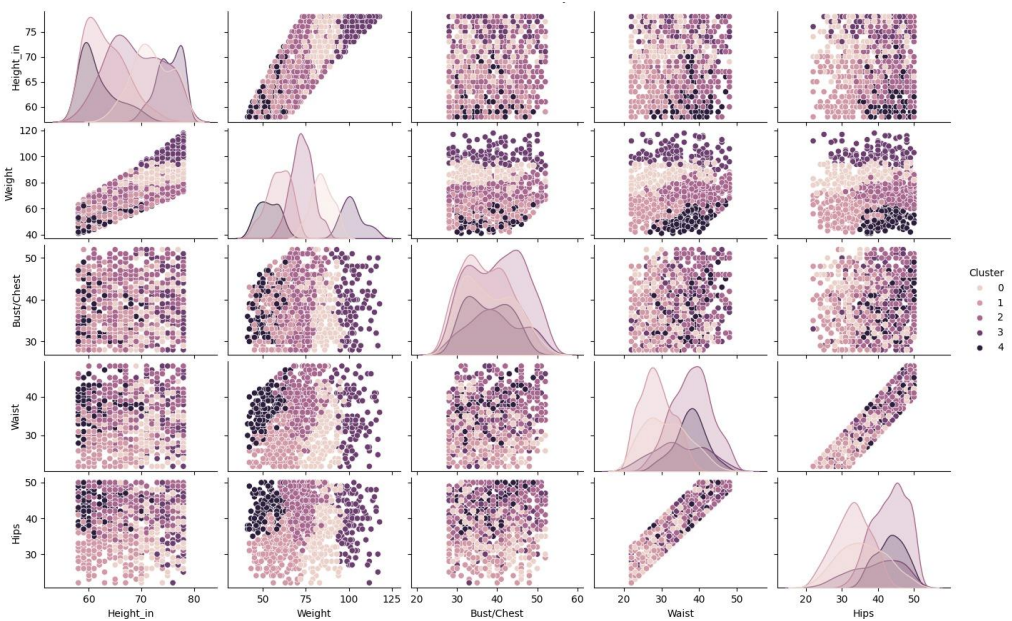
Size Recommendation

Based on the trained algorithms, the generator provides personalized size suggestions to customers during the shopping process.

Results and Analysis:

These visualizations depict how body measurements like height, weight, bust, waist, and hips are grouped into distinct clusters. The patterns reveal correlations between features and how individuals with similar measurements are categorized into specific size groups.





```
juhidwivedi@JUHIs-MacBook-Air size-chart-prediction % python3 scripts/evaluate_model.py
Model Evaluation Report:

              precision    recall  f1-score   support

           0       0.96      0.98      0.97       193
           1       0.99      0.98      0.98       248
           2       0.96      0.97      0.97       308
           3       1.00      0.97      0.99       118
           4       0.97      0.95      0.96       133

    accuracy                           0.97      1000
   macro avg       0.98      0.97      0.97      1000
weighted avg       0.97      0.97      0.97      1000
```

Code suggestions are disabled because there is no user account

The first result shows the model's evaluation report, highlighting an overall accuracy of 97% across all clusters, with strong precision, recall, and f1-scores, indicating robust performance. The second result demonstrates a prediction example where the model successfully classified user measurements into Cluster 3 and recommended a size XL.

```
juhidwivedi@JUHIs-MacBook-Air size-chart-prediction % python3 scripts/predict.py
Model loaded successfully.
Simulating User Interaction...
User's measurements: [70, 150, 38, 32, 40]
/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages/sklearn/base.py:493: UserWarning: X does not ha
ve valid feature names, but RandomForestClassifier was fitted with feature names
  warnings.warn(
Predicted Cluster: 3
Recommended Size: Size XL
juhidwivedi@JUHIs-MacBook-Air size-chart-prediction %
```

Hierarchy of Files

```
size-chart-prediction/
|
├── data/
|   ├── raw/
|   |   └── body_measurements.csv        # Raw dataset
|   ├── processed/
|   |   └── processed_data.csv           # Processed data after cleaning
|   └── external/                        # External datasets if any
|
├── notebooks/
|   ├── EDA.ipynb                        # Jupyter notebook for exploratory data analysis
|   └── model_dev.ipynb                  # Jupyter notebook for model development and experiments
|
├── scripts/
|   ├── data_preprocessing.py            # Script for data cleaning and preprocessing
|   ├── clustering.py                    # Script for clustering and analysis
|   ├── train_model.py                   # Script for training the model
|   ├── evaluate_model.py                # Script for evaluating the model
|   └── predict.py                       # Script for making predictions with the trained model
|
├── models/
|   ├── size_chart_model.pkl             # Trained model saved using joblib
|   └── size_chart_model_pipeline.pkl    # Trained pipeline model
|
├── tests/
|   ├── test_data_preprocessing.py       # Unit tests for data preprocessing
|   ├── test_clustering.py               # Unit tests for clustering
|   ├── test_model.py                    # Unit tests for model training and evaluation
|   └── test_predict.py                  # Unit tests for prediction script
|
├── requirements.txt                     # List of Python dependencies
├── .vscode/
|   ├── settings.json                    # VSCode-specific settings
|   ├── launch.json                      # Configurations for debugging
|   └── tasks.json                       # Automated tasks configuration
|
├── .gitignore                           # Files and directories to ignore in Git
├── README.md                            # Project overview and documentation
└── main.py                              # Main script to run the entire pipeline
```

Challenges and Limitations:

During the development of the AI-Powered Size Chart Generator, we encountered several challenges and limitations, particularly related to data quality and handling missing values.

1. Missing Values and NaN Data: One of the significant challenges was dealing with missing values (NaN) in the dataset. Specifically, the height data, which is a crucial feature for clustering and model prediction, had several missing entries. When we initially attempted to clean the dataset by removing rows with NaN values, we found that it led to a substantial loss of data, effectively wiping out the majority of the dataset. This posed a significant challenge as it would undermine the model's ability to generalize and provide accurate predictions.

2. Handling Missing Data: To address this issue, we implemented strategies to handle missing data without compromising the integrity of the dataset. Instead of removing rows with missing values, we employed techniques such as imputing missing values with the most frequent value (mode) in the column or randomly selecting valid values from the existing data. This approach allowed us to retain the dataset's size and diversity while ensuring that the model could still learn from complete data.

3. Impact on Model Performance: While these strategies allowed us to preserve the dataset, they also introduced a limitation in terms of the potential accuracy and robustness of the model. Imputing missing values can sometimes introduce bias, especially if the missing data is not random. Additionally, the use of random valid values for imputation might not fully capture the true distribution of the missing data, which could affect the clustering outcomes and, subsequently, the size recommendations.

In conclusion, handling missing data, particularly in critical features like height, was a challenging aspect of this project. Although we successfully implemented strategies to mitigate the issue, the approach has its limitations, which could impact the overall performance and accuracy of the model. Moving forward, collecting more complete and accurate data would be essential to further improve the model's effectiveness.

Challenges and Limitations (Contd) :

Limitations:

Data Dependency:
Limitation: The accuracy of size recommendations heavily relies on the quality and quantity of the data used for training. Limited or biased data can lead to less accurate predictions.

Real-Time Performance:
Limitation: While the model is designed for accuracy, the computational cost of real-time predictions might impact performance, especially under heavy user loads.

Clustering, particularly Agglomerative Clustering requires high computation units.Algorithms like KMeans, DBScan, etc were unsuitable here due to their sensitivity to outliers.

Recommendations and Future Work:

Recommendations:

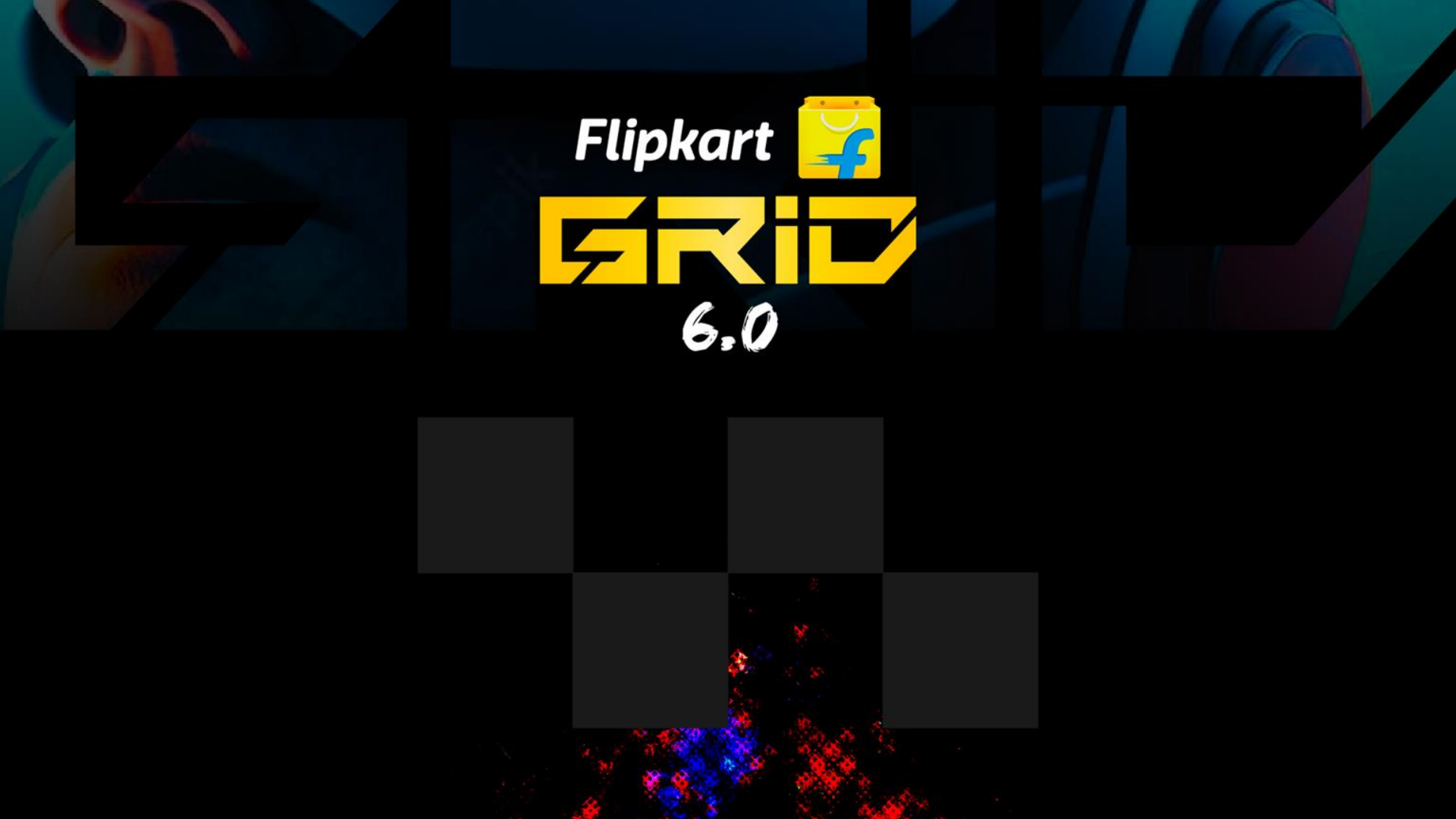
- Conduct further testing on diverse datasets to ensure the model’s adaptability.
- Implement bias detection and mitigation techniques to enhance fairness.

Future Work:

- Develop advanced visualizations and reporting tools for designers.
- Implement continuous learning mechanisms for the model to adapt based on new data.
- Explore cross-cultural adaptability by incorporating regional datasets.

Appendices:

GitHub : Click Here To Visit The Repository


Flipkart
GRID
6.0