1. Solution and Implementation Details:

* Platform and Language used: GNU Octave 5.2.0
* The implementation uses Collaborative Filtering to predict the missing rating values of the user-item pairs. Initially, the given data is divided into 2 parts: Train and Test. The splitting is done in a way where 70% of every user’s data is put into training set and the rest 30% is put into the testing set.
* The system uses the concept of Bagging to avoid overfitting and obtain high accuracy results. The training set samples are randomly bagged with replacement into K bags of initial training set size. Every data sample that is not included in the Kth bag (because of replacement=true and uniform bag size) acts as the testing set for that respective bag to calculate the Out-of-Bag error (OOB error).
* For each bag, the following steps are performed:
  + The training and the testing sets of that bag are converted into 1682x943 dimensional matrices.
  + The parameters required for calculating the predictions, Theta and X are randomly initialized. These random initializations lie in a range of (-1,1).
  + With the random initialized parameters and the training set (which now has been converted into 1682x943 matrix), we calculate the associated cost function J with respect to Theta.
  + Next step includes the usage of ***fmincg*** algorithm, which is an advanced optimization algorithm that helps in reducing the cost of the function, thereby returning the optimal parameters Theta and X.
  + The prediction matrix, p is then calculated using the dot product of both the parameters,
    - p = X \* Theta’
* The prediction matrix is then converted into vector form and used for calculation of RMSE with respect to the test set of that particular bag (OOB error).
* As we can see, the RMSE for each bag is moderate (between 1-3%).
* After we get the predictions from each bag, we calculate majority votes for a rating of a particular user-item pair. The prediction value with maximum votes is considered for the final prediction.
* This final prediction matrix is then compared with the actual test data that was split initially and the final RMSE is calculated. This final RMSE gives you a lot less RMSE as compared to the individual bagged RMSE’s (between 0.02-0.9%).
* These results with the actual ratings provided in the dataset given with the predicted ratings are then written into a text file (output.txt).

2. Solution for Cold Start Issue:

* A cold start issue occurs when a new user has never given a rating, or a new item has never been rated before.
* When an item is fairly new, a similar item (i.e. an item that has a feature vector closest to the feature vector of new item) from the dataset is found and a Baseline prediction equivalent to the similar item is used to generate the ratings.
* For a new user, a rating equivalent to the respective item’s average rating is considered.
* Similarly, all the missing values with no user/item ratings at all in the Rating matrix are filled using the above method.
* This method for new item demonstrates Item-Item interaction and helps in designing an item-based model.
* Whereas the solution for the new user cold start issue needs more of improvisation like initial user preference when entering into the system or more contextual based information of the new user.

3. Examples of the Solution for Matrix Completion Problem:

* *Matrix Completion Schemes to find Position of devices in a Wireless Network:*
  + In a network of wireless devices, we can use matrix completion to find the position of the devices.
  + In this problem, if we know the distance between all pairs of devices, we can find the position of any device as long as they follow the rigid motion by applying a technical called Multidimensional Scaling.
  + In the real world, due to problems such as interference, we cannot get the relative distances between positions of all the devices. The best we can have without approximations are the nearby devices.
  + The distances of the near-by located devices are stored in a matrix.
  + This matrix is completed using rank matrix factorization to get the positions of devices in the rigid motion and then applied to multidimensional scaling.
  + **Two dimensions:** Devices as rows and columns. If we have 10 devices, a matrix of 10x10 would be required for this problem.
  + **Entries of Matrix:** The entries of the matrix represent the Euclidean distance between the 2 devices corresponding to the row and column.
* *Predicting Traffic on all Links in a Network:*
  + Since monitoring all the links in a network would be costly and time-consuming, we can monitor a subset of a network at once and another subset at another time interval.
  + In this way, we can fill the matrix partially with monitored data and predict the rest of it using matrix completion.
  + **Two Dimensions**: Row: Each link in the network and Column: Each time interval.
  + **Entries of Matrix**: Volume of traffic at a particular link at the associated time interval.

4. Mini Survey for Papers on Recommender Systems:

* *A collaborative approach for research paper recommender system [2017]*
  + This paper provides a methodology to recommend personalized recommendations of research papers using publicly available contextual metadata to infer the hidden relations that exist between them.
  + It provides personalized recommendations irrespective of user’s expertise and research field.
  + In their approach, a candidate paper is considered only if it cited any of the Target paper’s references and there exists another paper that cited both the candidate and the Target paper simultaneously.
  + An assumption is made by the authors, that if there is a co-occurrence of target paper and the qualified candidate paper, then there exists a similarity between them. Based on this assumption, the top N similar papers are recommended.
* *On the Difficulty of Evaluating Baselines [2019]*
  + This paper focuses on the comparisons of various algorithms used recently for recommender systems and their ways of computing the baselines.
  + Their observations state that the baseline results of various publications in recent years are sub-optimal and a fine-tuned setup of Vanilla Matrix Factorization baseline, they can outperform that reported results of newly proposed algorithms.
  + The authors believe that the findings of the baseline results in most of the papers were questionable unless they were calculated on a standard benchmark, where the baselines were tuned by the research community.
  + They propose a stronger baseline based on modelling implicit activity and the act of considering temporal effects.
  + The paper concludes showing, how judging the work can be an arduous process and the empirical findings can only be trusted if there are standardized benchmarks and considerable tuning effort by community.
* *Hybrid Recommender Systems: A Systematic Literature Review [2019]*
  + This paper compares different hybrid models of recommender systems and addresses the issues with data mining and recommendation solution techniques.
  + The data mining and ML techniques studied and compared in this paper include, K-NN, Clustering, Association Rules, Fuzzy Logic, Matrix Manipulation and other.
  + The authors also compare various other recommendation techniques combined with Collaborative Filtering, making it a hybrid model.
  + The study shows that the hybrid model involving Collaborative Filtering and K-NN is the most successful and widespread hybrid RS.

5. Open Issues on Recommender Systems:

* ***Problem 1***: Data Sparsity 🡪 This issue occurs when we have a very low user-item ratings. Only a few users have rated and the number of items in the catalogue are very high. This affects out similarity deducing capabilities between users, thereby reducing the recommendation accuracy.
  + ***Solution:*** Usually, the final ratings are calculated by merging the ratings of same item by other users (average of ratings by other users for that item) or using ratings of similar users on other similar items. Or the baseline prediction method that was introduced in the Bellkor’s Netflix Winning Algorithm.
* ***Problem 2***: Accuracy🡪 There has a been a wide range of discussion about the scope of improvement for accuracy in recommender systems. This problem usually co-occurs with data sparsity. The more the sparsity, more the compromises being made with regards to accuracy.
  + ***Solution***: An interesting study I came across uses Bayesian Network Model where user, item and feature nodes are used to combine collaborative and content-based filtering to improve the accuracy. Other means to improve accuracy is by involving the context-based data and history learnings of a system w.r.to users and items and find their behavioral patterns.
* ***Problem 3:*** Scalability 🡪 The ratio of number of users to number of items in a system is an important aspect for better predictions. The model that is designed for some 100 users to predict ratings would not behave the same for a million users, unless it is explicitly designed to.
  + ***Solution:*** One solution combines modified Pearson’s coefficient for content-based filtering with distance-to-boundary collaborative filtering. The nearest and the farthest neighbors are determined for all the users in order to reduce dataset. This improves scalability and also deals well with sparsity of matrix.
* ***Problem 4:*** Diversity🡪 Giving similar kind of recommendation to maximum users may result in another problem called popularity bias. Users with rare and different characteristic features may not be benefitted by the popular recommendations too much.
  + ***Solution:*** K-Furthest neighbor and the Inverted Neighborhood model of K-NN is used to create more diversified recommendations.
* ***Problem 5:*** Time Complexity & Compute Intensity🡪 This particular problem I faced while implementing my solution for RS for bagging in order to improve accuracy. I faced a higher computational overload and a lot more time for calculating the majority predictions from every bag.
  + ***Solution:*** Other CPU-efficient algorithms with optimizations can be used.