CHAPTER 3
RESEARCH METHODOLOGY

1. **Data Science Project Life Cycle**

1.1 Business Understanding

- Objective: To design a recommendation system based on collaborative filtering to predict user preferences and recommend relevant products on an online shopping platform.
- Goal: The goal is to enhance customer experience, increase user engagement, and boost sales conversion rates.
- Problem Statement: Given user-product interaction data, predict user preferences and recommend products they are likely to purchase.
- Success Metrics:
  - Click-Through Rate (CTR)
  - Conversion Rate
  - Precision

1.2. Data Collection and Data Sources

Data Sources:

- Public Datasets:
  - Amazon Product Reviews Dataset (Users' reviews, ratings, and timestamps)
  - RetailRocket Dataset: clicks, add-to-cart and purchases of users
  - Instacart Dataset: This is for users' purchase habits.
- Custom Data Collection:
  - API: Amazon Product Advertising API; Shopify API
  - Web Scraping: E-commerce websites
  - User Interaction Logs: through website analytics tools.

Data Collection Methods:

- Explicit Feedback: Ratings, Reviews, User Surveys
- Implicit Feedback: Clickstream Data, Cart Additions, Purchases.
- Timestamp: Track temporal trends.

Sample Dataset Structure:

User_ID	Item_ID	Rating	Timestamp	Interaction_Type
101	A1	4.5	2024-06-10	Purchase
102	B2	5.0	2024-06-11	Click

1.3. Data Preprocessing

a. Data Cleaning:

- Remove duplicates.
- Handle missing values (e.g., impute missing ratings).
- Filter sparse data (e.g., users with very few interactions).

b. Data Transformation:

- Normalize ratings (e.g., scale between 0–1).
- Convert timestamps to meaningful features (e.g., day of week, hour).
- Encode categorical data if necessary.

c. Feature Engineering:

- User Features: Total purchases, average rating per user.
- Item Features: Average rating, number of interactions per item.
- Temporal Features: Seasonal trends, time-based patterns.

d. Train-Test Split:

- Split dataset into training (80%) and testing (20%).
- Use techniques like Time-Based Splitting if timestamps are important.

1.4. Exploratory Data Analysis (EDA)

- User Interaction Analysis: Active vs inactive users.
- Item Popularity: Most purchased/most viewed products.
- Sparsity Check: Density of user-item interaction matrix.
- Temporal Trends: Sales patterns across different months, days, or hours.

Visualizations:

- User-item interaction heatmaps.
- Top 10 most popular products.
- Rating distribution.

1.5. Model Building

a. Algorithm Selection:

1. User-Based Collaborative Filtering: Find similar users and recommend items liked by them.

2. Item-Based Collaborative Filtering: Recommend items similar to those a user has interacted with.
3. Matrix Factorization Techniques:

    - Singular Value Decomposition (SVD)
    - Alternating Least Squares (ALS)

b. Libraries/Tools:

- Python Libraries: Surprise, scikit-learn, PySpark MLlib
- Deep Learning Models: TensorFlow Recommenders

c. Training the Model:

- Build the user-item interaction matrix.
- Apply selected collaborative filtering algorithms.
- Optimize hyperparameters (e.g., number of latent factors, regularization parameters).

d. Generate Recommendations:

- For each user, predict product scores.
- Recommend Top-K products based on predicted scores.

1.6. Model Evaluation

Evaluate model performance using appropriate metrics:

- RMSE (Root Mean Square Error): Measures prediction accuracy.
- Precision@K: Proportion of top-K recommendations relevant to the user.
- Recall@K: Proportion of relevant items recommended in top-K.
- F1-Score: Harmonic mean of Precision and Recall.

Cross-Validation:

Use k-fold cross-validation to ensure generalization.

1.7. Deployment

- Deploy Model as API: Use tools like Flask or FastAPI.
- Cloud Deployment: AWS, Azure, or Google Cloud.
- Integration: Embed recommendations into the e-commerce website or app.
- Real-Time Recommendations: Update user data dynamically.

Tools for Deployment:

- AWS SageMaker
- Docker Containers
- CI/CD Pipelines: Automate model retraining and deployment

1.8. Monitoring and Maintenance

- Monitor model performance using dashboards (e.g., Grafana, Kibana).
- Track KPIs like CTR, Conversion Rate, Recommendation Clicks.
- Periodic retraining of the model with updated data.
- Handle data and concept drift.

1.9. Documentation and Reporting

- Create a project report outlining objectives, methods, results, and conclusions.
- Provide clear documentation for API integration and system maintenance.
- Prepare a presentation for stakeholders explaining key insights and outcomes.

1.10. Future Improvements

- Add Hybrid Recommendation Models (Content + Collaborative Filtering).
- Implement Context-Aware Recommendations (e.g., location, time, device).
- Explore Deep Learning Techniques for advanced recommendations.

2.Data Collection

2.1. Data Sources

For building a collaborative filtering recommendation system, the dataset typically includes user-item interactions. Below are key data sources:

a. Public Datasets

Amazon Product Reviews Dataset

1. Description: User reviews, ratings, product metadata, and timestamps from Amazon.
2. Data Type: Explicit feedback (ratings) and implicit feedback (purchases).
3. Source: Amazon Customer Reviews Dataset

RetailRocket Dataset

1. Description: User behavior logs, including clicks, add-to-cart, and purchase events.
2. Data Type: Implicit feedback (user actions).
3. Source: RetailRocket Dataset on Kaggle

Instacart Market Basket Analysis Dataset

1. Description: User purchase patterns from online grocery shopping.
2. Data Type: Transactional data (implicit feedback).
3. Source: Instacart Dataset on Kaggle

MovieLens Dataset *(If used for prototype testing)*

1. Description: User ratings on movies.
2. Data Type: Explicit feedback (ratings).
3. Source: MovieLens Dataset

b. Custom Data Collection

If public datasets are insufficient or specific business data is required, the following methods can be used:

APIs from E-commerce Platforms:

1. Examples: Amazon Product Advertising API, Shopify API, eBay API.
2. Collected Data: User IDs, product IDs, ratings, timestamps, purchase data.

Web Scraping:

1. Tools: BeautifulSoup, Scrapy, Selenium.
2. Collected Data: Product descriptions, ratings, reviews, prices, and user comments.

User Interaction Logs:

1. Source: Web analytics tools (e.g., Google Analytics, Mixpanel).
2. Collected Data: Clickstream data, time spent on product pages, purchase frequency.

Surveys and Questionnaires:

1. Method: Directly asking users for product preferences and satisfaction scores.
2. Collected Data: Explicit ratings, preferences, and feedback.

E-commerce Database Records:

1. Data sourced from the platform's transactional database.
2. Collected Data: Purchase history, cart abandonment patterns, wish list items.

2.2. Data Collection Methods

a. Explicit Feedback Collection

- Definition: Data where users explicitly express their preferences.
- Examples: Product ratings (e.g., 1–5 stars), written reviews, satisfaction surveys.
- Pros: Clear understanding of user intent.
- Cons: Limited data as not all users provide explicit feedback.

b. Implicit Feedback Collection

- Definition: Data inferred from user behavior and interactions.
- Examples: Clicks, page views, cart additions, purchases, time spent on products.
- Pros: Abundant data from regular browsing and purchasing activity.
- Cons: Ambiguity in user intent (e.g., a click doesn't always mean interest).

.c. Temporal Data Collection

- Definition: Capturing timestamps for each interaction.
- Examples: Date and time of purchases, seasonal trends, peak shopping hours.
- Pros: Enables time-based trend analysis.
- Cons: Requires additional processing for meaningful insights.

d. Social and Demographic Data Collection *(Optional)*

- Definition: Collect user demographics and social connections if relevant.
- Examples: Age, gender, location, social network influence.
- Pros: Can personalize recommendations further.
- Cons: Privacy concerns and regulatory compliance (e.g., GDPR).

2.3. Example Dataset Structure

User_ID	Item_ID	Interaction_Type	Rating	Timestamp
101	P123	Purchase	4.5	2024-06-10
102	P456	Click	-	2024-06-11
103	P789	Add_to_Cart	-	2024-06-12

Key Columns Explanation:

- User_ID: Unique identifier for users.
- Item_ID: Unique identifier for products.
- Interaction_Type: Type of interaction (click, purchase, add-to-cart).
- Rating: User-provided rating (if explicit feedback is available).
- Timestamp: Date and time of interaction.

2.4. Challenges in Data Collection

- Data Sparsity: Many users interact with only a small subset of items.
- Cold Start Problem: New users or new items have little to no interaction data.
- Privacy Concerns: Collecting and storing sensitive user data must comply with regulations (e.g., GDPR, CCPA).
- Data Integration: Combining data from multiple sources can be challenging.

3.Data Preprocessing Steps

Data preprocessing is a critical phase in building a collaborative filtering recommendation system. It ensures that the data is clean, consistent, and suitable for feeding into the model. Below are the key steps, broken down into Cleaning, Transformation, and Feature Engineering.

3.1. Data Cleaning

Objective: Remove inconsistencies, handle missing values, and ensure data integrity.

a. Handle Missing Values

- User_ID or Item_ID Missing: Remove rows with missing identifiers as they are critical for collaborative filtering.
- Missing Ratings (Explicit Feedback):
  - Fill with the average rating of the user or item.
  - Drop rows if the percentage of missing ratings is minimal.
- Missing Interaction Data (Implicit Feedback): Infer user interest from other actions (e.g., clicks, purchases).

Example:

User_ID	Item_ID	Rating	Interaction_Type	Timestamp
101	P123	4.5	Purchase	2024-06-10
102	P456	-	Click	2024-06-11

Action: Fill missing rating (-) with average user rating.

b. Remove Duplicates

- Eliminate duplicate records of user-item interactions.
- Ensure only unique combinations of User_ID and Item_ID exist.

c. Filter Sparse Data

- Inactive Users: Remove users with very few interactions (e.g., less than 3 purchases or clicks).
- Unpopular Items: Remove items with very few interactions.

Threshold Example: Remove users/items with fewer than 5 interactions.

d. Handle Outliers

- Remove unrealistic ratings (e.g., a rating of 100 on a 5-point scale).
- Analyze anomalies in user behavior (e.g., excessive purchases in a very short time).

3.2. Data Transformation

Objective: Ensure data is in the correct format for modeling.

a. Normalize Ratings (if using explicit feedback)

- Scale ratings between 0–1 or standardize them (mean = 0, std dev = 1).
- Techniques: Min-Max Scaling, Z-score Standardization.

Formula (Min-Max Scaling):

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Before Normalization:

User_ID	Item_ID	Rating
101	P123	2

User_ID	Item_ID	Rating
102	P456	5

After Normalization (0–1 range):

User_ID	Item_ID	Rating
101	P123	0.25
102	P456	1.0

b. Convert Timestamps to Meaningful Features

- Extract relevant time-based features from timestamps:
  - Day of week (e.g., Monday, Tuesday).
  - Month of year (e.g., December for holiday season).
  - Time of day (e.g., Morning, Afternoon, Evening).
- Identify temporal trends in interactions.

Example:

User_ID	Timestamp	Day_of_Week	Time_of_Day
101	2024-06-10	Monday	Afternoon

c. Interaction Matrix Construction

- Build a User-Item Interaction Matrix where:
  - Rows: Users
  - Columns: Items
  - Values: Ratings (explicit) or interaction frequency (implicit).

Example Interaction Matrix:

	Item_1	Item_2	Item_3
User_1	5	0	3
User_2	0	4	0

d. Encoding Categorical Variables (if applicable)

- Encode product categories, brands, or user demographics if included in the dataset.
- Techniques: One-Hot Encoding, Label Encoding.

3.3. Feature Engineering

Objective: Create additional meaningful features to improve the recommendation quality.

a. User-Level Features

- Average Rating Per User: Mean rating given by each user.
- Number of Interactions Per User: Total purchases, clicks, or ratings.
- Recency of Interaction: Time since the last interaction.

Example:

User_ID	Avg_Rating	Total_Interactions	Last_Interaction
101	4.2	15	3 days ago

b. Item-Level Features

- Average Rating Per Item: Mean rating received by an item.
- Item Popularity: Number of times an item was interacted with.
- Category or Brand Impact (if available): Popularity within a specific category or brand.

Example:

Item_ID	Avg_Rating	Total_Interactions	Category
P123	4.7	200	Electronics

c. Temporal Features

- Identify peak shopping hours, days, or seasons.
- Calculate interaction frequency trends (e.g., shopping spikes during sales or holidays).

d. Implicit to Explicit Feedback Conversion

- Assign weights to implicit actions:
  - Click: 0.2

- Add to Cart: 0.5
- Purchase: 1.0

Example:

User_ID	Item_ID	Interaction_Type	Interaction_Score
101	P123	Click	0.2
102	P456	Purchase	1.0

4. Train-Test Split

- Random Split: Split data into 80% training and 20% testing.
- Time-Based Split (if using temporal data): Train on older data, test on recent interactions.
- Cross-Validation: Use k-fold cross-validation for model robustness.

5. Final Prepared Dataset Example

User_ID	Item_ID	Rating	Interaction_Score	Avg_User_Rating	Avg_Item_Rating	Day_of_Week	Time_of_Day
101	P123	4.5	1.0	4.2	4.7	Monday	Afternoon