Software Design Documentation for

Recommendation System

Prepared by:
Abineet Thampi

Introduction

An Investment Recommendation System that aims to provide personalized stock recommendations to users based on their investment preferences. The system utilizes various machine learning techniques, including dimensionality reduction, clustering, and collaborative filtering, to analyze financial data and generate tailored recommendations.

Existing Implementations

Robo-advisors

Robo-advisors are digital platforms that provide automated, algorithm-driven financial planning services with minimal human supervision. Examples include Betterment, Wealthfront, and Ellevest. These platforms use machine learning algorithms to analyze user preferences, risk tolerance, and financial goals to provide tailored investment recommendations and portfolio management services.

Stock Screeners

Stock screeners are tools that allow investors to filter and search for stocks based on specific criteria, such as financial ratios, market capitalization, or industry. Examples include Yahoo Finance Stock Screener, Finviz, and Zacks Investment Research. While these tools do not provide personalized recommendations, they help investors narrow down their search and identify potential investment opportunities based on their preferences.

Investment Research Platforms

Investment research platforms provide comprehensive financial data, analysis, and insights to help investors make informed decisions. Examples include Morningstar, S&P Capital IQ, and Bloomberg Terminal. These platforms often incorporate machine learning algorithms to analyze large datasets, identify trends, and generate investment ideas and recommendations.

AI Technologies and Models Used

The key AI technologies and models used in this implementation are:

- Dimensionality Reduction
    - Principal Component Analysis (PCA): Used to reduce the dimensionality of the financial data features.

- Clustering
    - K-Means Clustering: Used to group companies into clusters based on their financial characteristics.

- Collaborative Filtering
    - Matrix Factorization using SVD: Employed to generate recommendations based on the relationships between companies and user preferences. SVD is used for dimensionality reduction.

- Regression
    - Ridge Regression: Used to build a model that maps user preferences to the identified clusters.

- Content-based Recommendations
  - Cosine Similarity: Utilized to find companies that are most similar to the user's preferences.

Dataset Description

The code uses a financial dataset, "2017_Financial_Data.csv" from Kaggle, which contains various financial metrics and ratios for a set of companies. The dataset includes features such as revenue, profit, assets, liabilities, cash flow, and valuation ratios.

Dependencies

Software Dependencies:

- **Pandas**: For data manipulation and analysis.
- **NumPy**: For numerical computations.
- **Scikit-learn**: For machine learning algorithms such as PCA, K-Means, and Ridge Regression.
- **Joblib**: For saving and loading the model.
- **SciPy**: For sparse matrix decomposition.

Hardware Dependencies:

- CPU: Multi-core processor
- RAM: 8 GB RAM
- Storage: SSD storage
- **Optional**: GPU for intensive tasks.

Detailed Workflow

1. Initialization:
   - The recommendation system is initialized, which likely involves loading necessary models and data.
2. Get User Preferences:
   - The system retrieves investor preferences using the function `get_investor_preferences()`.
3. Prepare User Preferences List:
   - The user preferences are wrapped in a list to be processed.
4. Evaluate Personalization:
   - The system evaluates the personalization of recommendations for the given user preferences.
   - For each user preference in the list, get the top 10 recommendations and append them to `all_recommendations`.
   - Flatten the list of recommendations.
   - Calculate the number of unique recommendations.

- Calculate the personalization score.

5. Generate Recommendations:
    - The system generates the top 10 recommendations for the user.
6. Print Recommendations:
    - The top recommendations are printed to the console.
7. Save the Model:
    - The trained recommendation model is saved to a file.

Usage Instructions

1. **Install the required dependencies**: Ensure that Python and the necessary libraries (Pandas, NumPy, Scikit-learn, SciPy, Joblib) are installed.
2. **Load the financial dataset**: Place the "2017_Financial_Data.csv" file in the appropriate directory.
3. **Run the main script**: Execute the main script, which will perform the following tasks:
    - Load the financial dataset
    - Build the investment recommendation model
    - Evaluate the model's performance
    - Prompt the user for investment preferences
    - Generate personalized stock recommendations based on the user's preferences
    - Save the trained model for future use
4. **Provide investment preferences**: When prompted, enter your investment preferences, including risk tolerance, growth preference, value preference, size preference, dividend preference, dividend yield preference, and debt-to-equity preference.
5. **Receive personalized recommendations**: The system will generate a list of top stock recommendations based on your preferences.

Step-by-Step Process:

1. **Load and preprocess the dataset**: The feature_engineering method is called to select relevant features, handle missing values, and perform feature engineering on the financial dataset.
2. **Normalize features**: The StandardScaler is used to normalize the features.
3. **Dimensionality reduction**: Principal Component Analysis (PCA) is applied to reduce the dimensionality of the normalized features.
4. **Clustering**: K-Means clustering is performed on the reduced features to group the companies into clusters based on their financial characteristics.

5. **Regression**: A Ridge regression model is trained to map the reduced features to the identified clusters.
6. **Collaborative Filtering**: A user-item matrix is created using the priceEarningsRatio feature, and Singular Value Decomposition (SVD) is used to perform matrix factorization for collaborative filtering recommendations.
7. **Content-based Recommendations**: Cosine similarity is calculated between the user's preferences and the reduced features to provide content-based recommendations.
8. **Recommendation Combination and Ranking**: The collaborative filtering and content-based recommendations are combined, and the final recommendations are ranked based on a weighted score.

The get_recommendations method is used to generate personalized stock recommendations for a given user's preferences by:

1. **Mapping user preferences to a cluster** using the trained K-Means model.
2. **Generating collaborative filtering recommendations** based on the user's cluster.
3. **Generating content-based recommendations** based on the similarity between the user's preferences and the reduced features.
4. **Combining and ranking the collaborative filtering and content-based recommendations**.

Implementation Process

The implementation process can be summarized as follows:

1. Load and Preprocess the Dataset: The code loads the financial dataset, selects relevant features, and handles missing values.

2. Feature Engineering: The code performs feature engineering, including normalizing the features and handling outliers.

3. Dimensionality Reduction: The code uses PCA to reduce the dimensionality of the financial data features.

4. Clustering: The code applies K-Means clustering to group the companies based on their financial characteristics.

5. Collaborative Filtering: The code uses matrix factorization to generate recommendations based on the relationships between companies and user preferences.

6. Content-based Recommendations: The code calculates the cosine similarity between the user's preferences and the company features to provide additional recommendations.

7. Recommendation Combination and Ranking: The code combines the collaborative filtering and content-based recommendations, and ranks the final recommendations based on a weighted score.

8. Model Evaluation: The code evaluates the clustering performance using the Silhouette score and the personalization of the recommendations.

9. Model Saving and Loading: The code provides functionality to save and load the trained investment recommendation model.

Training and Evaluation Metrics

Training Metrics

Silhouette Score: The code evaluates the clustering performance using the Silhouette score, which measures the quality of the clustering. The Silhouette score obtained is 0.8651589754708124, indicating a well-defined clustering structure.

Evaluation Metrics

Personalization Score: The code calculates a personalization score to measure the diversity and personalization of the recommendations for different user preferences. The personalization score obtained is 1.0, suggesting that the recommendations are highly personalized for the given user preferences.

Findings

The code provides a comprehensive investment recommendation system that leverages various machine learning techniques to analyze financial data and generate personalized stock recommendations. The system's ability to cluster companies, perform collaborative filtering, and provide content-based recommendations allows it to generate tailored recommendations for users based on their investment preferences.

The evaluation metrics, such as the Silhouette score and the personalization score, indicate that the recommendation system is effective in grouping companies based on their financial characteristics and providing highly personalized recommendations.

References

1. https://medium.com/@khang.pham.exxact/what-are-recommendation-systems-6bb5036042db
2. https://muvi-com.medium.com/the-difference-between-collaborative-and-content-based-recommendation-engines-ade24b5147f2
3. “How to Design and Build a Recommendation System Pipeline in Python (Jill Cates)”: https://www.youtube.com/watch?v=v_mONWiFv0k