Building A Predictive Model

An example of a product recommendation engine

Alex Lin
Senior Architect
Intelligent Mining
alin@intelligentmining.com

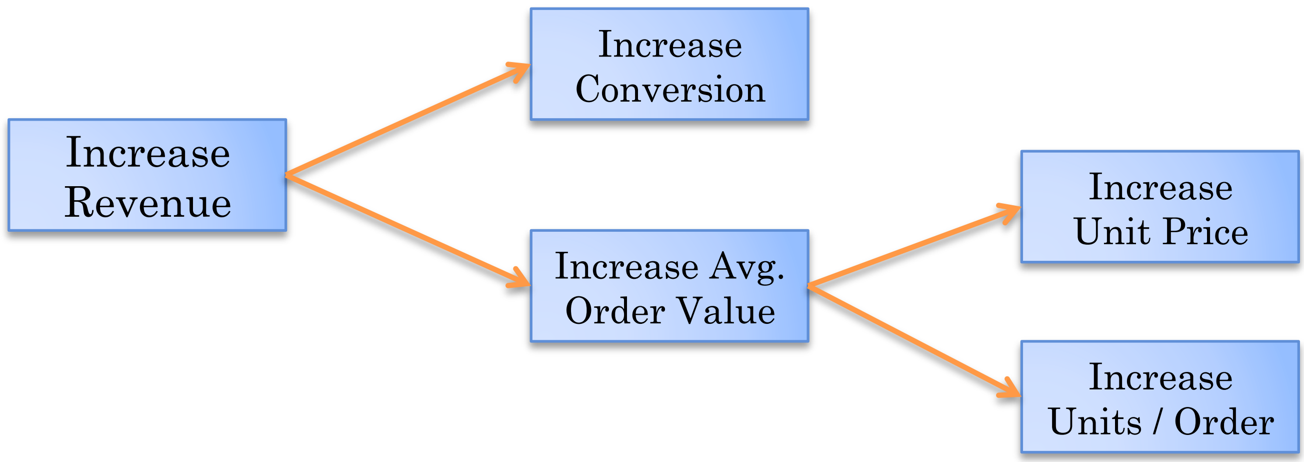
Outline

- Predictive modeling methodology
- k-Nearest Neighbor (kNN) algorithm
- Singular value decomposition (SVD) method for dimensionality reduction
- Using a synthetic data set to test and improve your model
- Experiment and results

The Business Problem

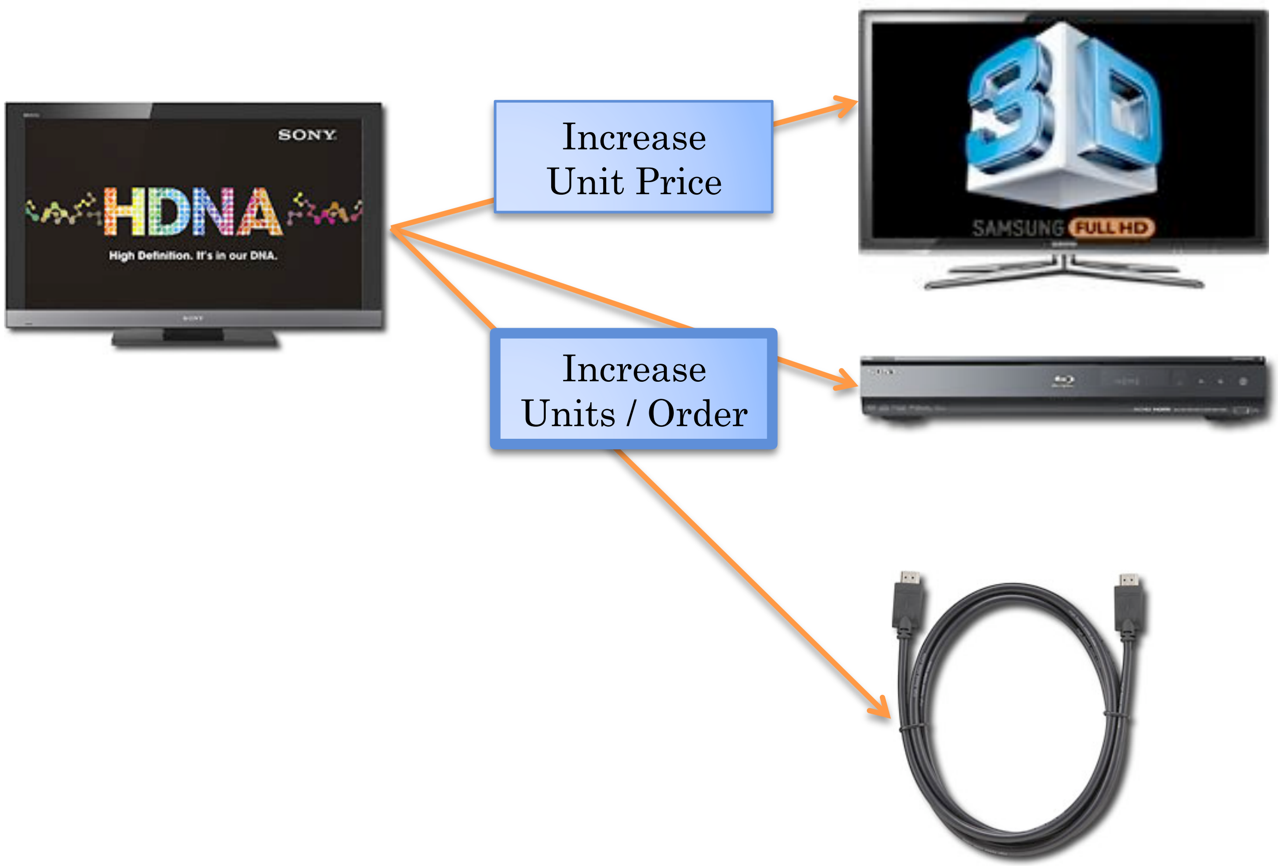
- Design product recommender solution that will increase revenue.

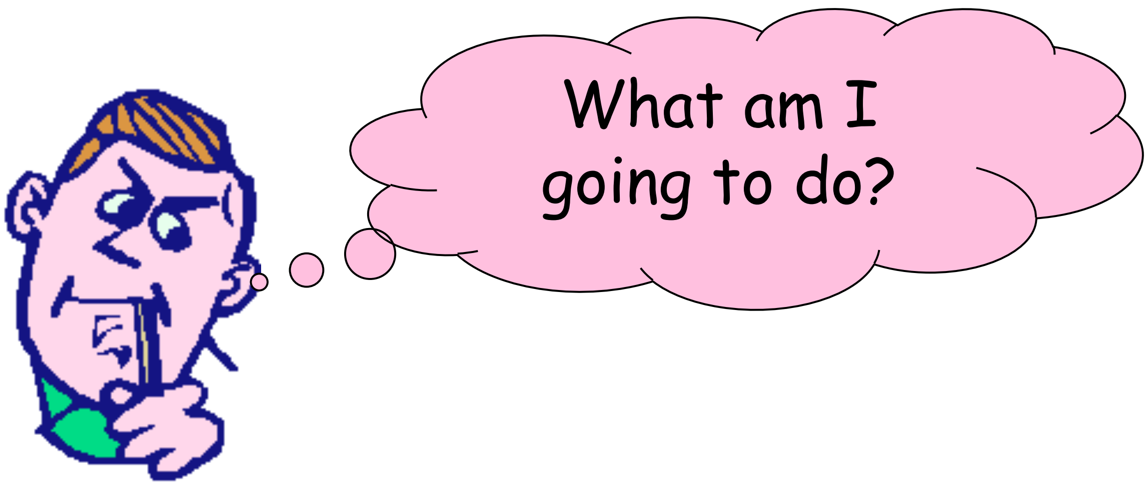
How Do We Increase Revenue?



Example

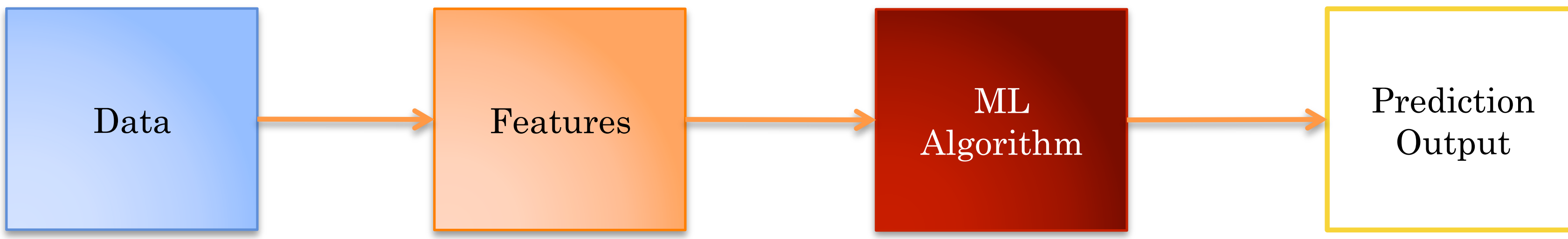
- Is this recommendation effective?




What am I
going to do?

Predictive Model

- Framework



What data?

What feature?

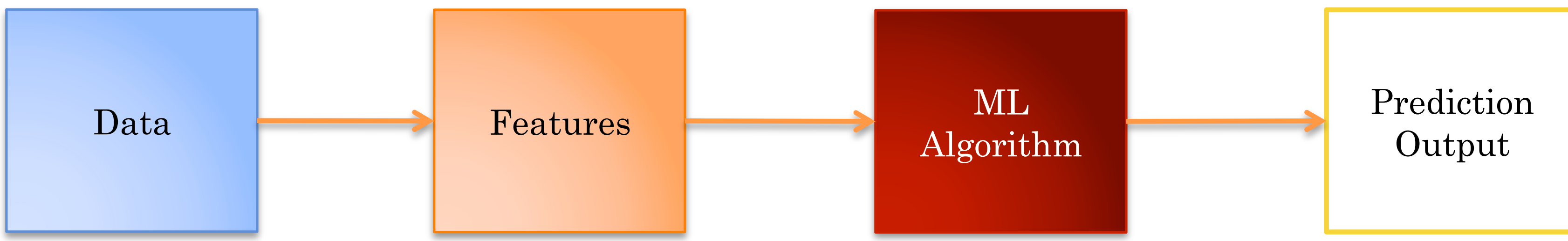
Which Algorithm ?

Cross-sell & Up-sell Recommendation

What Data to Use?

- Explicit data
  - Ratings
  - Comments
- Implicit data
  - Order history / Return history
  - Cart events
  - Page views
  - Click-thru
  - Search log
- In today's talk we only use Order history and Cart events

Predictive Model



Order History
Cart Events

What feature?

Which Algorithm ?

Cross-sell & Up-sell
Recommendation

What Features to Use?

- We know that a given product tends to get purchased by customers with similar tastes or needs.
- Use user engagement data to describe a product.

users

item	1	2	3	4	5	6	7	8	9	10	…	n
17	1		.25			.25		1		.25		

user engagement vector

Data Representation / Features

- When we merge every item's user engagement vector, we got a m x n item-user matrix

users

items

	1	2	3	4	5	6	7	8	9	10	…	n
1	1		.25			1				.25		
2							.25					
3	1			.25				1				
4		.25			1			.25	1			
⋮			1				1					
m												

Data Normalization

- Ensure the magnitudes of the entries in the dataset matrix are appropriate

users

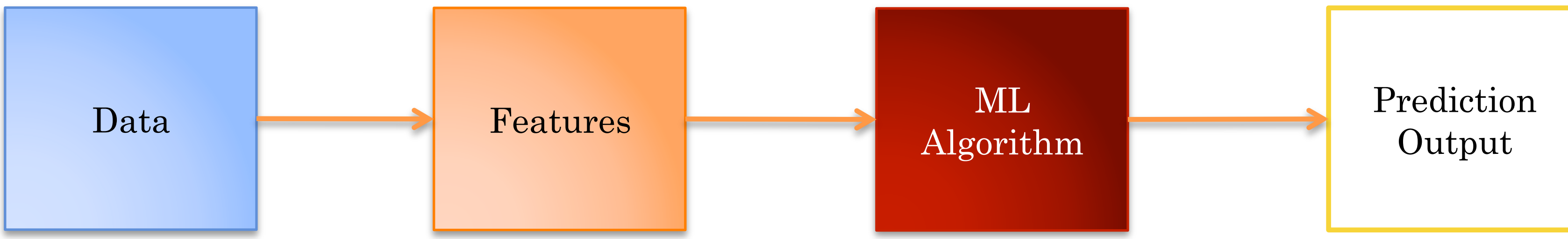
items	1	2	3	4	5	6	7	8	9	10	…	n
1	.5		.9			.92				.49		
2							.79					
3	.67			.46				.73				
4		.39			.82			.76	.69			
⋮			.52				.8					
m												

- Remove column average – so frequent buyers don't dominate the model

Data Normalization

- Different engagement data points (Order / Cart / Page View) should have different weights
- Common normalization strategies:
  - Remove column average
  - Remove row average
  - Remove global mean
  - Z-score
  - Fill-in the null values

Predictive Model



Order History
Cart Events

User engagement vector

users

item	1	2	3	4	5	6	7	8	9	10	...	n
17	1		.5			1		.5		1		

Data Normalization

Which Algorithm ?

Cross-sell & Up-sell Recommendation

Which Algorithm?

- How do we find the items that have similar user engagement data?

users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	1		.25			1				1		
2							1					
17	1			1		1		.25		.25		
18		1			.25	1		1	1			
⋮			.25				1					
m												

- We can find the items that have similar user engagement vectors with kNN algorithm

k-Nearest Neighbor (kNN)

- Find the k items that have the most similar user engagement vectors

users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	.5		1			1				1		
2		1					.5			1		
3	1			1				1	1			
4		1			.5		1		1			
⋮			.5				1					
m				1					.5			

- Nearest Neighbors of Item 4 = [2,3,1]

Similarity Measure for kNN

users

items	1	2	3	4	5	6	7	8	9	10	…	n
2		1					.5			1		
4		1			.5		1		1			

- Jaccard coefficient:

$$sim(a,b) = \frac{(1+1)}{(1+1+1)+(1+1+1+1)-(1+1)}$$

- Cosine similarity:

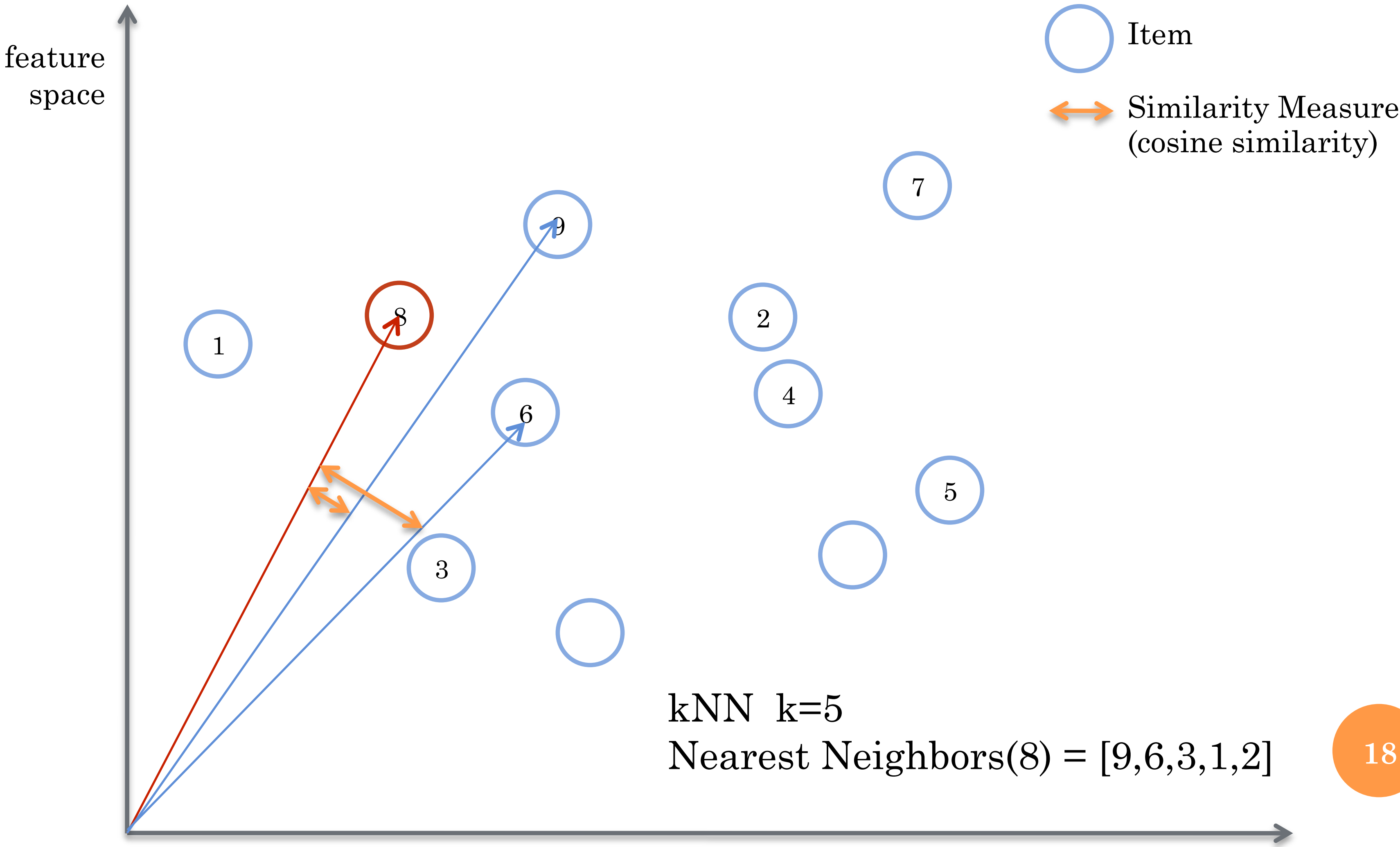
$$sim(a,b) = \cos(\vec{a},\vec{b}) = \frac{\vec{a} \bullet \vec{b}}{\left\|\vec{a}\right\|_2 * \left\|\vec{b}\right\|_2} = \frac{(1*1+0.5*1)}{\sqrt{(1^2+0.5^2+1^2)*(1^2+0.5^2+1^2+1^2)}}$$

- Pearson Correlation:

$$corr(a,b) = \frac{\sum_i (r_{ai}-\overline{r_a})(r_{bi}-\overline{r_b})}{\sqrt{\sum_i (r_{ai}-\overline{r_a})^2 \sum_i (r_{bi}-\overline{r_b})^2}} = \frac{m\sum a_i b_i - \sum a_i \sum b_i}{\sqrt{m\sum a_i^2 - (\sum a_i)^2}\sqrt{m\sum b_i^2 - (\sum b_i)^2}}$$

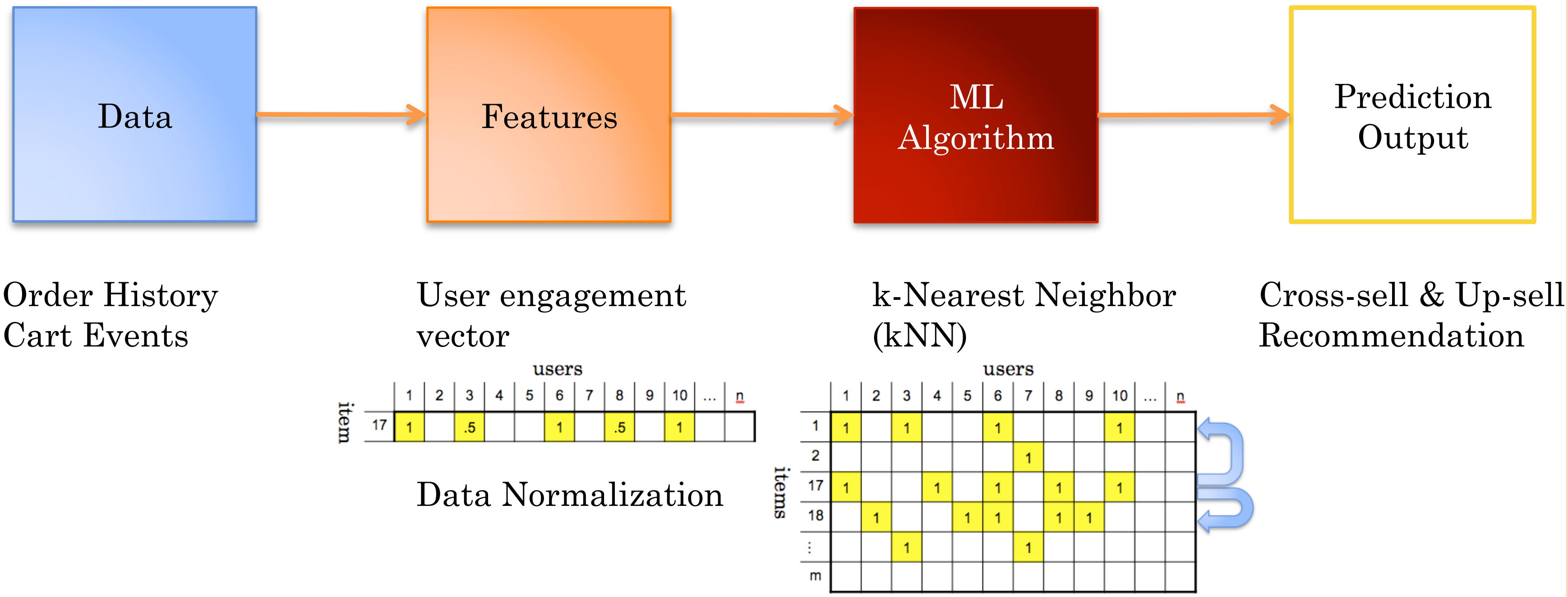
$$= \frac{match\_cols * Dotprod(a,b) - sum(a) * sum(b)}{\sqrt{match\_cols * sum(a^2) - (sum(a))^2}\sqrt{match\_cols * sum(b^2) - (sum(b))^2}}$$

k-Nearest Neighbor (kNN)



Predictive Model

- Ver. 1: kNN



Cosine Similarity – Code fragment

```
long i_cnt = 100000; // number of items 100K
long u_cnt = 2000000; // number of users 2M
double data[i_cnt][u_cnt]; // 100K by 2M dataset matrix (in reality, it needs to be malloc allocation)
double norm[i_cnt];

// assume data matrix is loaded
……
// calculate vector norm for each user engagement vector
for (i=0; i<i_cnt; i++) {
    norm[i] = 0;
    for (f=0; f<u_cnt; f++) {
        norm[i] += data[i][f] * data
    }
    norm[i] = sqrt(norm[i]);
}

// cosine similarity calculation
for (i=0; i<i_cnt; i++) {    // loop
   for (j=0; j<i_cnt; j++) {  // loop
        dot_product = 0;
        for (f=0; f<u_cnt; f++) { // loop thru entire user space 2M
            dot_product += data[i][f] * data[j][f];
        }
        printf("%d %d %lf\n", i, j, dot_product/(norm[i] * norm[j]));
    }

// find the Top K nearest neighbors here
…….
```

1. 100K rows x 100K rows x 2M features --> scalability problem

   kd-tree, Locality sensitive hashing,

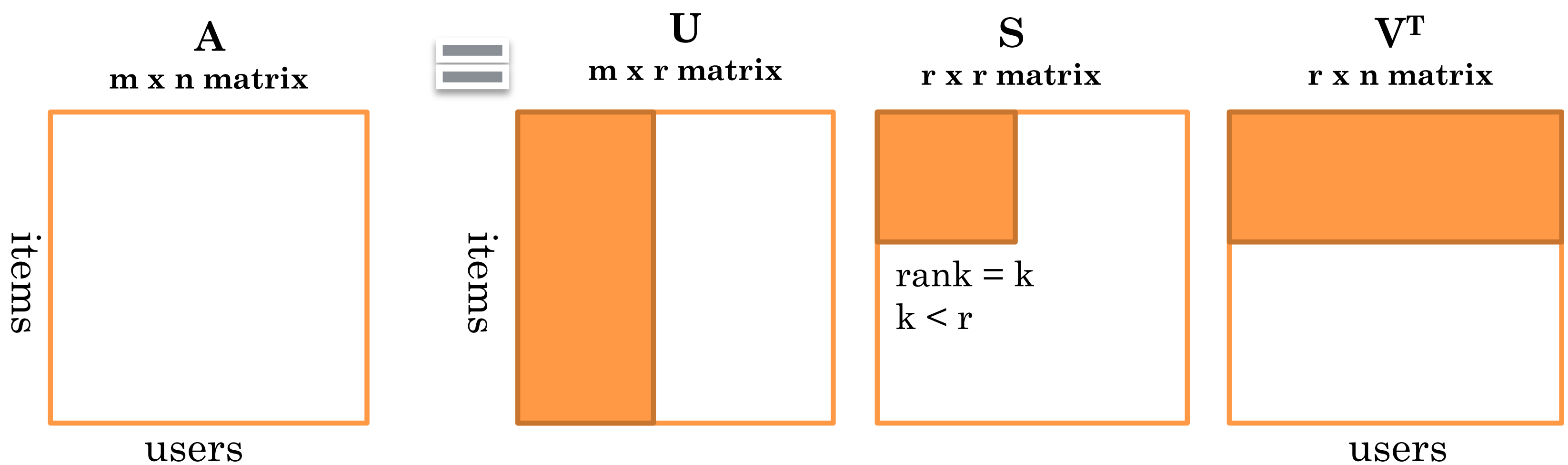
   MapReduce/Hadoop, Multicore/Threading, Stream Processors

2. data[i] is high-dimensional and sparse, similarity measures are not reliable --> accuracy problem

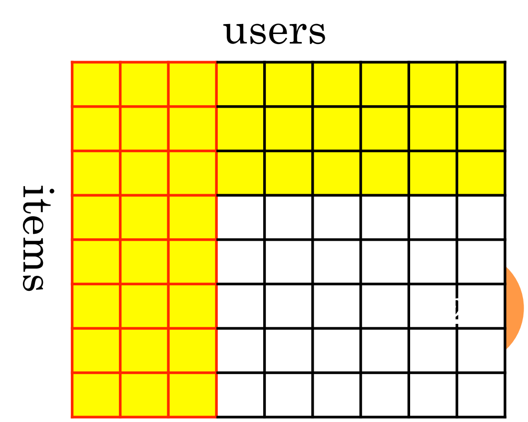
   This leads us to The SVD dimensionality reduction !

Singular Value Decomposition (SVD)

$$A = U \times S \times V^T$$



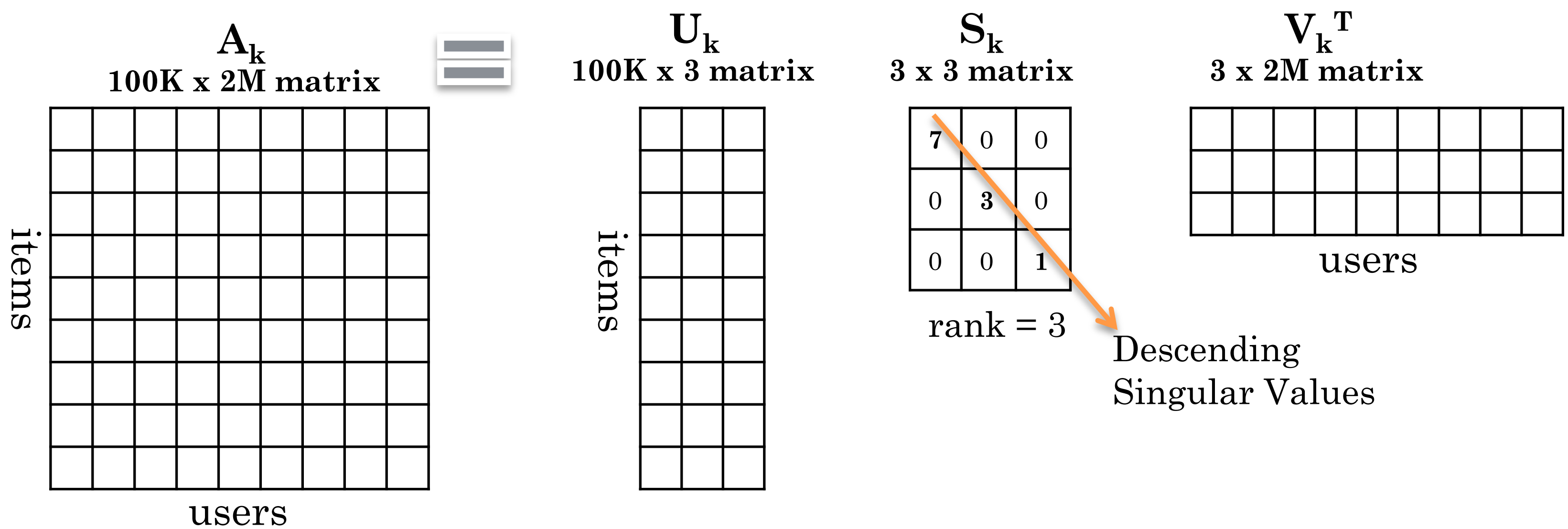
$$A_k = U_k \times S_k \times V_k^T$$



- Low rank approx. Item profile is $U_k * \sqrt{S_k}$
- Low rank approx. User profile is $\sqrt{S_k} * V_k^T$
- Low rank approx. Item-User matrix is $U_k * \sqrt{S_k} * \sqrt{S_k} * V_k^T$

Reduced SVD

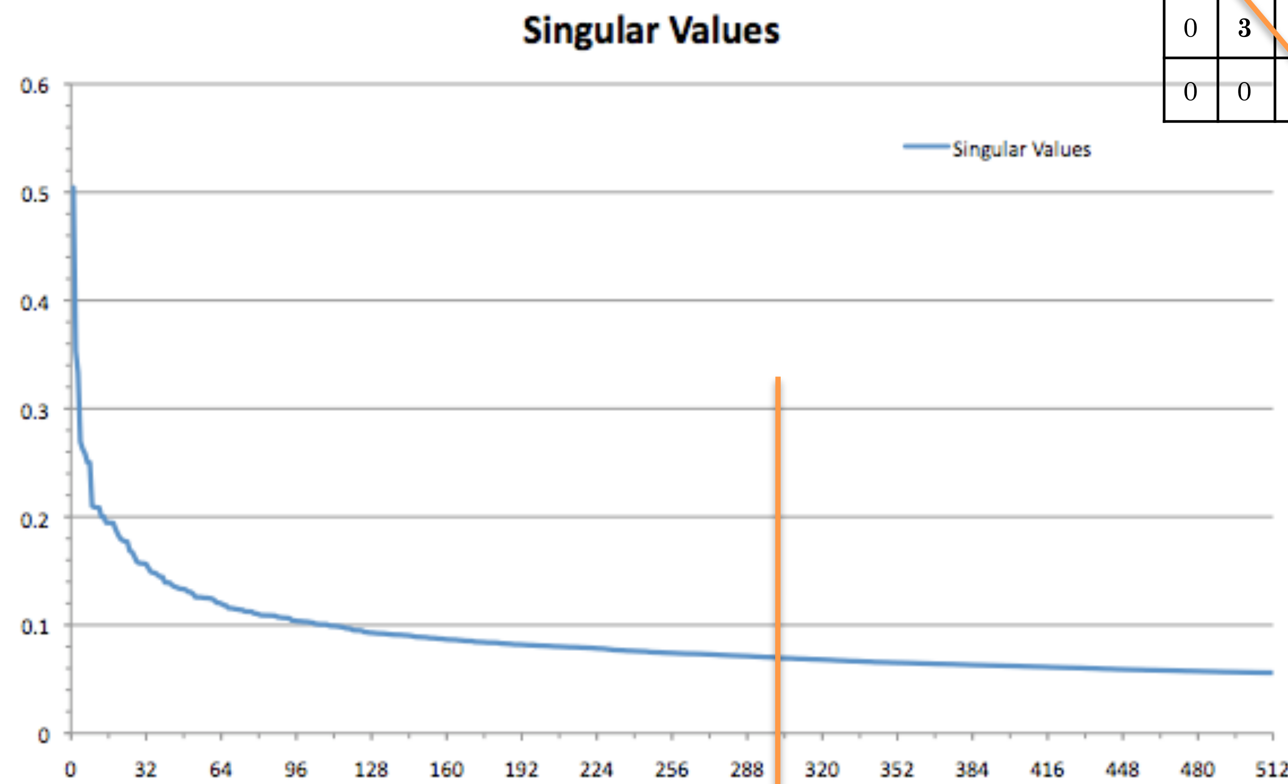
$$A_k = U_k \times S_k \times V_k^T$$



- Low rank approx. Item profile is $U_k * \sqrt{S_k}$

SVD Factor Interpretation

- Singular values plot (rank=512)



S

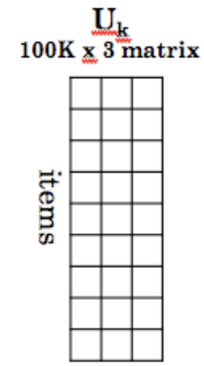
3 x 3 matrix

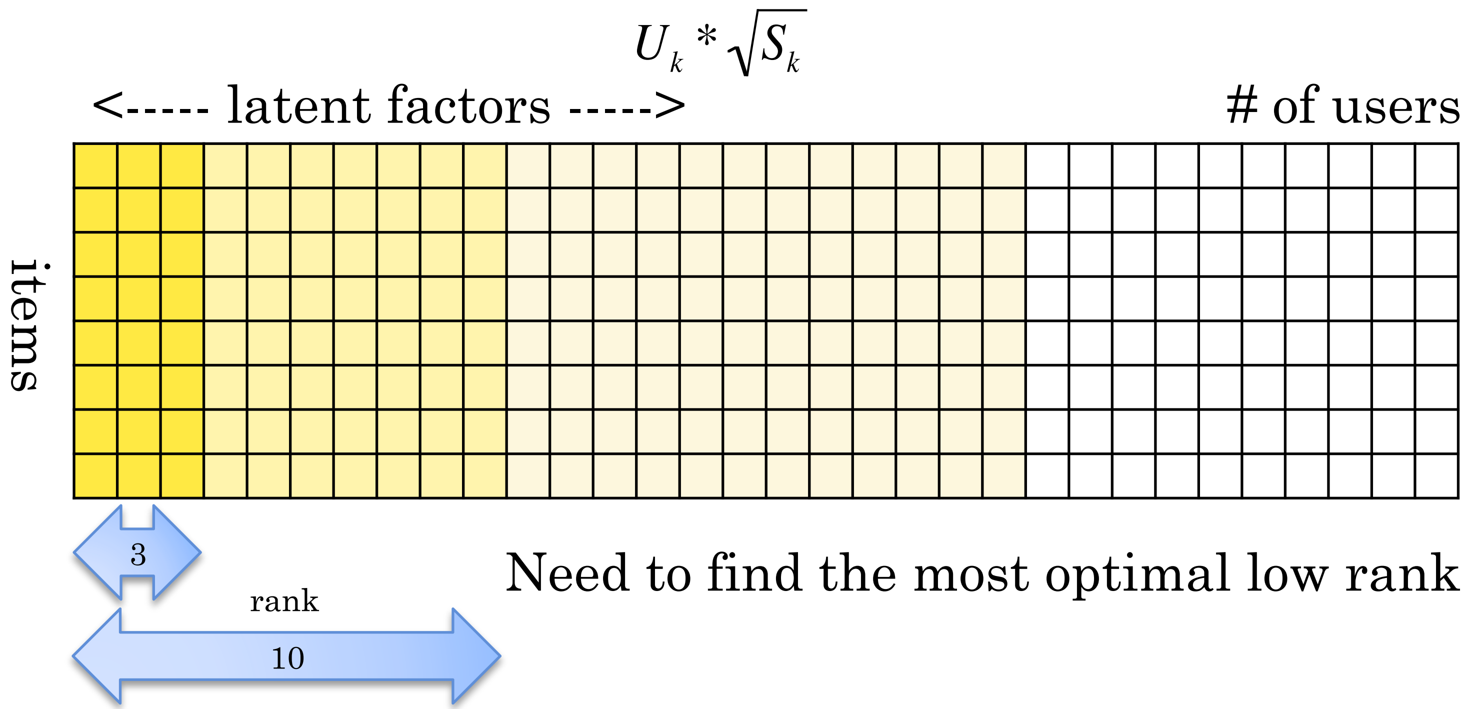
7	0	0
0	**3**	0
0	0	**1**

Descending Singular Values

More Significant — Latent Factors | Noises + Others — Less Significant

SVD Dimensionality Reduction





Need to find the most optimal low rank !!

Missing values

users

item	1	2	3	4	5	6	7	8	9	10	…	n
17	1		.5			1		.5		1		

- Difference between “0” and “unknown”
- Missing values do NOT appear randomly.
- Value = (Preference Factors) + (Availability) – (Purchased elsewhere) – (Navigation inefficiency) – etc.
- Approx. Value = (Preference Factors) +/- (Noise)
- Modeling missing values correctly will help us make good recommendations, especially when working with an extremely sparse data set

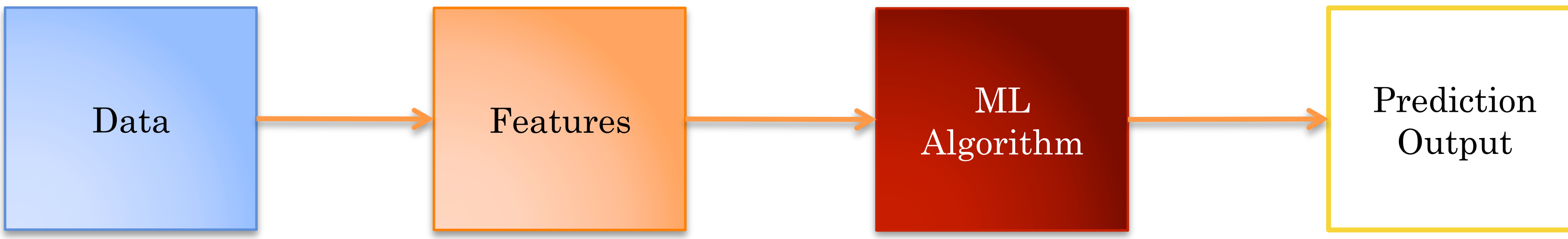
Singular Value Decomposition (SVD)

- Use SVD to reduce dimensionality, so neighborhood formation happens in reduced user space
- SVD helps model to find the low rank approx. dataset matrix, while retaining the critical latent factors and ignoring noise.
- Optimal low rank needs to be tuned
- SVD is computationally expensive

- SVD Libraries:
  - Matlab [U, S, V] = svds(A,256);
  - SVDPACKC http://www.netlib.org/svdpack/
  - SVDLIBC http://tedlab.mit.edu/~dr/SVDLIBC/
  - GHAPACK http://www.dcs.shef.ac.uk/~genevieve/ml.html

Predictive Model

- Ver. 2: SVD+kNN



Order History
Cart Events

User engagement vector

users

item	1	2	3	4	5	6	7	8	9	10	...	n
17	1		.5			1		.5		1		

Data Normalization

SVD

latent factors

item	1	2	3	4	5
17	.21	.19	1.2	0.1	0.4

k-Nearest Neighbors (kNN) in reduced space

latent factors

item	1	2	3	4	5
1	.21	.14	1.7	0.11	0.4
2	-.35	-.29	1.2	0.4	-0.5
17	.45	.17	0.02	0.06	.32
18	.21	.19	.15	-0.1	0.6
..					
m	.1	-.38	.8	0.31	0.4

Cross-sell & Up-sell Recommendation

Synthetic Data Set

- Why do we use synthetic data set?

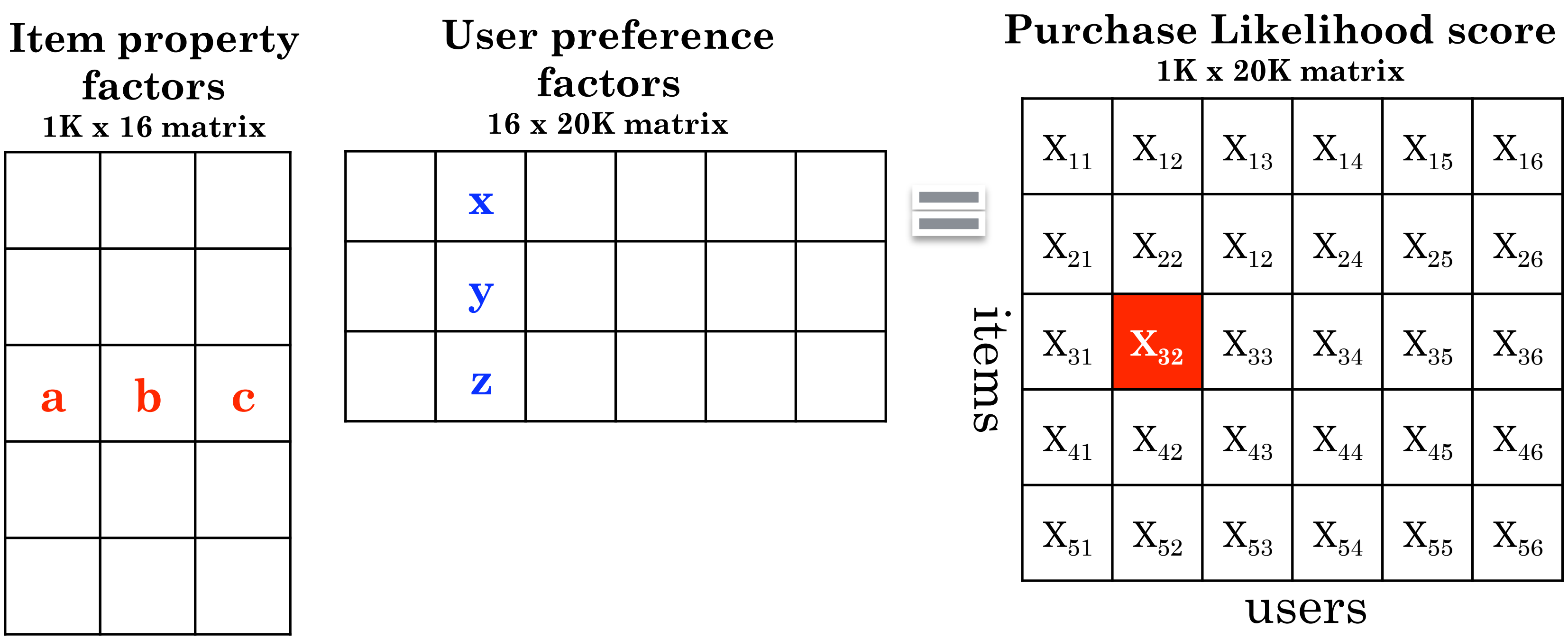
- So we can test our new model in a controlled environment

Synthetic Data Set

- 16 latent factors synthetic e-commerce data set
  - Dimension: 1,000 (items) by 20,000 (users)
  - 16 user preference factors
  - 16 item property factors (non-negative)
  - Txn Set: n = 55,360 sparsity = 99.72 %
  - Txn+Cart Set: n = 192,985 sparsity = 99.03%
  - Download: http://www.IntelligentMining.com/dataset/

user_id	item_id	type
10	42	0.25
10	997	0.25
10	950	0.25
11	836	0.25
11	225	1

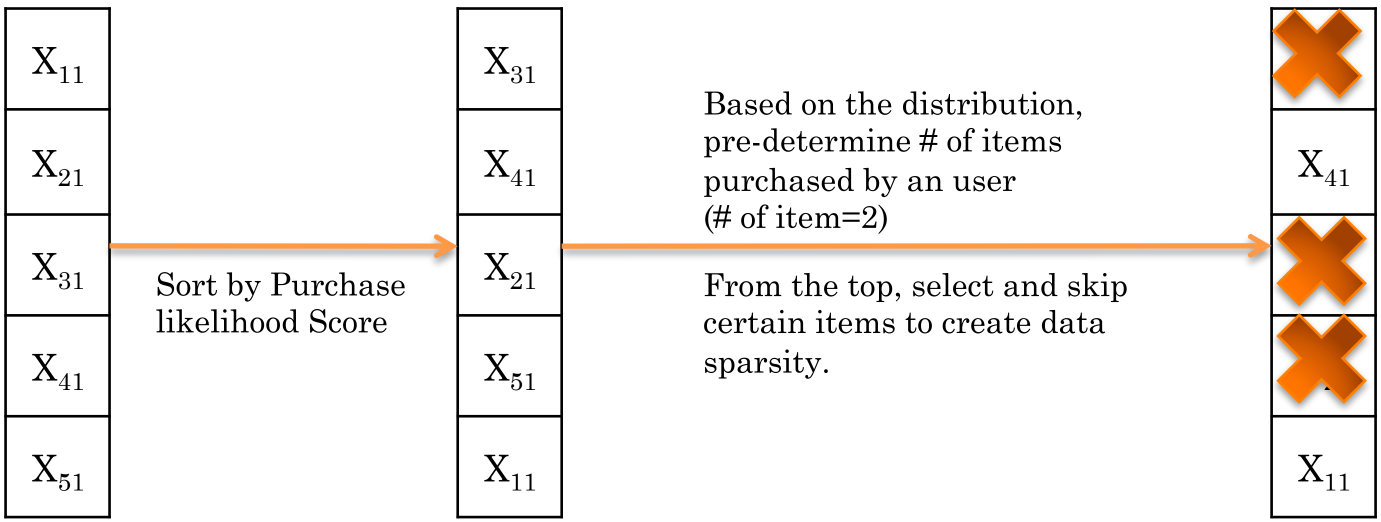
Synthetic Data Set



$$X_{32} = (a, b, c) \,.\, (x, y, z) = a * x + b * y + c * z$$

X_{32} = **Likelihood of Item 3 being purchased by User 2**

Synthetic Data Set



- User 1 purchased Item 4 and Item 1

Experiment Setup

- Each model (Random / kNN / SVD+kNN) will generate top 20 recommendations for each item.
- Compare model output to the actual top 20 provided by synthetic data set
- Evaluation Metrics :
  - Precision %: Overlapping of the top 20 between model output and actual (higher the better)

$$\text{Precision} = \frac{\left|\{Found\_Top20\_items\} \cap \{Actual\_Top20\_items\}\right|}{\left|\{Found\_Top20\_items\}\right|}$$

  - Quality metric: Average of the actual ranking in the model output (lower the better)

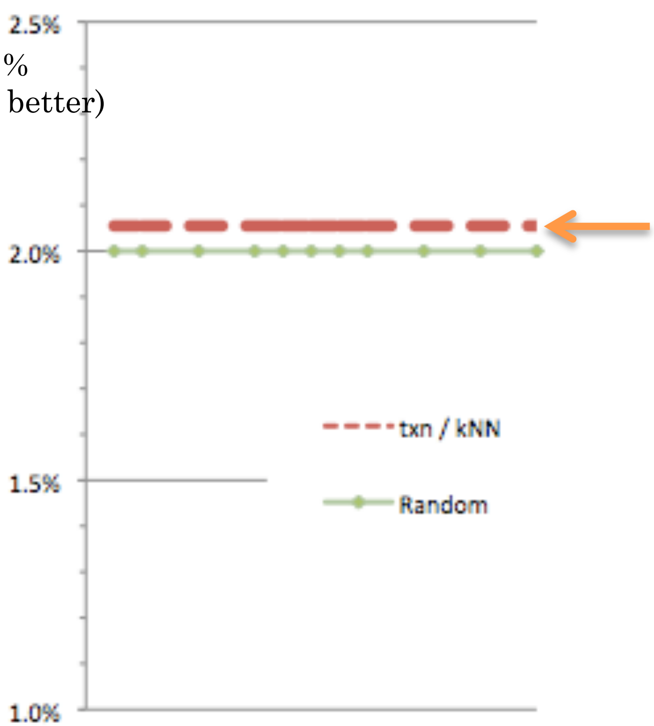
1	2	30	47	50	21

1	2	368	62	900	510

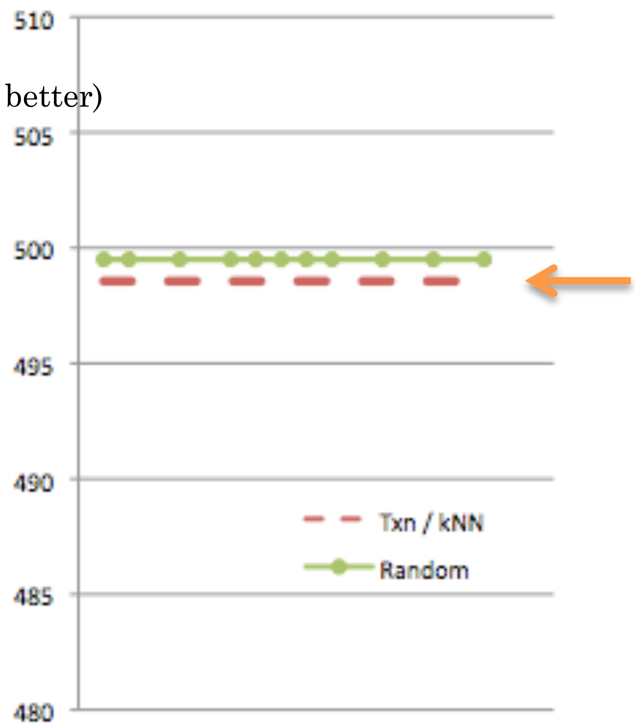
Experimental Result

- kNN vs. Random (Control)

Precision %
(higher is better)

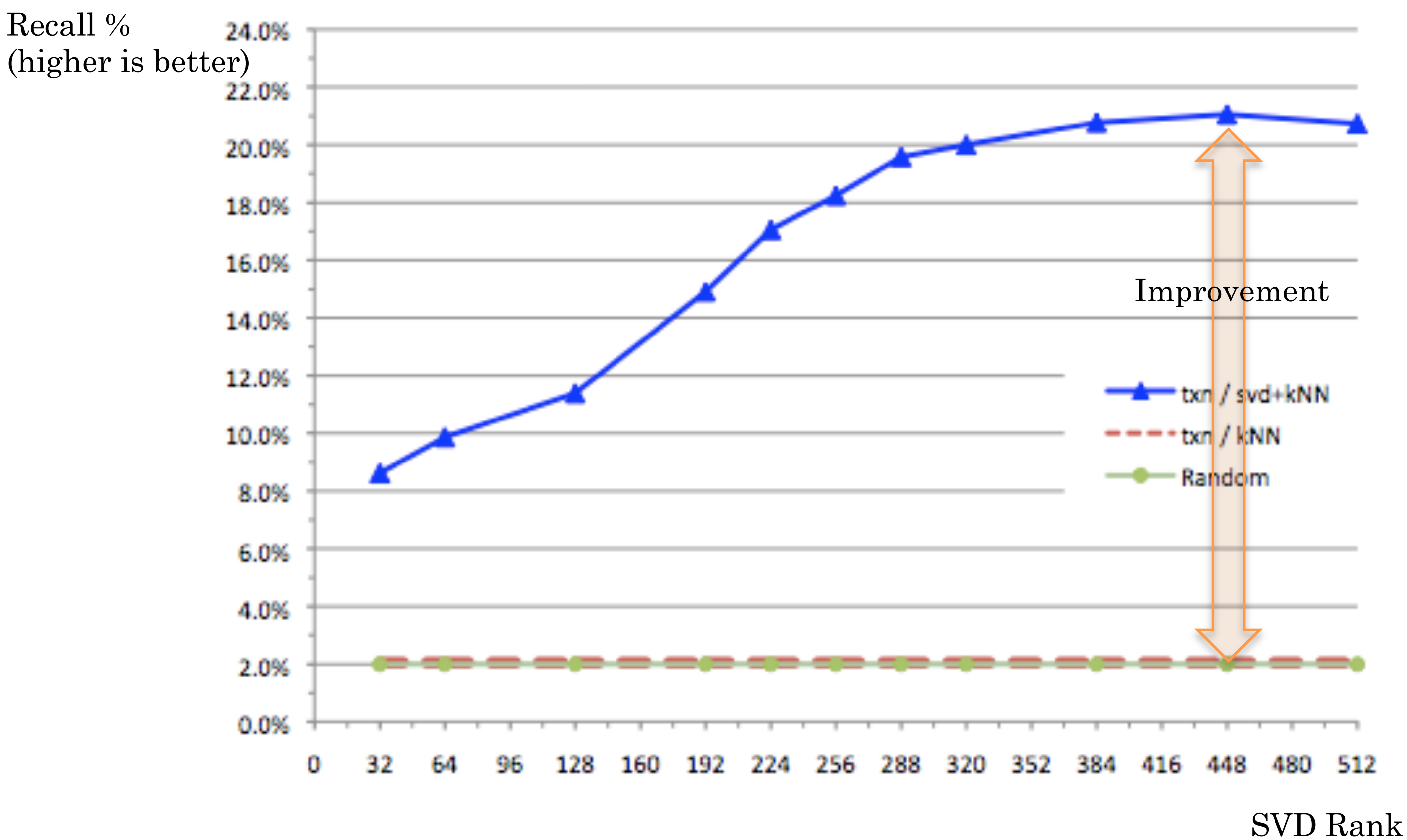


Quality
(Lower is better)



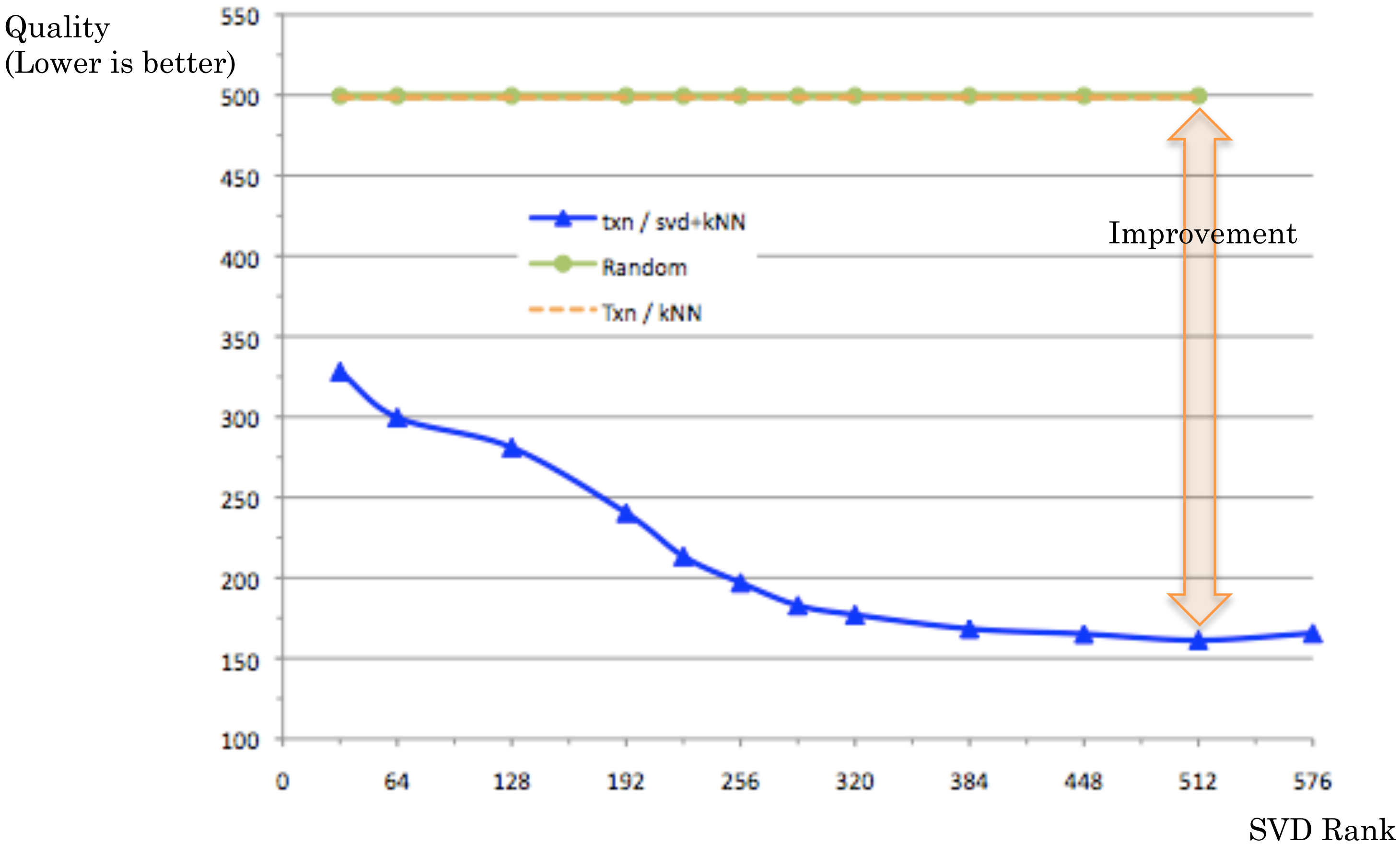
Experimental Result

- Precision % of SVD+kNN



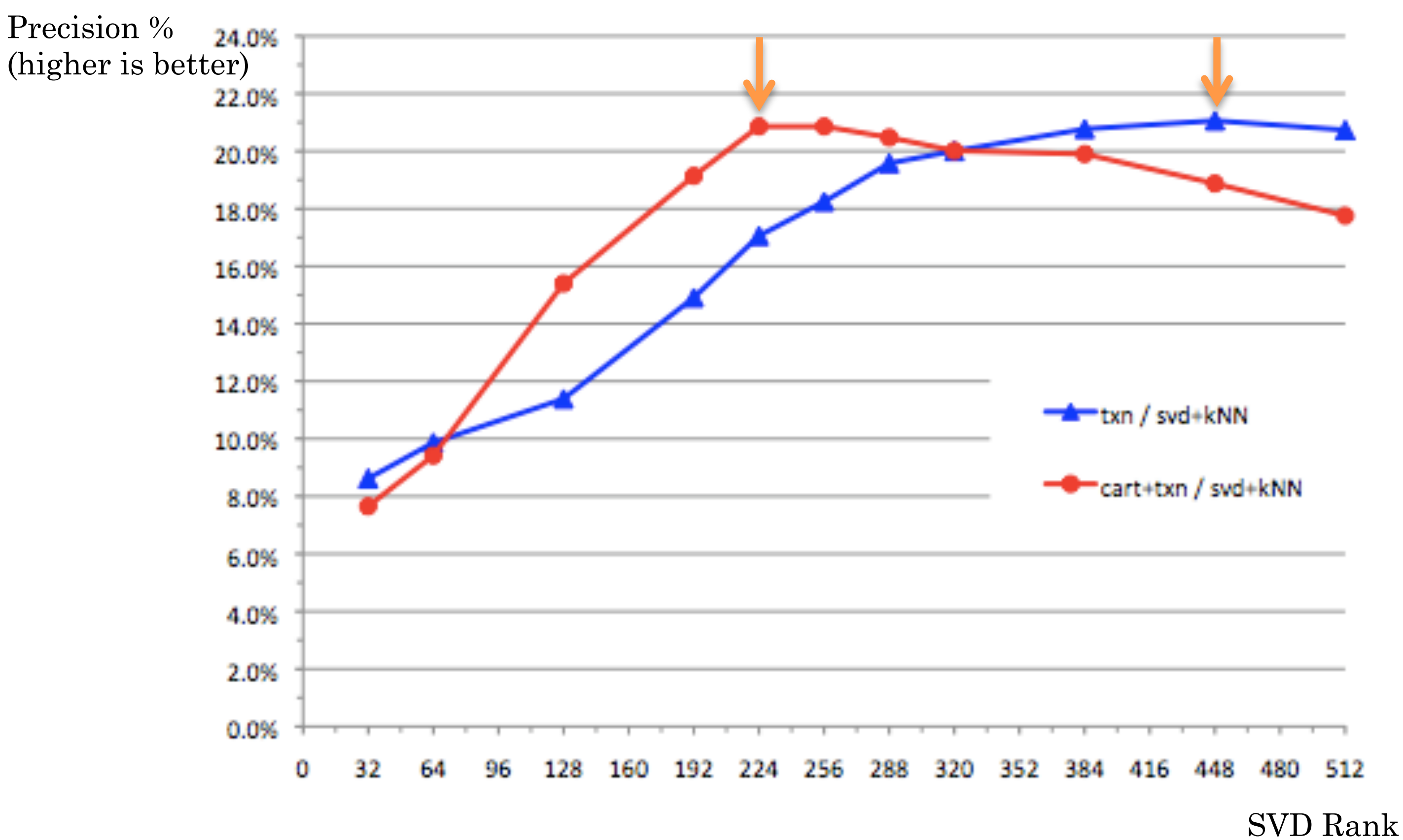
Experimental Result

- Quality of SVD+kNN



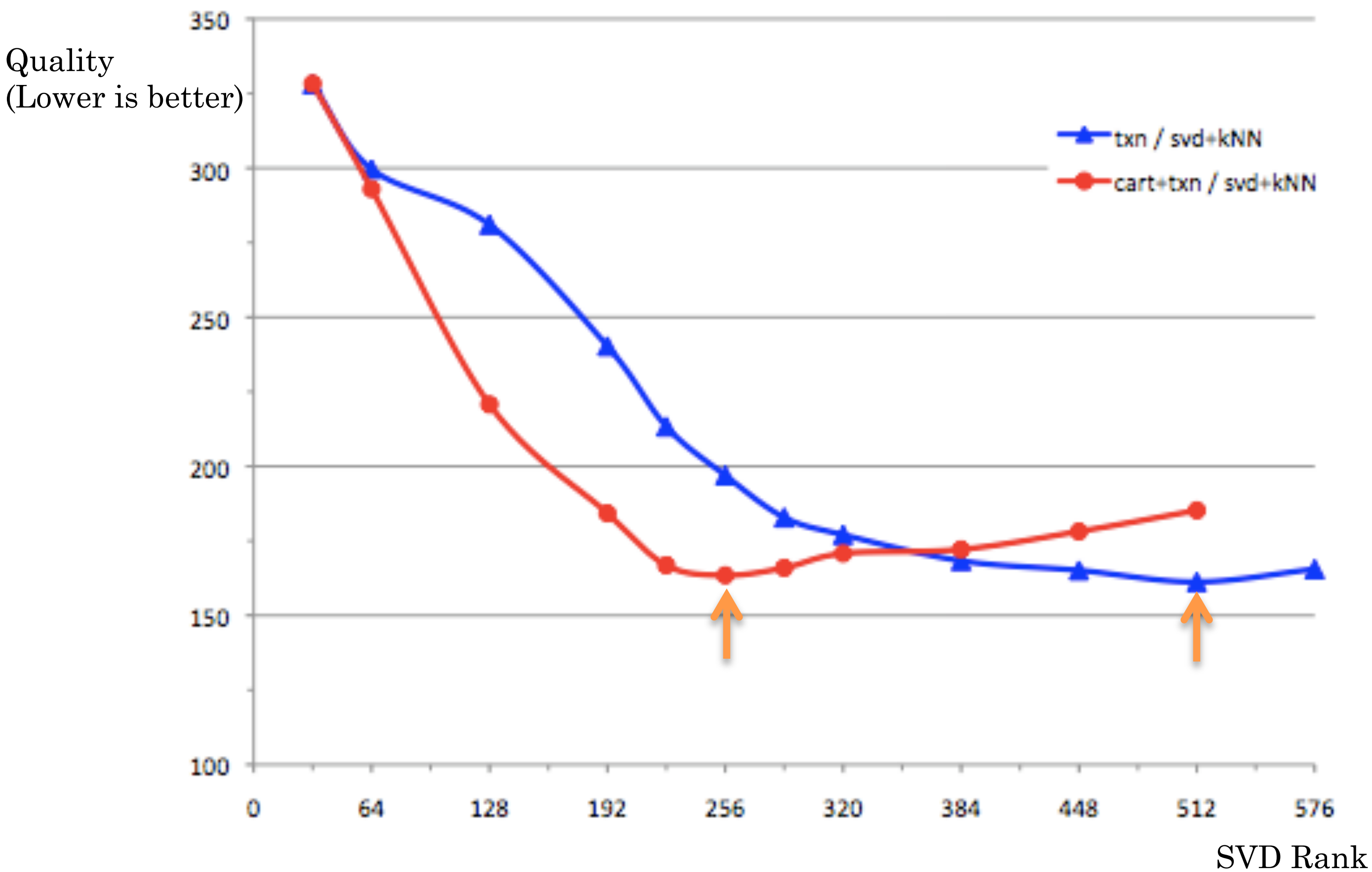
Experimental Result

- The effect of using Cart data



Experimental Result

- The effect of using Cart data



Outline

- Predictive modeling methodology
- k-Nearest Neighbor (kNN) algorithm
- Singular value decomposition (SVD) method for dimensionality reduction
- Using a synthetic data set to test and improve your model
- Experiment and results

References

- J.S. Breese, D. Heckerman and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," in Proceedings of the Fourteenth Conference on Uncertainity in Artificial Intelligence (UAI 1998), 1998.
- B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in Proceedings of the Tenth International Conference on the World Wide Web (WWW 10), pp. 285-295, 2001.
- B. Sarwar, G. Karypis, J. Konstan, and J. Riedl "Application of Dimensionality Reduction in Recommender System A Case Study" In ACM WebKDD 2000 Web Mining for E-Commerce Workshop
- Apache Lucene Mahout http://lucene.apache.org/mahout/
- Cofi: A Java-Based Collaborative Filtering Library http://www.nongnu.org/cofi/

Thank you

- Any question or comment?