Evaluating Recommender Systems

How do you tell if users are satisfied with RS?

- **RS returns products relevant to users**
  - How do you assess this at scale?

- **Before evaluation, you should have data to train your model.**
  - What we test is actually the performance of a recommendation model.

Goals of Evaluation

- **Selection of recommender algorithms**
  - Many algorithms, many choices
- **Tuning recommender algorithms**
  - Most algorithms have hyperparameters those couldn’t be trained
    - KNN (K-Nearest Neighborhood)
  - Which feature is useful
- **Knowing user experience**

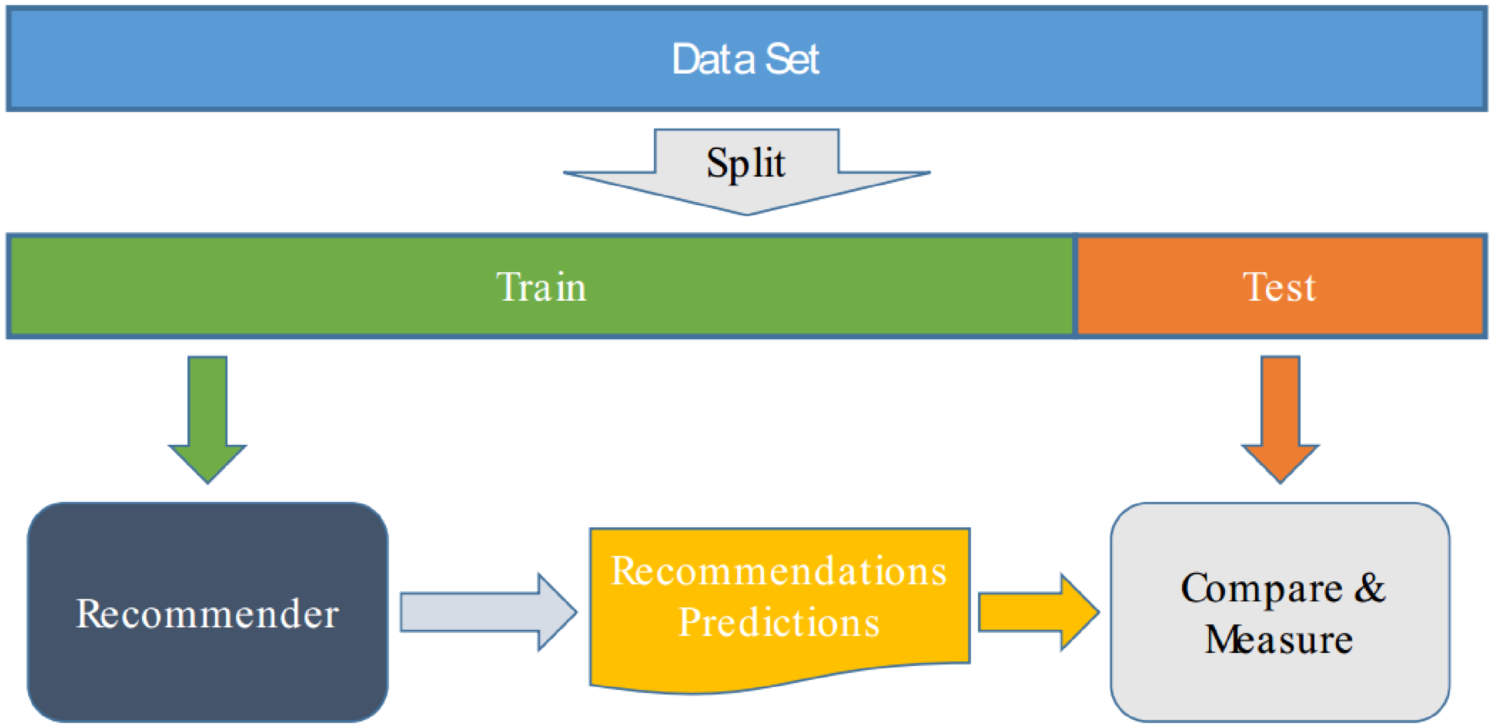
Evaluation in RS

- **In which scenario?**
- **1. Offline Evaluation**
  - Use historical data to simulate the testing
- **2. Online Evaluation**
  - Take real-world users for experiments

- **What type of a task?**
- **1. Rating Prediction**
  - Rating: a measurement of absolute or relative ‘goodness’ of items
- **2. Top-N Recommendation List (Ranking)**
  - Binary relevance: if an item is ‘good’ or not

Evaluation Framework (Offline evaluation)

- **Data segmentation**

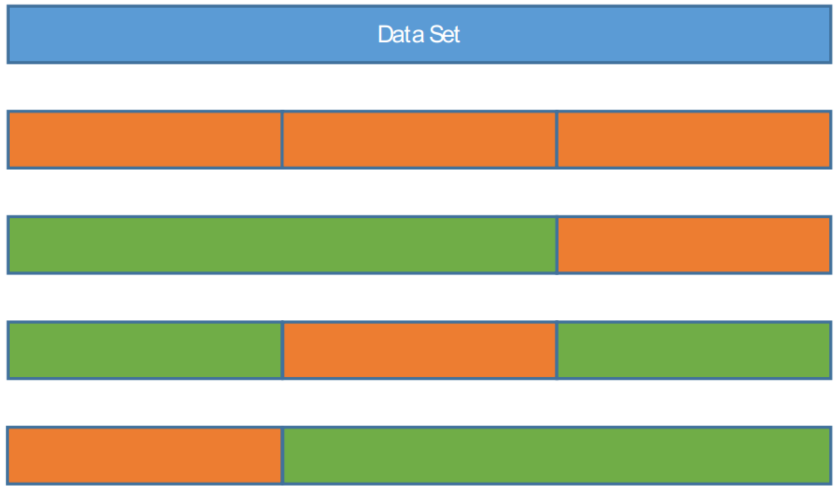


Ways of Splitting Data

- **Split ratings**
  - Commonly used
  - Performing random splitting
- **Split users**
  - Allows more control for measuring expected user experience
  - Cold-start user recommendation scenario
- **Split items**
  - Cold-start item recommendation scenario

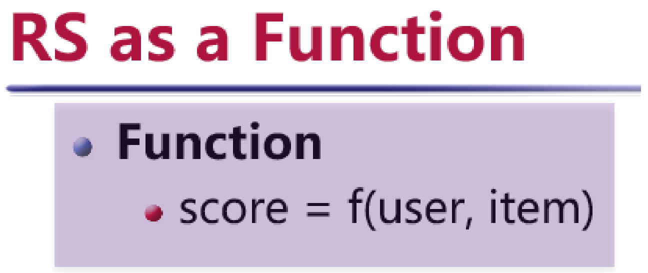
Cross Validation (Offline evaluation)

- **Intuition: what if train-test split randomly selects mostly ‘easy’ and ‘hard’ users?**
  - Not stable for a small test set
- **Solution: multiple splits**
  - Partition data into k-partitions (k = 5 or 10)
  - For each partition, train on others and test with it



Offline Evaluation (Tasks)

- **For offline evaluation, the user-item interactions are already recorded**
- **For a specific user, the score for item *i* is:**
  - Predicted score: p_i
  - Ground-truth score: r_i, from records.



RS as a Function

- **Function**
  - score = f(user, item)

- **Rating Prediction**
  - Directly use the score
- **Top-N Recommendation List (Ranking)**
  - Should first use the scores (for all items) to generate a item ranking list

Evaluation in RS – Rating Prediction

- **Datasets with items rated by users**
  - MovieLens datasets 100K-10M ratings
  - Netflix 100M ratings
  - Amazon datasets
- **Historical user ratings constitute ground truth**
- **Metrics measure error rate**
  - Mean Absolute Error (*MAE*) computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | p_i - r_i |$$

  - Root Mean Square Error (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (p_i - r_i)^2}$$

Example

Nr.	UserID	MovieID	Rating (r_i)	Prediction (p_i)	$\|p_i-r_i\|$	$(p_i-r_i)^2$
1	1	134	5	4.5	0.5	0.25
2	1	238	4	5	1	1
3	1	312	5	5	0	0
4	2	134	3	5	2	4
5	2	767	5	4.5	0.5	0.25
6	3	68	4	4.1	0.1	0.01
7	3	212	4	3.9	0.1	0.01
8	3	238	3	3	0	0
9	4	68	4	4.2	0.2	0.04
10	4	112	5	4.8	0.2	0.04
					4.6	5.6

- **MAE = 0.46**
- **RMSE = 0.75**

Consideration of Average

- **For RMSE and MAE, average operation usually means:**
  - Average over all ratings
- **Alternative, average over user averages**
- **Difference:**
  - What if one user has 3000 ratings and another 10?

Evaluation RS - Top-N Recommendation

- **Top-N Recommendation**
  - Provide a length-N recommendation list for each user
  - Examine the hit items in the recommendation list

Rank	Hit?
1	
2	X
3	X
4	X
5	

Evaluation RS - Top-N Recommendation

- **Top-N Recommendation**
  - Provide a length-N recommendation list for each user
  - Examine the hit items in the recommendation list

Rank	Hit?
1	
2	X
3	X
4	X
5	

X
X
X
X

- **Confusion matrix**

		Reality	
		Actually right	Actually wrong
Prediction	Recommended	True Positive (tp) 3	False Positive (fp) 2
	Not Recommended	False Negative (fn) 4	True Negative (tn)

All recommended items

All good items

Metrics: Precision and Recall

		Reality	
		Actually right	Actually wrong
Prediction	Recommended	True Positive (tp) 3	False Positive (fp) 2
	Not Recommended	False Negative (fn) 4	True Negative (tn)

Rank	Hit?	
1		
2	X	**X**
3	X	**X**
4	X	**X**
5		**X**

- **Precision: a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved**
  - E.g. the proportion of recommended movies that are actually good
  - 3/5

$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|\text{all recommendations}|}$$

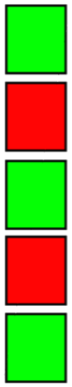
- **Recall: a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items**
  - E.g. the proportion of all good movies recommended
  - 3/7

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

Precision@K

- Set a rank threshold K
- Compute % relevant in top K
- Ignores documents ranked lower than K

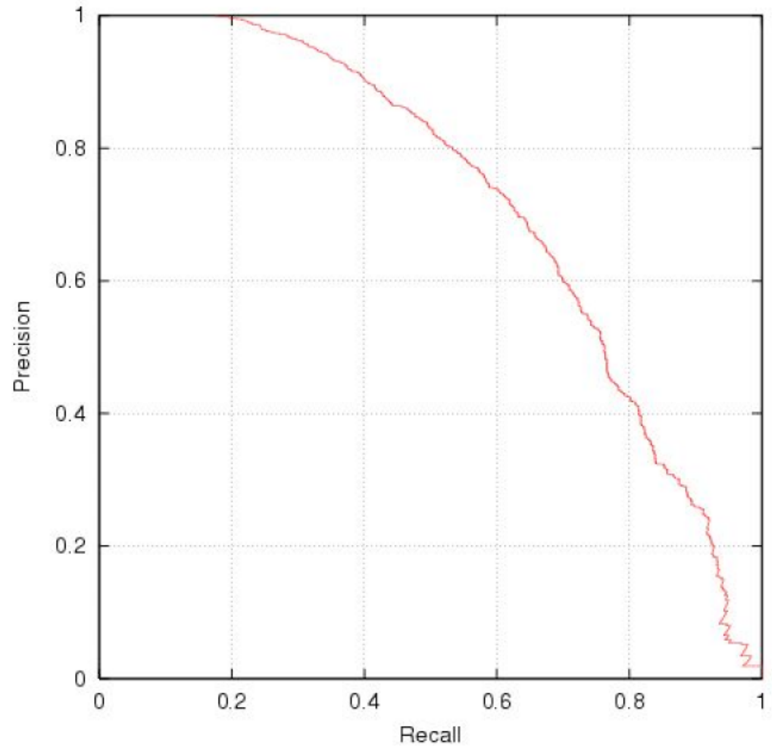
- Ex:
  - Prec@3 of 2/3
  - Prec@4 of 2/4
  - Prec@5 of 3/5



- In a similar fashion we have Recall@K

Precision vs. Recall

- **E.g. typically when a recommender system is tuned to increase precision, recall decreases as a result (or vice versa)**
  - E.g. recommend a lot a items
  - Recall is high
  - Precision may be very small



F_1 Metric

- **The F_1 Metric attempts to combine Precision and Recall into a single value for comparison purposes.**
  - May be used to gain a more balanced view of performance

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

- **Optimal value = 1**

Metrics: Rank position matters

For a user:

Actually good
Item 237
Item 899

Recommended (predicted as good)
Item 345
Item 237
Item 187

Recommended (predicted as good)
Item 237
Item 345
Item 187

- **Rank metrics extend recall and precision to take the positions of correct items in a ranked list into account**
  – Relevant items are more useful when they appear earlier in the recommendation list
  – Particularly important in recommender systems as lower ranked items may be overlooked by users

Rank	Hit?
1	
2	X
3	X
4	X
5	

Metrics: Rank Score

- **Rank Score extends the recall metric to take the positions of correct items in a ranked list into account**
  - Particularly important in recommender systems as lower ranked items may be overlooked by users
- **Rank Score is defined as the ratio of the Rank Score of the correct items to best possible Rank Score for the user, i.e.**

$$rankscore = \frac{rankscore_p}{rankscore_{\max}}$$

$$rankscore_p = \sum_{i \in h} 2^{-\frac{rank(i)-1}{\alpha}}$$

$$rankscore_{\max} = \sum_{i=1}^{|T|} 2^{-\frac{i-1}{\alpha}}$$

Where:
- *h* is the set of correctly recommended items, i.e. hits
- *T* is the set of all items of interest
- *α* is the *ranking half life*, i.e. an exponential reduction factor

- when *α* -> ∞, *rankscore -> recall*

Example

- **Assumptions:**
  - |T| = 3
  - Ranking half life (alpha) = 2

Rank	Hit?
1	
2	X
3	X
4	X
5	

$$rankscore = \frac{rankscore_p}{rankscore_{\max}} \approx 0.71$$

$$rankscore_p = \frac{1}{2^{\frac{2-1}{2}}} + \frac{1}{2^{\frac{3-1}{2}}} + \frac{1}{2^{\frac{4-1}{2}}} = 1.56$$

$$rankscore_{\max} = \frac{1}{2^{\frac{1-1}{2}}} + \frac{1}{2^{\frac{2-1}{2}}} + \frac{1}{2^{\frac{3-1}{2}}} = 2.21$$

$$rankscore = \frac{rankscore_p}{rankscore_{\max}}$$

$$rankscore_p = \sum_{i \in h} 2^{-\frac{rank(i)-1}{\alpha}}$$

$$rankscore_{\max} = \sum_{i=1}^{|T|} 2^{-\frac{i-1}{\alpha}}$$

Metrics: Normalized Discounted Cumulative Gain

Rank	Hit?
1	
2	X
3	X
4	X
5	

- **Discounted cumulative gain (DCG)**
  - Logarithmic reduction factor

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \qquad \mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i}-1}{\log_2(i+1)}$$

Where:
- *p* denotes the position up to which relevance is accumulated
- rel_i returns the relevance of recommendation at position *i (0/1, or other scales)*

- **Idealized discounted cumulative gain (IDCG)**
  - Assumption that items are ordered by decreasing relevance

$$\mathrm{IDCG_p} = \sum_{i=1}^{|REL|} \frac{2^{rel_i}-1}{\log_2(i+1)}$$

- **Normalized discounted cumulative gain (nDCG)** $\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p}$
  - Normalized to the interval [0..1]

Example

- **Assumptions:**
  - |T| = 3
  - Ranking half life (alpha) = 2

Rank	Hit?
1	
2	X
3	X
4	X
5	

$$DCG_5 = \frac{1}{\log_2(1+2)} + \frac{1}{\log_2(1+3)} + \frac{1}{\log_2(1+4)} = 1.56$$

$$IDCG_5 = \frac{1}{\log_2(1+1)} + \frac{1}{\log_2(1+2)} + \frac{1}{\log_2(1+3)} = 2.13$$

NDCG=DCG/IDCG=0.73

Mean Average Precision

- Mean Average Precision (M*AP*) is a ranked precision metric that places emphasis on highly ranked correct predictions (hits)

- Essentially it is the average of precision values determined after each successful prediction, i.e.

Rank	Hit?
1	
2	X
3	X
4	X
5	

$$AP = \frac{1}{3}\left(\frac{1}{2} + \frac{2}{3} + \frac{3}{4}\right) = \frac{23}{36} \approx 0.639$$

$$AP = \frac{1}{3}\left(\frac{1}{1} + \frac{2}{4} + \frac{3}{5}\right) = \frac{21}{30} = 0.7$$

Rank	Hit?
1	X
2	
3	
4	X
5	X

Mean Reciprocal Rank

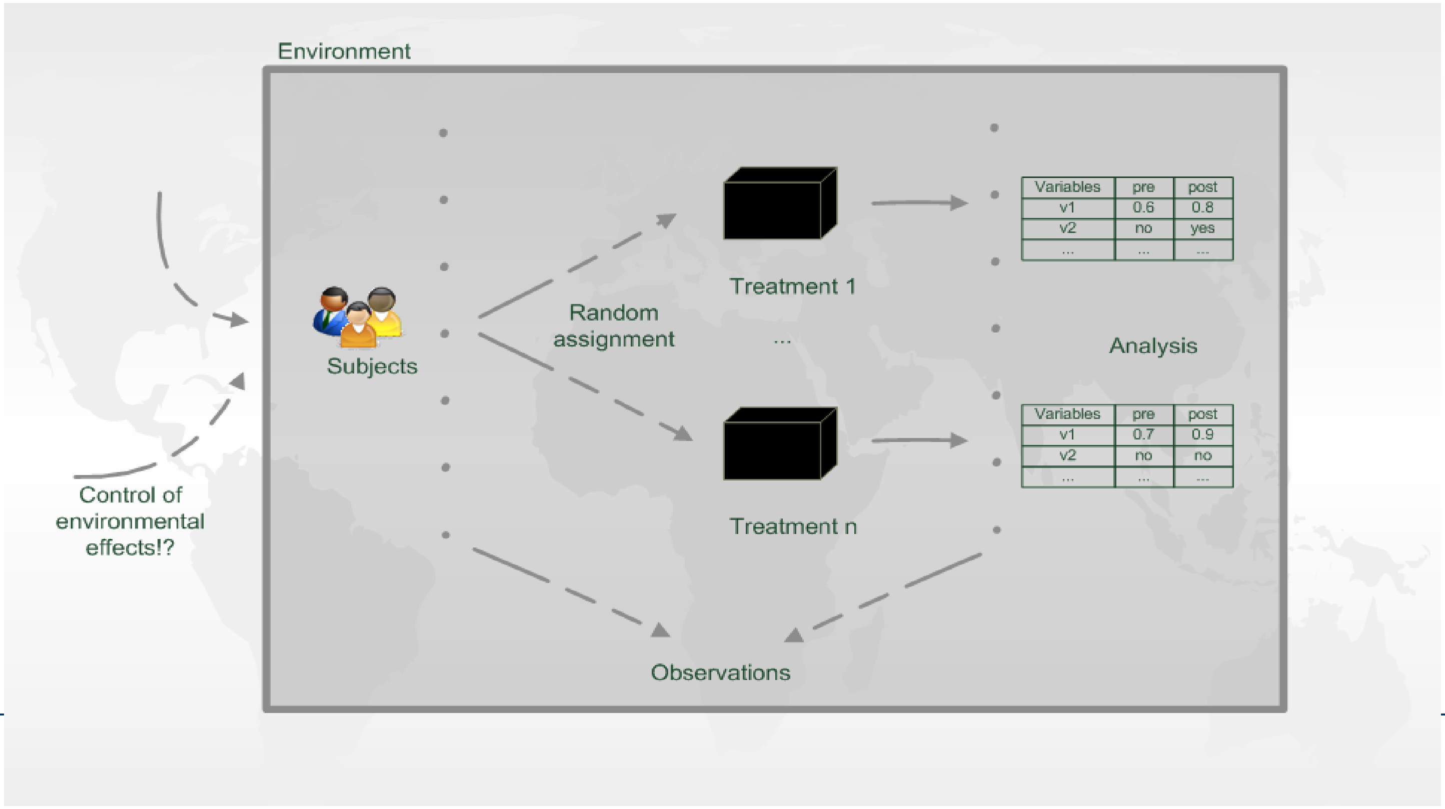
- **Very simple ranking metric**
  - Denoting where is the first relevant item?
  - For binary relevance judgement
- **Detailed computational procedure**
  - For each user u
    - Find rank k_u of its first relevant recommendation
    - Compute reciprocal rank $\frac{1}{k_u}$
  - Overall performance

$$\mathrm{MRR}(O, U) = \frac{1}{|U|} \sum_{u \in U} \frac{1}{k_u}$$

Online Evaluation

Online Evaluation

- **Evaluate with the behavior of real users**
  - A/B test
  - Still refer to previous measures for numerical evaluation



Variables	pre	post
v1	0.6	0.8
v2	no	yes
...	...	...

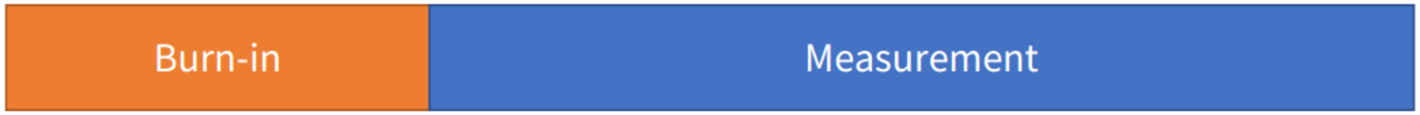
Variables	pre	post
v1	0.7	0.9
v2	no	no
...	...	...

Online Evaluation

- **Benefits**
  - Examine impact on actual user behavior
  - Controlled experiment results in reliable knowledge
  - Powerful ability to measure recommender’s impact on your business
- **Measuring impact**
  - Focus on a small number of Key Performance Indicators (KPIs)
    - Videos watched
    - Sales volume
    - Conversion rate
  - Pick KPIs carefully
    - Measuring the right thing
    - Good KPIs measure long-term customer value

Running Time of Online Evaluation

- **Don’t just run a couple days & measure**



- **Have 1-2 week burn-in period**
  - Record data, but don’t analyze for effect
  - Goal: flush out novelty effect
  - Important for anything new
  - Stop experiment if users react badly
- **Measurement Time**
  - Measure for multiple full weeks
  - Different days have different patterns

Other Metrics

Some Other Metrics

- **To understand metrics that relate more closely to business goals and/or user experience, specifically:**
  - Coverage
  - Novelty
  - Diversity
  - Personalization
  - Serendipity

Coverage

- **Coverage is the measure of the percentage of products for which a recommender can make a prediction**
  - Or a prediction that’s personalized
  - Or a prediction above a confidence threshold
- **There are three commonly used measurement methods:**
  - Calculate the proportion of all recommended commodities in the total number of commodities
  - Information Entropy
    - $H = -\sum_{i=1}^{n} p(i) log p(i)$
    - p(i) is the popularity of item i divided by the popularity of all items.
  - Gini Coefficient
    - $Gini = \frac{1}{n-1}\sum_{j=1}^{n}(2j-n-1)p(i_j)$
    - i_j is the j th item in the list of items sorted by item popularity p.

Popularity /Novelty

- **A simple metric that can be applied to recommendations (and list of recommendations) is the popularity of items recommended:**
  - Popularity can be measured as percentage of users who buy/rate the item, or used from external sources.
  - Novelty is often expressed as the inverse of popularity.

- **The easiest way to achieve novelty in a website is to filter out items that users have previously acted on in the site.**
- **The easiest way to measure novelty is to use the average popularity of the recommendation results, because the less popular the item, the more likely it is to be perceived as novel.**

Diversity

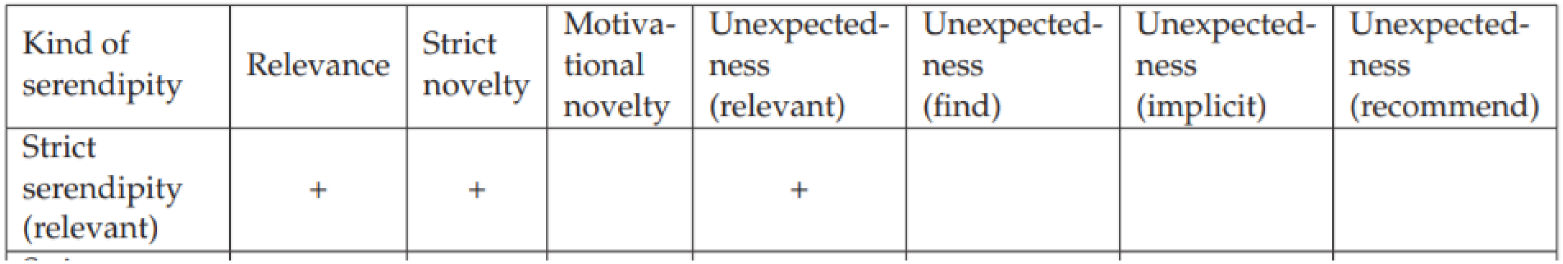
- **Measure of how different the recommended items are:**
  - Applied to a top-n list
  - Describes the dissimilarity between the two items in the recommendation list.
- **Intra-list similarity is the average pairwise similarity, lower score means higher diversity.**
  - Assuming that $s(i, j) \epsilon [0,1]$ defines the similarity between item i and item j, then the diversity of user u's recommendation list R(u) is defined as:
    - $Diversity\big(R(u)\big) = 1 - \frac{\sum_{i,j \epsilon R(u), i \neq j} s(i,j)}{\frac{1}{2}|R(u)|(|R(u)|-1)}$
- **A good recommendation system requires a combination of diversity requirements and the main interests of the user to avoid turning away customers who are not currently interested in a narrow portion of your catalog.**

Personalization

- **Several ways to measure how personalized a recommender is:**
  - Variance of predictions per item across users
  - Average difference in top-N lists among users
    - Related coverage metrics
  - User-perceived personalization (survey)

Serendipity

- Definition: "the occurrence and development of events by chance in a happy or beneficial way"
- In recommender systems: surprise, delight, not the expected results
- This is achieved by defining the similarity between the recommendation result, the user's historical preferences and the user's satisfaction with the recommendation result.



Kind of serendipity	Relevance	Strict novelty	Motiva-tional novelty	Unexpected-ness (relevant)	Unexpected-ness (find)	Unexpected-ness (implicit)	Unexpected-ness (recommend)
Strict serendipity (relevant)	+	+		+			

Discussion & summary

- General principles of empirical research an current state of practice in evaluating recommendation techniques were presented
- Focus on how to perform empirical evaluations on historical datasets
- Discussion about different methodologies and metrics for measuring the accuracy or coverage of recommendations.
- From a technical point of view, measuring the accuracy of predictions is a well accepted evaluation goal
  - but other aspects that may potentially impact the overall effectiveness of a recommendation system remain largely under developed.