Attack Resistant Collaborative Filtering

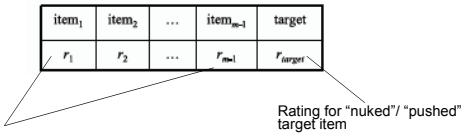
(B. Mehta, T.Hofmann, P.Fankhauser)

Attacks in Collaborative Filtering

- Creating multiple profiles and injecting opinion in order to influence others
- K-Nearest-Neighbour-Algorithm for Collaborative filtering very vulnerable
- Want algorithm naturally robust against attacks.

General form of an attack profile

- Profile Injection Attacks (Shilling attacks)
  - Either “nuke” or push” a item



Kind of attacks considered

- *Random* attacks

Item 1	Item 2	…	Item m	Item Target
Average($R_{(1…m)}$)	Average($R_{(1…m)}$)	…	Average($R_{(1…m)}$)	R_{target}

- *Average* attacks

Item 1	Item 2	…	Item m	Item Target
Average($R_{(1)}$)	Average($R_{(2)}$	…	Average($R_{(m)}$)	R_{target}

- *Bandwagon* attacks

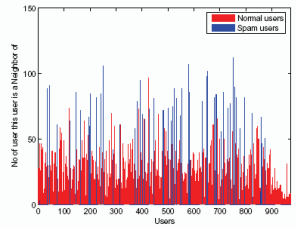
Item 1	Item 2	…	Item m	Item Target	Frequently rated Items n
Average($R_{(1)}$)	Average($R_{(2)}$	…	Average($R_{(m)}$)	R_{target}	High Rate 1…n

Characteristics of spam profiles (1)

- Spam users are highly correlated
  - Very high correlation-coefficient (> 0,9)
  - Covariance between spam users quite low compared to authentic users
- Spam users work together
  - Magnification of each other's effects, pushing attacked item together
- Low deviation from mean votes value, But high deviation from the mean for the attacked item
  - Significant increase of similarity with other users

Characteristics of spam profiles (2)

- High similarity with large number of users
  - High Correlation --> Authoritative neighbor and prominent in the set of k-nearest-neighbors



PCA (Principal Component Analysis)

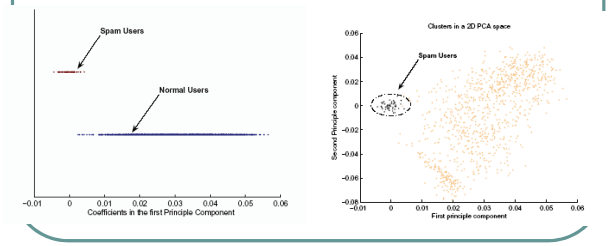
- Reduction of dimension of datasets though preserving the information
- selecting subset which virtually contains all the information
- Our case:
  - Attacker profile highly correlated to each other, AS WELL AS with normal users
  - Hence, they contain the least information (in the sense of variance)

Identifying the attackers: VarSelect

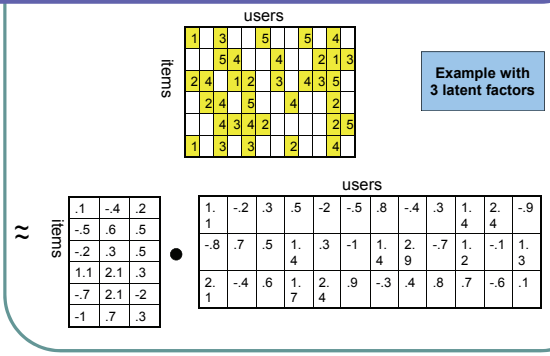
- **Main Idea:** Perform PCA on rating matrix to find the top *d* factors, and score users according to correlation with these factors. Users with the lowest scores are flagged.
  - Works because attack profiles have low variance, so are unlikely to be correlated with the factors with the highest variance.

VarSelect

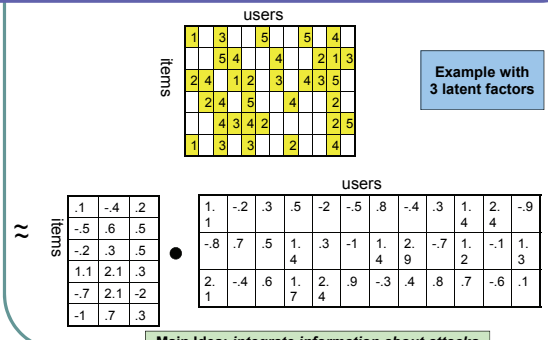
- Clear clusters of data, sophisticated and unsophisticated attacks
- Good for variety of attacks, attack sizes, filler sizes, Uncoordinated attacks (like in real life)



SVD for Collaborative Filtering



SVD for Collaborative Filtering



Main Idea: *integrate information about attacks from VarSelect into this SVD for CF.*

Training SVD: Hebbian Learning

- Iterative process equivalent to minimizing some sum of residuals

$$E = \sum r(x)^2$$

- Can be done using gradient descent
- Idea: ignore entries from flagged users

VarSelect SVD

Algorithm 1 VarSelectSVD ($\mathbf{D}$)

1: $\mathbf{D} \leftarrow$ z-scores($\mathbf{D}$) {$\mathbf{D}$ has N users and M items}
2: $\mathbf{U}\Lambda\mathbf{V}^{\mathrm{T}}$= SVD($\mathbf{D}$,3) {Get 3 principal components $\mathbf{U}^{T}$}
3: $PCA_1 \leftarrow \mathbf{U}(:,1), PCA_2 \leftarrow \mathbf{U}(:,2), PCA_3 \leftarrow \mathbf{U}(:,3)$ {First 3 PC loadings }
4: **for all** columnid $user$ in $\mathbf{D}$ **do**
5: $Score(user) \leftarrow (|PCA_1(user)| + |PCA_2(user)| + |PCA_3(user)|)/3$ {using LC ranking scheme}
6: **end for**
7: Normalize and Sort $Score$ {$Score$ now sum to 1.}
8: $r_1 \leftarrow$ number of users with $Score$ below $\frac{1}{N}$
9: $r_2 \leftarrow N/5$ {Cutoff set to 20%.}
10: $r \leftarrow min(r_1, r_2)$
11: Flag top r users with smallest $Score$ values
12: **for** Factor f_k with $k \leftarrow 1$ to d **do**
13: $\mathbf{D} = \mathbf{D} - \mathbf{G}_{k-1} \cdot \mathbf{H}_{k-1}^{\mathrm{T}}$
14: **repeat**
15: $res_{ij} = \mathbf{D}_{ij} - \hat{G}_i \cdot \hat{H}_j$ {set $\kappa = 0.01$}
16: $\triangle\hat{G}_i = \lambda(\hat{H}_j \cdot res_{ij} - \kappa \cdot \hat{G}_i)$
17: **if** u is not flagged **or** $v_{min} < D_{ij} < v_{max}$ **then**
18: $\triangle\hat{H}_j = \lambda(\hat{G}_i \cdot res_{ij} - \kappa \cdot \hat{H}_j)$
19: **end if**
20: **until** Convergence of $\hat{G}_i, \hat{H}_j$ for all i, j
21: **end for**

Output: Return Matrix factors $\mathbf{G}, \mathbf{H}$

Ignoring flagged users

Previous Robust CF Algorithms

- Robust Matrix Factorization (RMF)
  - Idea: instead of minimizing squared residuals in SVD, minimize w.r.t.

$$\rho(r) = \begin{cases} |r| \le \gamma & \frac{1}{2\gamma} r^2, \\ |r| > \gamma & |r| - \frac{\gamma}{2} \end{cases}$$

  - This gives less weight to outliers.

Evaluation

- MovieLens dataset
  - 6040 users
  - 3942 movies.
- Randomly takes out 20% of the dataset as out-of-sample test
- Generates attacks by adding profiles according to standard models.

Evaluation

- Compare with k-NN, SVD-GHA, and RMF.
- Metrics:
  - Prediction shift: change in prediction of ratings of attacked item
  - Hit ratio: Average change in the top-k list for all users.
  - Mean Average Error: change in prediction value for all missing ratings.

Controlling for attack size

- Filler size:
  - % of items voted for in the attacker profile. (The more fillers, the more inconspicuous the attack is. Since attacker behaves “normally” for a while.)
- Attack size:
  - number of injected attack profiles, and is measured as a percentage of the pre-attack training set.

Model Tuning

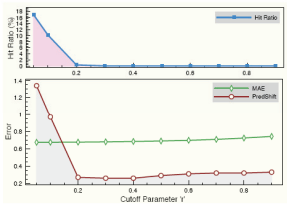


Figure 1: MAE, Pred Shift and Hit ratio for 5% Average Attacks (7% filler) with various values of r.

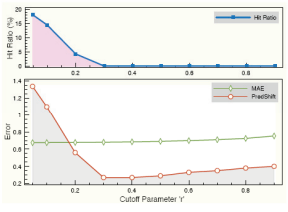


Figure 2: MAE, Pred Shift and Hit ratio for 5% Average Attacks (10% filler) with various values of r.

Idea: Need cuttoff to be large enough

Results: random attacks

Table 3: Effect of random Shilling attacks on various CF approaches

Attack size	Filler size	SVD-GHA		RMF		PCARobustCF	
		Hit Ratio	Pred Shift	Hit Ratio	Pred Shift	Hit Ratio	Pred Shift
1%	1%	4.90	1.23	2.15	0.96	0.00	0.26
	3%	5.32	1.22	2.21	0.97	0.00	0.25
	5%	5.90	1.23	2.59	0.96	0.00	0.25
	7%	6.57	1.23	2.30	0.95	0.00	0.25
	10%	6.49	1.19	2.08	0.92	0.01	0.33
3%	1%	9.42	1.63	14.34	1.71	0.00	0.25
	3%	12.63	1.69	12.18	1.69	0.00	0.26
	5%	12.71	1.65	11.32	1.61	0.00	0.29
	7%	13.30	1.62	10.91	1.56	0.01	0.35
	10%	12.18	1.55	10.10	1.51	0.24	0.40
7%	1%	22.74	2.07	25.43	2.20	0.00	0.24
	3%	24.07	2.04	23.34	2.10	0.00	0.28
	5%	21.77	1.95	21.17	1.98	0.05	0.31
	7%	19.86	1.85	21.06	1.92	2.46	0.60
	10%	17.28	1.73	20.04	1.53	4.69	0.78
10%	1%	30.54	2.21	30.53	2.29	0.00	0.24
	3%	30.92	2.16	27.72	2.20	0.10	0.35
	5%	26.22	2.04	25.93	2.10	2.74	0.66
	7%	22.70	1.93	25.28	2.03	5.09	0.84
	10%	19.50	1.78	21.60	1.89	7.86	0.99

Their algorithm is unaffected when attack size is small 1-7% or attack is concentrated (filler size 1-5%)

Results: average attacks

Table 4: Effect of Average Shilling attacks on various CF approaches

Attack size	Filler size	SVD-GHA		RMF		PCARobustCF	
		Hit Ratio	Pred Shift	Hit Ratio	Pred Shift	Hit Ratio	Pred Shift
1%	1%	8.14	1.25	10.78	1.24	-0.02	0.38
	3%	10.11	1.28	9.09	1.25	-0.14	0.40
	5%	9.93	1.24	7.30	1.22	-0.11	0.40
	7%	10.67	1.24	7.79	1.22	0.14	0.42
	10%	11.10	1.24	6.29	1.19	0.82	0.74
3%	1%	17.16	1.59	18.92	1.61	-0.14	0.41
	3%	19.51	1.61	15.20	1.56	-0.13	0.41
	5%	19.93	1.58	14.87	1.54	0.11	0.43
	7%	19.12	1.55	13.04	1.27	1.74	0.57
	10%	19.50	1.52	13.98	1.49	13.44	1.01
7%	1%	31.24	1.86	24.16	1.85	-0.09	0.40
	3%	32.86	1.85	21.31	1.79	-0.02	0.44
	5%	31.66	1.80	21.60	1.73	6.43	0.75
	7%	30.64	1.78	20.78	1.71	16.25	1.12
	10%	31.76	1.76	21.76	1.68	24.32	1.38
10%	1%	39.57	1.96	29.44	1.96	0.08	0.39
	3%	39.98	1.94	23.71	1.89	0.82	0.37
	5%	38.27	1.90	25.49	1.84	14.11	1.08
	7%	38.20	1.87	24.88	1.79	23.53	1.35
	10%	36.86	1.77	24.39	1.70	31.44	1.48

Average attacks are more ponent than random attacks.

Result: prediction precision

Table 6: Overall MAE on a test set for various CF approaches after attack profiles (Average Model) have been inserted.

Attack Size	Filler Size	SVD without attack	SVD	RMF	SVD with VarSelect	Pearson based k-NN
3%	2%	0.6755	0.6760	0.6691	0.6762	0.8095
	5%	0.6755	0.6763	0.6694	0.6768	0.8115
	10%	0.6755	0.6776	0.6699	0.6763	0.8135
5%	2%	0.6755	0.6761	0.6689	0.6764	0.8077
	5%	0.6755	0.6767	0.6699	0.6761	0.8087
	10%	0.6755	0.6783	0.6703	0.6769	0.8093
10%	2%	0.6755	0.6766	0.6719	0.6768	0.8062
	5%	0.6755	0.6774	0.6723	0.6783	0.8062
	10%	0.6755	0.6785	0.6769	0.6778	0.8073

Under attacks, their algorithm is slightly worse than RMF, but much better than naïve K-NN.

Summary

- A SVD-based algorithm that both identifies attackers and perform CF taking into account this information
- Performs well under random or average attacks, when attack size is reasonably small or concentrated.

Class Comments

- Questionable precision
  - Similar normal users are removed
- Unconvincing attack types considered
- Odd that model predicts flagged users' behavior
- Not clear how complicated attacks can be handled
- Hard to tune tau.
- Worse for large datasets