Movie Recommendation System

Project By
Tejas Shinde

Problem Description

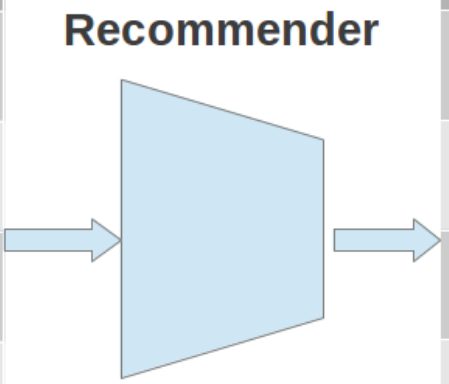
- Group of users
- Set of movies
- Every user has watched some movies but not all
- If users watch a movie, they rate it as well

Problem Statement

- Given a user how to recommend a new movie using the ratings?

Problem Formulation

1	0	3	0	0
0	0	0	5	0
1	2	0	0	4
4	0	3	3	0
5	5	2	1	0



1.1	2.1	3.1	4.3	3.9
4.5	3	3.1	4.9	4.1
1	1.9	2.3	3.2	4.3
3.8	4.7	3.1	2.8	3.9
4.9	5	1.9	1.2	3.7

Matrix M with N_u rows and N_m columns

Possible Approaches

- Collaborative Filtering
  - K-Nearest Neighbors
  - Using cosine similarity metric

$$sim(m_1,m_2) = \frac{< r_{m1} - r_{mean},\ r_{m2} - r_{mean} >}{|r_{m1} - r_{mean}|_2 \times |r_{m2} - r_{mean}|_2}$$

- **Low Rank Approximation**
  - **$R = U V^T$** for some lower dimensional feature space S_F.

Implementation

- 'n' users, 'm' movies
- discover the latent 'k' features
- Define squared error,
  - $e_{ij}^2 = (r_{ij} - p_{ij})^2$,
  - $p_{ij} = sum_k(u_{ik}, v_{kj})$
- Objective: find 'k', minimizing 'e'
- Can achieve using **Gradient Descent**

Implementation

- Obtain gradient by differentiating 'e' w.r.t. 'u' and 'v'
- Update rule,
  - $u_n = u_{n-1} + 2 * a * e_{n-1} * v_{n-1}$
  - $v_n = v_{n-1} + 2 * a * e_{n-1} * u_{n-1}$
- 'a' is the learning rate

Implementation

- **Regularization**
- Modify squared error 'e' as,
  - $e_{ij}^2 = (r_{ij} - p_{ij})^2 + b * sum_k(|U|_F + |V|_F)$
  - 'b' is regularizer
- Regularized update rule,
  - $u_n = u_{n-1} + 2 * a * (e_{n-1} * v_{n-1} - b * u_{n-1})$
  - $v_n = v_{n-1} + 2 * a * (e_{n-1} * u_{n-1} - b * v_{n-1})$

Pseudo Code

- Initialize U and V randomly
- Until convergence
- For i ={1, 2 … n}
  - For j={1, 2, … m}
    - $E_{ij} = R_{ij} – sum_k(U_{ik}V_{kj}) + b * |U|_F * |V|_F$
    - If (error < threshold) **return**
    - $u_{ik} = u_{i,k-1} + 2 * a * (e_{ij} * v_{k-1,j} - b * u_{i,k-1})$
    - $v_{kj} = v_{k-1,j} + 2 * a * (e_{ij} * u_{i,k-1} - b * v_{k-1,j})$

Post-processing

- Bias removal:
  - Correct prediction by adding
  - Correction = $\text{mean}_{\text{training}} - \text{mean}_{\text{prediction}}$
- Prediction Truncation:
  - (Prediction < 1) = 1
  - (Prediction > 5) = 5
- Round-off prediction to nearest integer if difference < 0.1

Results

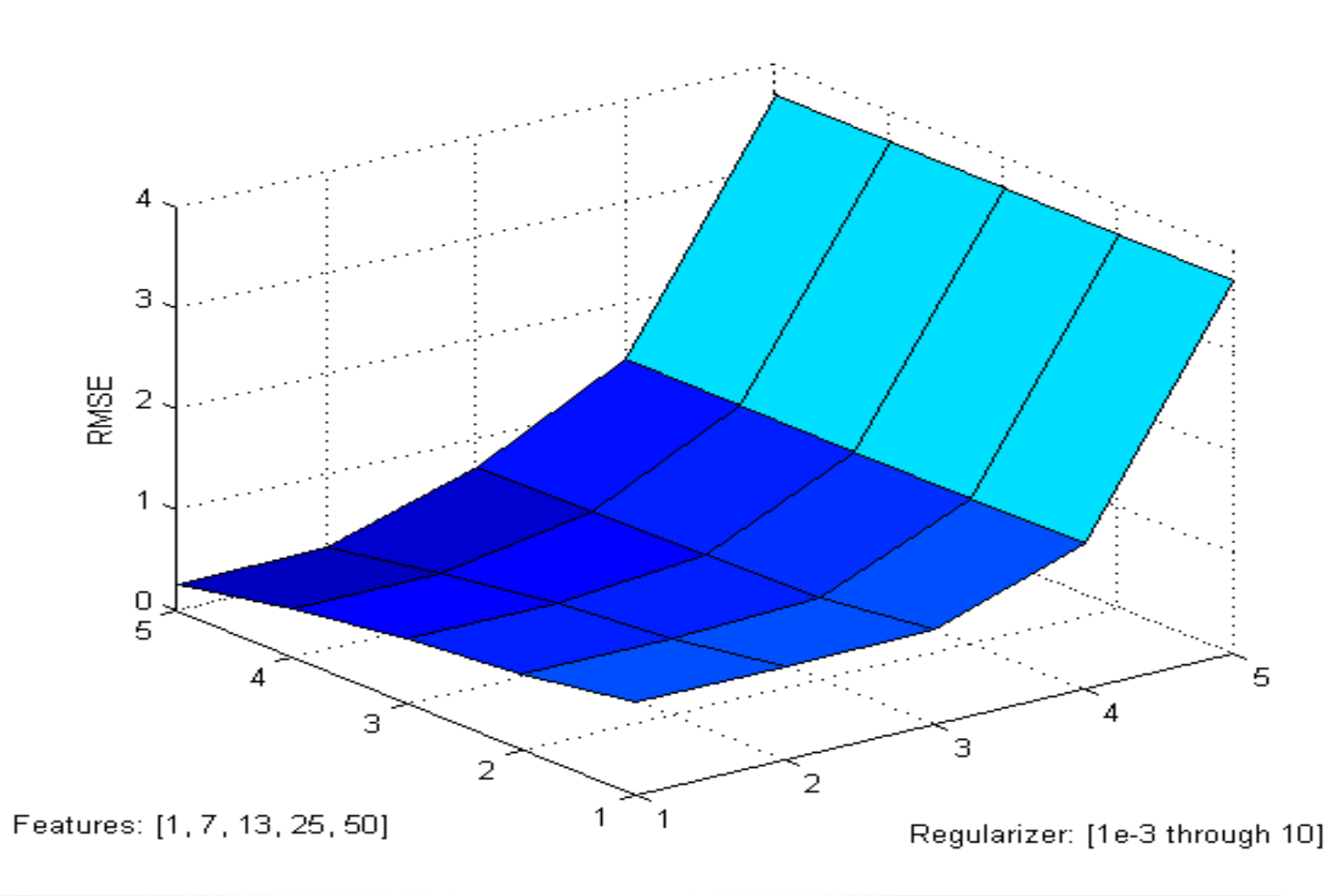


Fig: Train Error Surface

Results

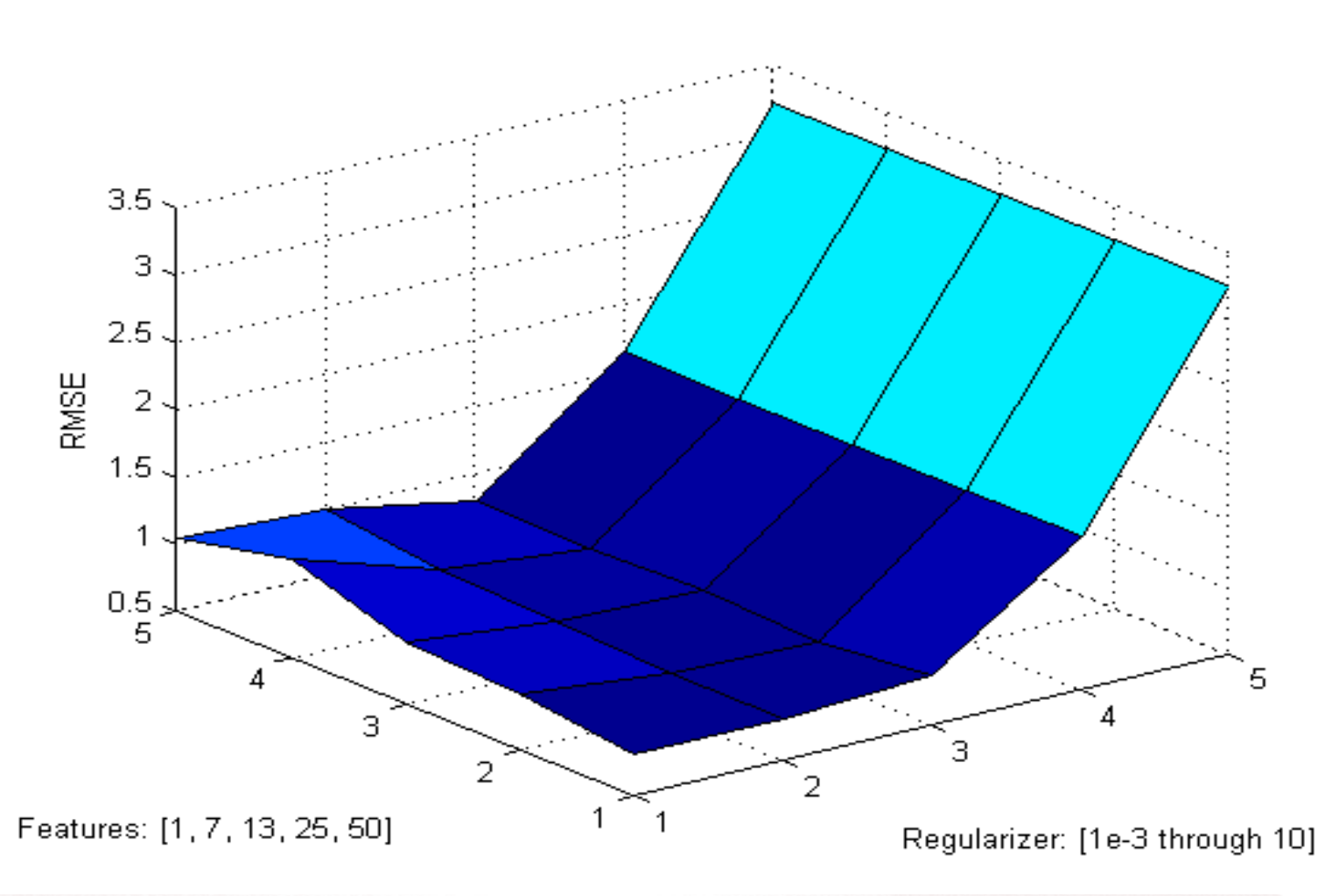
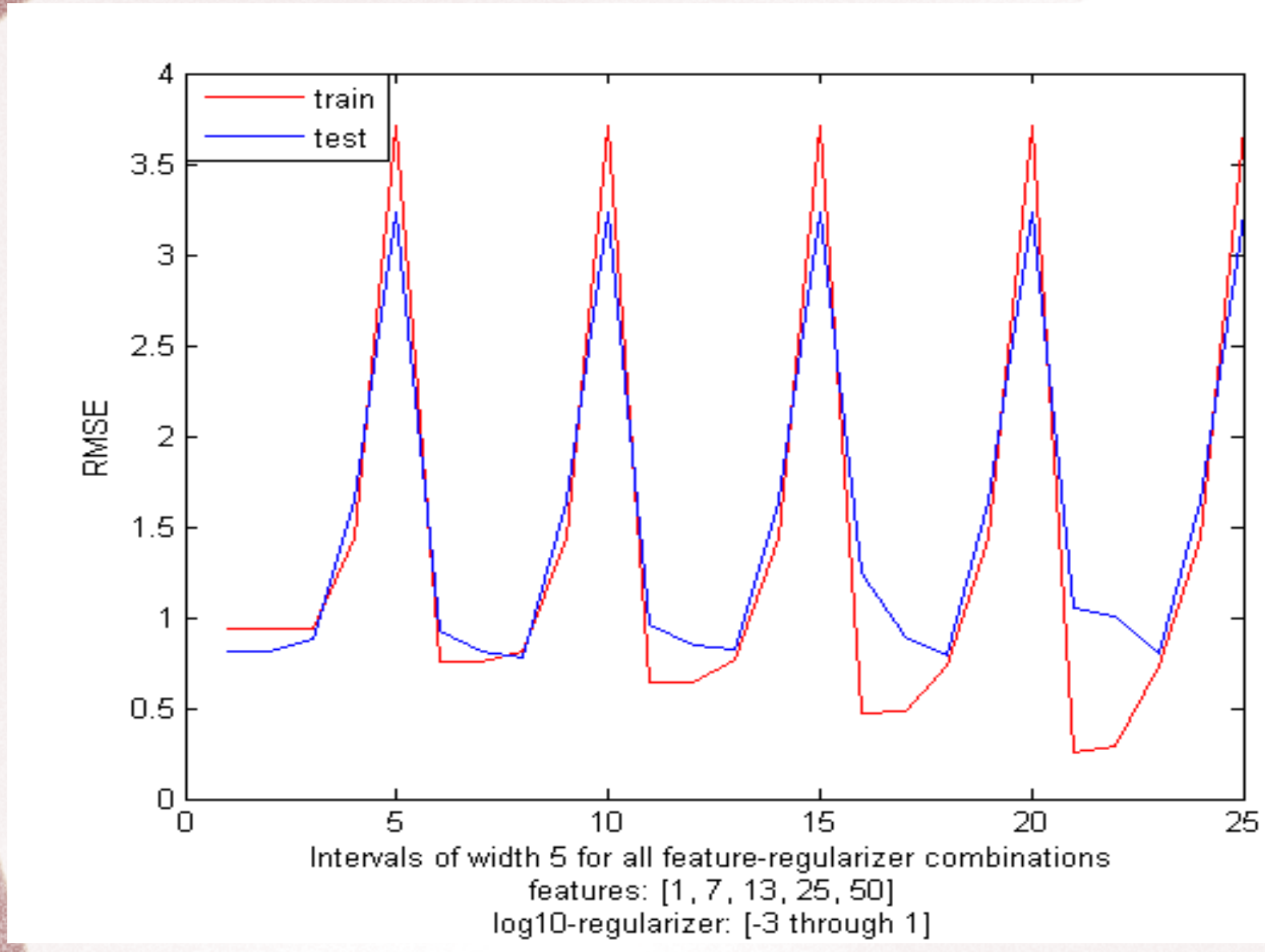
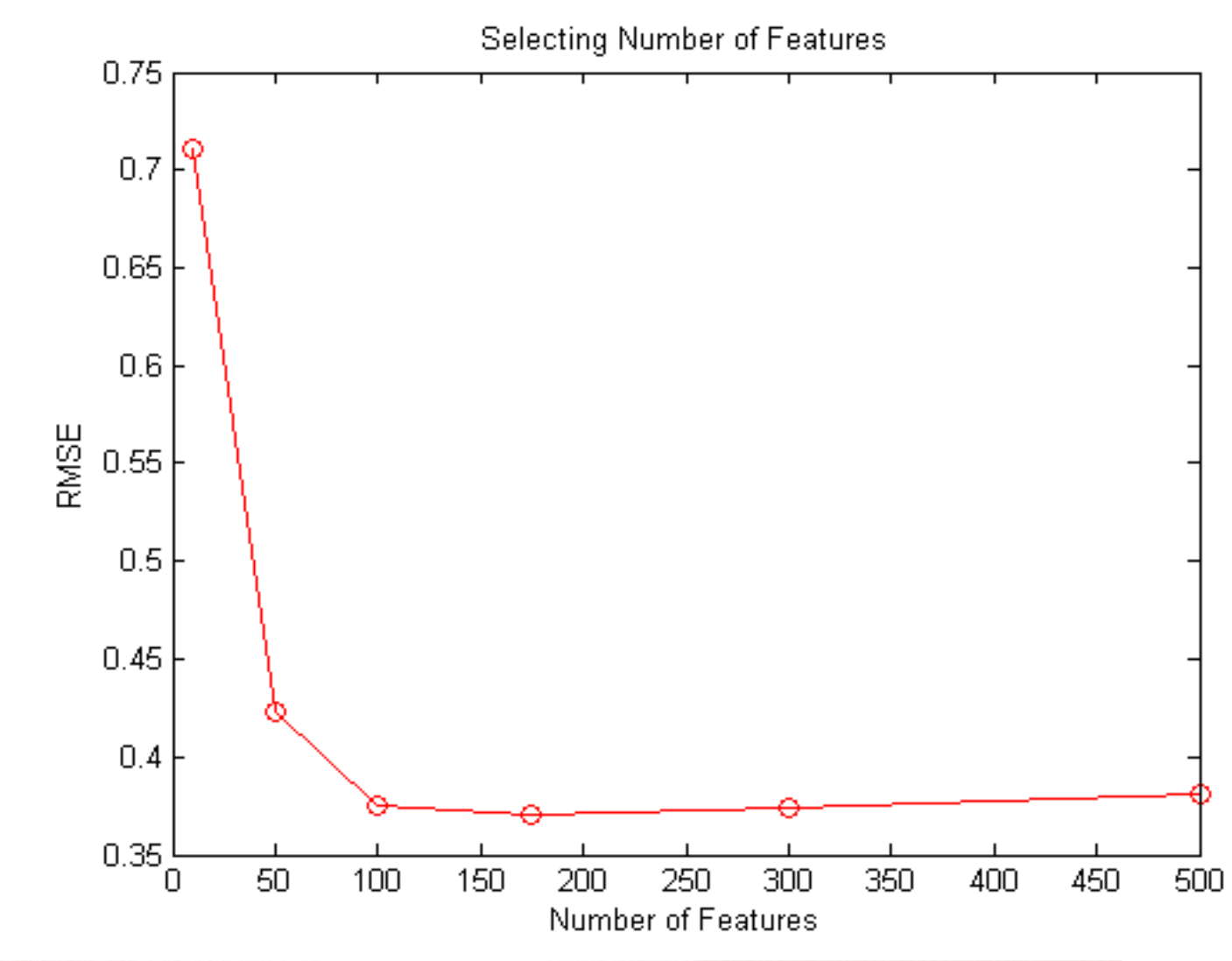


Fig: Test Error Surface

Results


train
test
RMSE
0
0.5
1
1.5
2
2.5
3
3.5
4
0
5
10
15
20
25
Intervals of width 5 for all feature-regularizer combinations
features: [1, 7, 13, 25, 50]
log10-regularizer: [-3 through 1]

Results

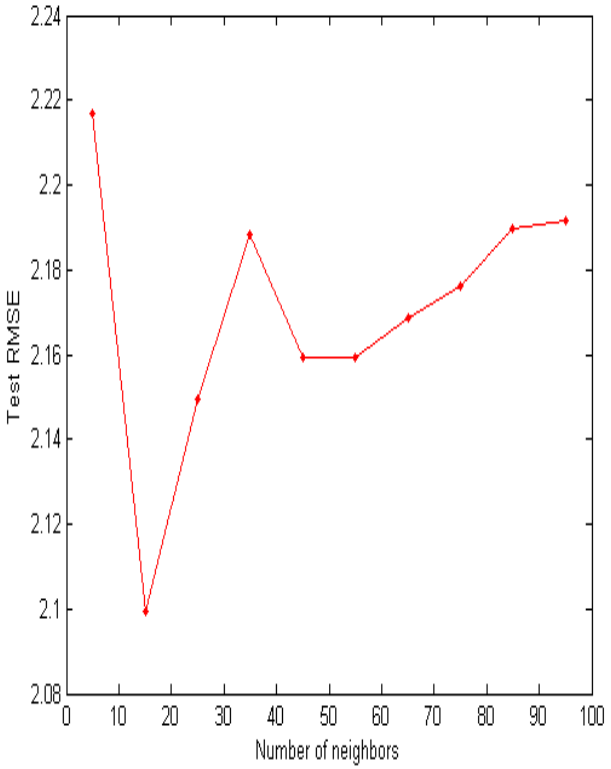


- Features : 175
- Regularizer : 0.04

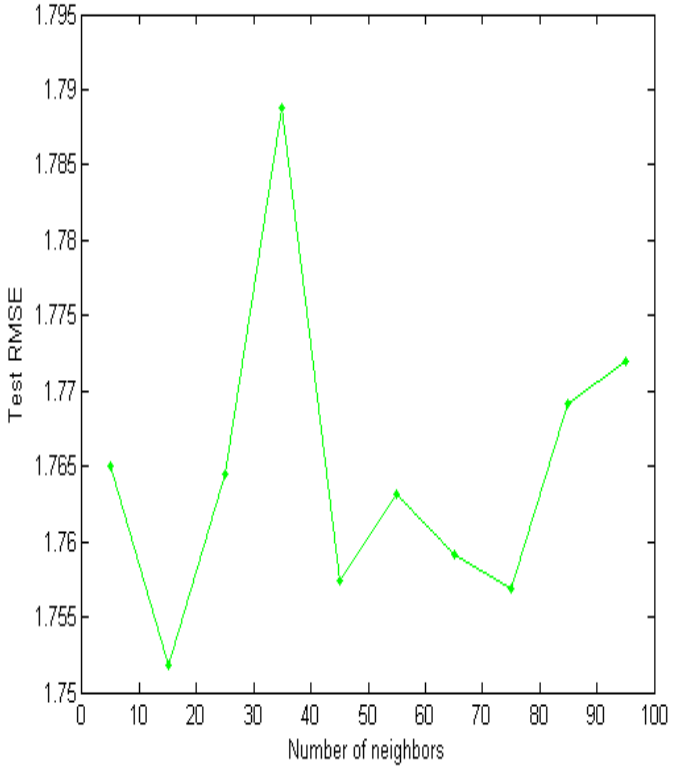
Experiments

- Algorithm mixture:
  - Compute prediction as convex combination of two algorithms
  - $P = \lambda * P_1 + (1-\lambda) * P_2$
  - P_1 predicted by 1st algorithm : ALS
  - P_2 predicted by 2nd algorithm : K-NN

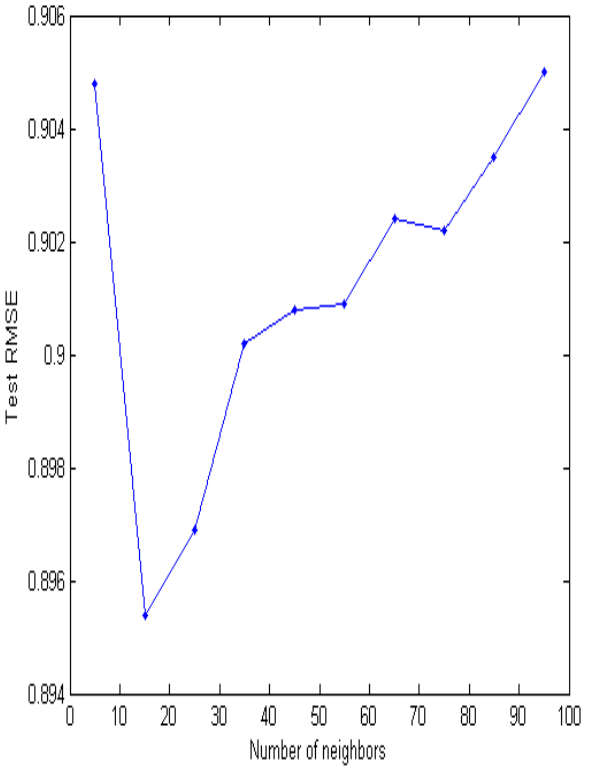
Results



λ = 0.05



λ = 0.5



λ = 0.95

References

- “Large-Scale Parallel Collaborative Filtering for the Netflix Prize”, Y. Zhou et.al.
- “Recommendation System Based on Collaborative Filtering”, Zheng Wen