CRISP-DM Methodology

Introduction

The *Fashion Recommender and Virtual Try-On Project* combines the use of advanced machine learning with computer vision techniques to enhance the way users interact with online fashion choices. This project also offers integration with the Kling AI for the virtual try-on feature, along with offering personalized recommendations. The development followed the CRISP-DM methodology widely accepted framework for data science projects. This document highlights how each phase of CRISP-DM has been applied to the project.

---

1. Business Understanding

The first step was to clearly define the project's objectives and align them with user needs and market demands. Key goals included:

- **Enhancing User Experience**: Create a platform that provides personalized fashion recommendations based on user input.
- **Driving Engagement**: Enable a virtual try-on feature to increase customer satisfaction and garment fitting capabilities.
- **Business Viability**: Address challenges like style mismatchment and return rates by helping users make informed purchasing decisions.

Key questions addressed during this phase:

- The data needed to personalize recommendations effectively.
- What tool can be used to make a virtual try-on system intuitive and accurate?
- The technological stack, Models, and resources required.

---

2. Data Understanding

Understanding data was essential for ensuring the system's reliability. The project made use of datasets from public sources like Kaggle, which included details about fashion categories, user preferences, and various images of fashion products.

Activities:

- **Data Collection:**
  - Primary dataset: Kaggle fashion product images dataset.
  - Deepfashion dataset with 44,441 garment images for transfer learning.
- **Exploratory Data Analysis (EDA)**: The datasets were examined for completeness, visual attributes like fabric, collar, sleeve, and product metadata and attributes.
- **Virtual try-on data:**
  - Person images for try-on
  - Garment images from recommendation
  - KlingAI API integration requirements

Challenges encountered:

- Inconsistent data formats across sources.

---

3. Data Preparation

To prepare the data for analysis and model training:

- **Data Selection:**
  - Image preprocessing for neural network input
  - Feature extraction using ResNet50 architecture
  - Creation of image embedding database
- **Data Cleaning**: Inconsistent sizing formats were standardized, format conversion, resolution normalization
- **Data Transformation:**
  - Image format standardization for KlingAI API
  - Person image preprocessing
  - Garment image preparation
- **Splitting**: Data was divided into training, validation, and test sets to ensure unbiased evaluation.

The outcomes of this phase ensured that the models received clean, structured, and relevant input for effective learning.

---

4. Modeling

Modeling was split into two components:

1. **Model Selection:**
   - CNN using ResNet50 architecture with transfer learning
   - Additional layers for fine-tuning
   - Nearest Neighbor algorithm for recommendations
2. **Model Building:**
   - Feature extraction using CNN
   - Embedding generation
   - Similarity computation using Cosine Similarity
3. **Virtual Try-On Integration:**
   - KlingAI client implementation
   - JWT authentication system
   - API response handling

---

5. Evaluation

Evaluation ensures the system meets both technical and user-experience goals that include:

- **Results Assessment:**
  - Model accuracy metrics
  - Low error rates
  - Good F-score performance
- **Model Validation:**
  - Testing on the validation dataset
  - Visual similarity verification
  - Recommendation relevance checking
- **Try-On Quality Assessment:**
  - Visual fidelity of try-on results
  - API response time
  - Integration reliability

---

6. Deployment

The final system was deployed as a user-facing application. This phase involved:

- **Implementation:**
  - Web interface using Streamlit
  - Integration with OpenCV for image processing
  - Recommendation system using scikit-learn
- **Tools Used:**
  - TensorFlow for deep learning
  - Pandas for data manipulation
  - PIL for image processing
  - Streamlit for user interface
- **Enhanced Implementation:**
  - Streamlit interface with try-on feature
  - KlingAI API integration
  - JWT token management
- **Additional Tools:**
  - JWT for authentication
  - Requests library for API calls
  - Base64 encoding for image transfer

---

7. Next Steps

While the project has achieved its primary objectives, future work includes:

- Expanding the dataset with real-world user feedback to improve model robustness.
- Integrating more diverse fashion catalogs.
- Enhancing the virtual try-on feature with 3D rendering capabilities.

Conclusion

This CRISP-DM approach ensured a systematic development process, from understanding the business need for fashion recommendations to the final deployment of a user-friendly system that provides end-to-end functionality from product recommendation to virtual try-on, creating a more personalized experience for informed shopping decisions.