The Chinese University of Hong Kong, Shenzhen

DDA4260: Networked Life

Project Report

Author:	*Student Number:*
Liang Ruoyu	121090311
Zhao Rui	121090820
Xu Bowen	121090661

December 13, 2023

Contents

1 Abstract

The project's goal is to introduce new algorithms and techniques to provide custom recommendations to users. This report includes theory, implementation and analysis of two methods, **Linear Regression** and **Restricted Boltzmann Machine**. RBM is extended with **momentum**, **adaptive learning rate**, **early stopping**, **regularization**, **mini-batch**, **biases**, and **Neighbourhood method**.

2 Dataset

The dataset provided in two parts, the training set, containing 80% of the data and the validation set, with 20%. Both files have three columns: movie ID, user ID and rating ID. For example, a row of 5,3,4 means that user 3 has given to movie 5 a rating of 4.
Two datasets in the folder (code): training.csv: imported to a matrix by getTrainingData function in projectLib file. validation.csv: imported to a matrix by getValidationData function in projectLib file. **Keywords:** recommendation system, linear regression, restricted Boltzmann machine

3 Theory

3.1 Linear Regression

Linear regression is a supervised machine learning algorithm used for predicting the relationship between two variables. The goal of linear regression is to find the appropriate parameters b by minimizing a loss function.
In this project, linear regression is applied for rating prediction, the program is saved in LinearRegression.py.
When we implement a linear regression algorithm, we want to find the appropriate b, that minimizes

$$\|\mathbf{Ab} - \mathbf{c}\|^2$$

where $\mathbf{A}$ is a matrix that contains information from training data, and $\mathbf{c}$ is the corresponding given results. $\|\mathbf{Ab} - \mathbf{c}\|^2$ can be written as:

$$(\mathbf{Ab} - \mathbf{c})^T(\mathbf{Ab} - \mathbf{c}) = \mathbf{b}^T\mathbf{A}^T\mathbf{Ab} - 2\mathbf{b}^T\mathbf{A}^T\mathbf{c} + \mathbf{c}^T\mathbf{c}$$

Taking the derivative with respect to $\mathbf{b}$ and setting it to 0 gives

$$2\mathbf{A}^T\mathbf{A}\mathbf{b} - 2\mathbf{A}^T\mathbf{c} = 0$$

Therefore, in order to minimize $\mathbf{b}$, the corresponding $\mathbf{b}$ should be

$$(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{c}$$

To prevent overfit, we use regularization. Therefore, $\mathbf{b}$ is expected to be relatively small. So, the loss function is changed to:

$$\min\|\mathbf{A}\mathbf{b} - \mathbf{c}\|^2 + \lambda\|\mathbf{b}\|^2$$

Again, taking the derivative with respect to $\mathbf{b}$ and setting it to 0 gives $\mathbf{b}$ as

$$(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{c}$$

3.2 Restricted Boltzmann Machine

RBM is a generative stochastic artificial neural network with visible and hidden layers. Inference involves passing values between layers. Training involves calculating positive and negative gradients for gradient descent. Additional techniques such as momentum, biases, regularization, mini-batch, early stopping, and adaptive learning rates are employed for performance improvement. RBM is made of two layers which are the visible layers and the hidden layers. RBM is a bipartite graph where every node in the hidden layer is connected to every node in the visible layer. The weights of the connection are denoted as W, and the weight of the connection between the i-th node in the visible layer and the j-th node in the hidden layer is denoted as W_{ij}. We denote a vector v to represent the input (value of the visible layer) where v_i represents the i-th value of the visible layer. Here, v_i represents a vector of size 5. Similarly, we denote h_j to represent the j-th value of the hidden layer.

Regular RBM

The focus now shifts to the inference process of the Restricted Boltzmann Machine (RBM). Initially, the visible layer's values are transmitted to the hidden layer using Equation 1, where $\sigma(x)$ represents the sigmoid function.

$$P(h_j = 1|\mathbf{v}) = \sigma\left(\sum_{i \in V} v_i W_{ij}\right) \tag{1}$$

where

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Subsequently, the hidden layer's values are propagated back to the visible layer through Equation 2, incorporating the softmax function for normalization.

$$P(v_i^k = 1|\mathbf{h}) = \text{softmax}\left(\sum_{j \in J} h_j W_{ij}^k\right) \tag{2}$$

where

$$\text{softmax}(x_k) = \frac{e^{x_k - x_m}}{\sum_{l \in K} e^{x_k - x_m}} \quad x_m = \max_{l \in K} x_l$$

Here, the obtained $P(v_i = 1|h)$ from Equation 2 represents the likelihood of movie i being rated k points by the current user. The final rating of the movie is determined by selecting the highest value in vector $\mathbf{v_i}$ or averaging its elements, each multiplied by their respective scores (1, 2, 3, 4, or 5).
To train the model, gradient descent techniques are employed. Positive and negative gradients are calculated using Equations 3 and 4, respectively.

$$(PG)_{i,j,k} = P(h_j = 1|\mathbf{v}) \cdot v_i^k \tag{3}$$

$$(NG)_{i,j,k} = P(h_j = 1|\overline{\mathbf{v}}) \cdot \overline{v}_i^k \tag{4}$$

The weight matrix (W) is updated using the gradient descent formula in Equation 5.

$$W = W + \text{learningRate} \cdot (PG - NG) \tag{5}$$

Extended RBM

In the pursuit of heightened performance, a repertoire of sophisticated techniques is integrated into the Restricted Boltzmann Machine model. These enhancements include the strategic application of momentum, biases, regularization, mini-batch processing, early stopping, and adaptive learning rates.

Momentum The incorporation of momentum imparts a sense of inertia to the model, enabling it to navigate through gradients with a historical perspective. This is achieved by updating weights using a combination of the previous momentum and the current gradient, as expressed in Equation 6.

$$Momentum_t = \alpha Momentum_{t-1} + Gradient_t \tag{6}$$

Regularization To curtail the complexity of the weight matrix and foster model generalization, regularization is employed. The regularization process, governed by the L2 norm, penalizes intricate weight matrices, as denoted in Equation 7.

$$W = W - \alpha W \cdot \text{learningRate} \tag{7}$$

Biases Biases play a pivotal role in refining the model's expressiveness. The biases for each node in the visible and hidden layers are updated using Equations 8 and 9, respectively.

$$b_{vi} = \mathbf{v} - \mathbf{v}' \tag{8}$$

$$b_{hi} = \text{posHiddenProb} - \text{negHiddenProb} \tag{9}$$

The introduction of biases necessitates modifications to Equations 1 through 9 to account for these additional factors.

Early Stopping and Adaptive Learning Rate To mitigate overfitting, the model employs early stopping and adaptive learning rate mechanisms. The learning rate is dynamically adjusted based on the observed rate of RMSE change. Training is either slowed down or halted when the change in RMSE becomes marginal, promoting model stability.

Mini-Batch Processing Computation time is optimized through the implementation of mini-batch processing. Instead of processing the entire dataset at once, the model updates weights after processing a subset (mini-batch) of the data. This approach strikes a balance between efficiency and accurate model updates.

Cross-Validation While beyond the initial project scope, the pursuit of superior performance prompts the implementation of cross-validation. This technique, discussed in subsequent sections, involves partitioning the dataset into subsets for training and testing, enhancing the model's ability to generalize to unseen data and ensuring robust performance.

4 Implementation

4.1 Linear Regression

Within the scope of this project, matrix A is treated as a sparse representation of data, capturing the interactions between raters and movies. Simultaneously, vector $\mathbf{c}$ is derived from the ratings dataset. Each row of matrix A corresponds to a rating, where elements representing the associated rater and movie are set to 1. For example, in the first rating, the second person rated the third movie, resulting in the first row having columns corresponding to the second person and the third movie set to 1, and all others set to 0. Vector $\mathbf{c}$ is obtained by subtracting the mean value of all ratings (denoted as 'rBar' in the program) from the original ratings. Consequently, our predictions are based on the average rating. Moreover, to adhere to actual rating constraints, the final predictions are restricted to the range [1, 5]. Any predictions exceeding 5 are capped at 5.
In Python, the `param` function is defined as follows:

```
def param(A, c):
    b = np.linalg.inv(A.T @ A) @ A.T @ c
    return b
```

Similarly, the `param_reg` function is defined to incorporate regularization:

```
def param_reg(A, c, l):
    I = np.identity((A.T @ A).shape[0])
    b = np.linalg.inv(A.T @ A + l * I) @ A.T @ c
    return b
```

4.2 Restricted Boltzmann Machine

The foundational RBM model is implemented adhering to a standardized template, with additional enhancements such as adaptive learning rate, early stopping, and mini-batch processing. Adaptive learning rate dynamically adjusts the training rate or halts training based on observed RMSE trends. Mini-batch processing facilitates weight matrix updates after processing a subset of users, optimizing computation time.
The core implementation of the basic RBM model is straightforward and aligns with established templates, obviating the need for detailed discussion. However, the more intricate aspects of adaptive learning rate, early stopping, and mini-batch processing warrant thorough exploration.
To implement adaptive stopping, the learning rate is halved or training is halted when RMSE exhibits slow decrease. This involves comparing the

average of a certain number of recent RMSE values with the average of an extended set. If the difference falls below a predefined threshold, such as 0.001, the learning rate is halved or training is terminated.
For mini-batch processing, the weight matrix is updated only after processing a specified number of users (batch size). A counter is employed to track the number of users processed, and once it surpasses the batch size, gradient descent is performed, and the counter is reset.

5 Analysis

5.1 Linear Regression

Model Performance

The primary objective of implementing linear regression in the recommendation system was to predict ratings based on the relationship between various variables in the training data. The model's performance is evaluated based on its ability to accurately predict ratings for unseen data. The performance metrics used include Root Mean Squared Error (RMSE) for training set and RMSE for validation set.

Parameter Tuning

The linear regression model involves finding appropriate parameters, denoted as b, that minimize the loss function $\|Ab - c\|^2$. The analysis includes an exploration of the impact of different values of the regularization parameter (λ) on the model's performance. This exploration is very important for preventing overfitting and achieving a balance between accuracy on training data and generalization to unseen data.
The relationship between λ and RMSE in Figure 1 (next page).

The visual analysis of the graph indicates that the local optimal point for RMSE falls between $\lambda = 3$ and $\lambda = 4$.

Overfitting Prevention

To prevent overfitting, regularization is incorporated into the loss function. The term $\lambda\|b\|^2$ penalizes large values of b, encouraging the model to learn simpler and more generalizable patterns.

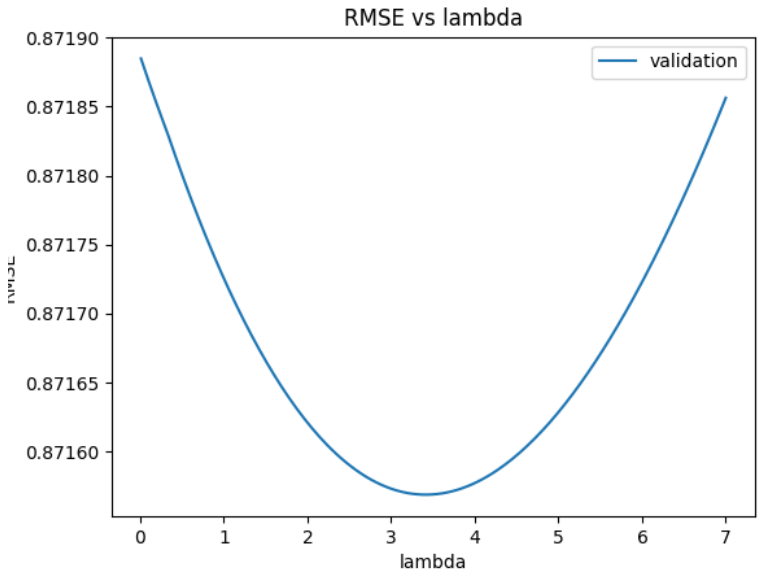

Figure 1: Impact of LR on RMSE

Interpretability of Results

Lastly, the results addresses the interpretability of the linear regression model. This involves examining the significance and contribution of individual features in predicting ratings. A more interpretable model is valuable in understanding the factors influencing the recommendation outcomes.

Conclusion

Overall, the analysis aims to provide a comprehensive understanding of the strengths, limitations, and practical implications of applying linear regression in the context of a recommendation system. Through this evaluation, insights are gained into the model's performance and its potential for real-world applications.

5.2 Restricted Boltzmann Machine

Analysis of Extended Parameters

The analysis of the RBM model's performance involves the examination of various extension parameters. Each parameter's impact on the model's performance is evaluated individually.

5.2.1 Batch Size

The influence of the batch size on training efficiency and model convergence is analyzed. The relationship between different batch sizes and the resulting RMSE is examined and visualized in Figure 2.

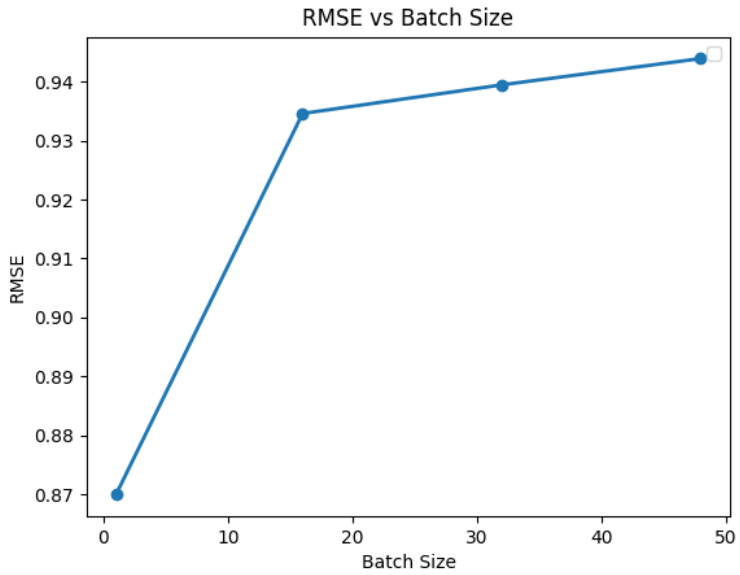

Figure 2: Impact of Batch Size on RMSE

Upon inspecting the results, a discernible pattern emerges. The RMSE consistently exhibits an increase as the batch size increasing. Notably, the growth rate is observed to be relatively large within the batch size range of 0 to 10, after which it experiences a gradual decrease.

Several factors may contribute to this observed behavior:

5.2.1.1 Noise in Larger Batches The augmentation of batch size may introduce increased noise into the training process. Larger batches might hinder the model's ability to generalize, as it may not adequately capture the intricate patterns present in the entire dataset.

5.2.1.2 Generalization Issues Smaller batch sizes allow the model to make more frequent updates based on smaller subsets of data, potentially aiding in better generalization. Larger batches may compromise this ability,

leading to poorer generalization.

5.2.2 Beta

The impact of the beta parameter on the model's momentum and convergence is assessed. A range of beta values is considered, and the corresponding effects on the model's performance are depicted in Figure 3 (next page).

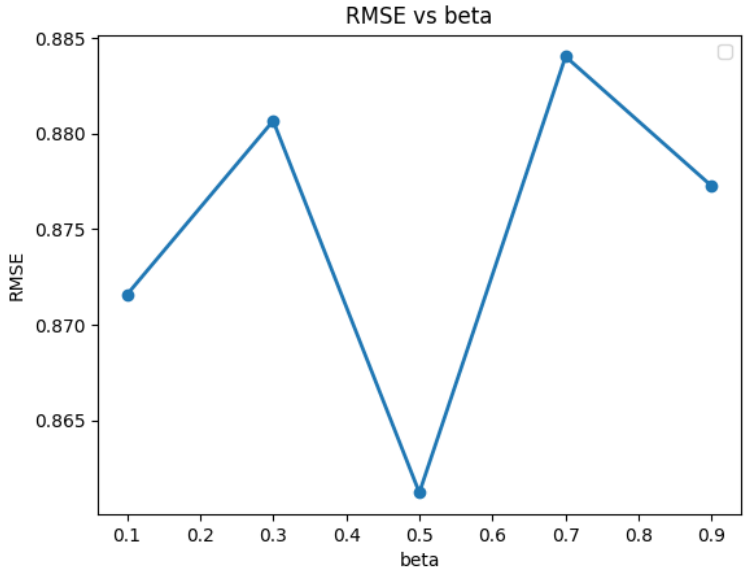

Figure 3: Effect of Beta on Model Momentum

The visual analysis of the graph indicates that the local optimal point for RMSE is achieved when beta is set to 0.5.

5.2.3 Decay Factor

The analysis explores the impact of the decay factor on the learning rate's adaptation over time. The relationship between decay factor values and the learning rate is illustrated in Figure 4.
The examination of the data reveals a notable pattern in the Root Mean Square Error values. A discernible decrease in RMSE is observed within the range of delay factor values from 0.84 to 0.93. Subsequently, there is an increase in RMSE between 0.93 and 0.96, followed by a pronounced sharp decline after 0.96.
Therefore, the following nuanced conclusions can be drawed:
the delay factor emerges as a pivotal determinant of predictive performance, with a optimal range for enhanced accuracy. However, careful consideration

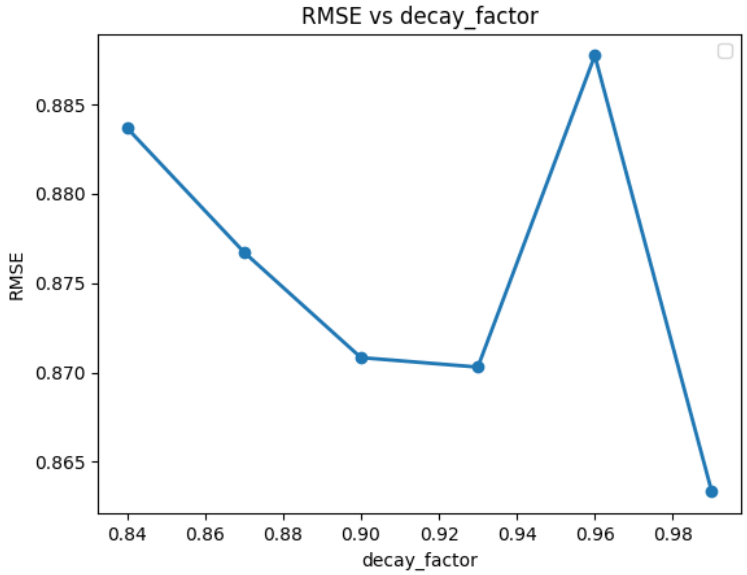

Figure 4: Impact of Decay Factor on Learning Rate

is needed when venturing beyond a delay factor of 0.93, as deviations may lead to a precipitous decline in model efficacy. Consequently, the judicious fine-tuning of the delay factor within the identified optimal range is recommended to optimize overall model performance.

5.2.4 F

F is the number of hidden units in the RBM model, which is a parameter of great significance to the results. The relationship between F values and the resulting RMSE is visualized in Figure 5.

RMSE Trends with Increasing F As the number of hidden units (F) increases from 1 to 36, a consistent rise in RMSE is observed. This initial increase is indicative of the model's struggle to effectively capture relevant patterns with higher complexity.

Local Minimum at $F = 44$ Surprisingly, the RMSE experiences a local minimum at $F = 44$. This suggests that, within our experimental conditions, an optimal balance is struck between model complexity and the ability to generalize, resulting in improved predictive accuracy.

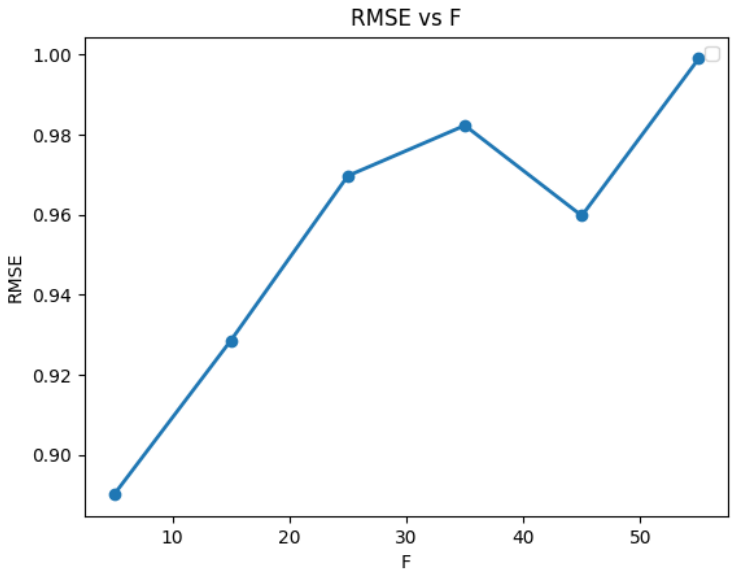

Figure 5: Effect of F on RMSE

Subsequent Increase in RMSE Beyond $F = 44$, the RMSE begins to increase once again. This phenomenon indicates that further increasing the number of hidden units may lead to overfitting, causing a decline in the model's ability to generalize to unseen data.

Possible Explainations The observed rise in RMSE with an increasing number of hidden units initially reflects the model's struggle with increased complexity. The local minimum at $F = 44$ highlights a critical point where the model achieves a balance between capturing intricate patterns and maintaining generalizability.
The subsequent increase in RMSE beyond this optimal point underscores the importance of avoiding excessive model complexity, as it can lead to overfitting and diminished performance on new data.

5.2.5 Learning Rate

The analysis explores the impact of different learning rates on model convergence and training speed. The relationship between learning rate values and RMSE is presented in Figure 6.
The optimal value for this model is a learning rate of 0.03.

5.2.6 Interaction between F and Learning Rate

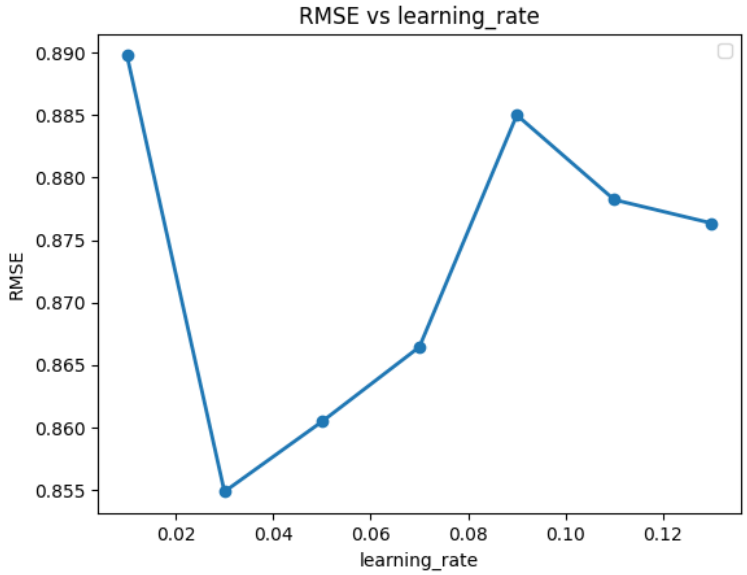

Figure 6: Impact of Learning Rate on Model Convergence

In addition to the individual analysis of extension parameters, we conducted a comprehensive investigation into the joint effects of F and learning rate. Our findings suggest that within a defined range ($F = 0$ to 50), an optimal learning rate can be determined to achieve favorable model fitting. For instance, a learning rate of 0.01 is recommended when F is set to 5, and 0.02 is preferable when F is set to 25.

These analyses collectively provide insights into the sensitivity of the RBM model to each extension parameter, aiding in the selection of optimal values for improved performance.

6 Conclusion

In conclusion, our investigation into the Netflix recommendation system encompassed the application and analysis of Linear Regression, Restricted Boltzmann Machine (RBM), and extensions of RBM.
Linear Regression effectively predicted user ratings, contributing to the recommendation system's foundation.
RBM demonstrated its potential as a generative stochastic neural network, with insights gained from parameter analyses.
Exploring extensions, including batch size, beta, decay factor, F and learning

rates, provided a nuanced understanding of the model's adaptability. Further, the combined effects of F and learning rate were explored, revealing optimal configurations for improved fitting. The integration of these models enhances the recommendation system's overall efficacy, laying the groundwork for informed design decisions and potential future refinements to boost predictive accuracy and user satisfaction.