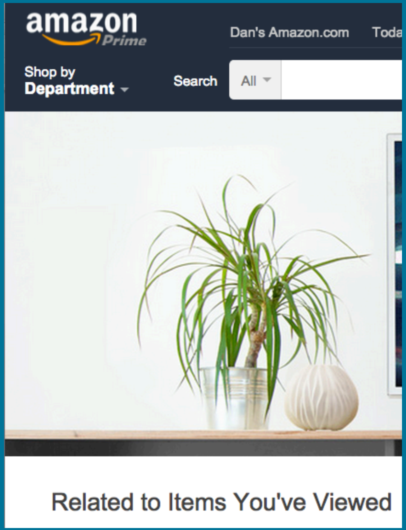
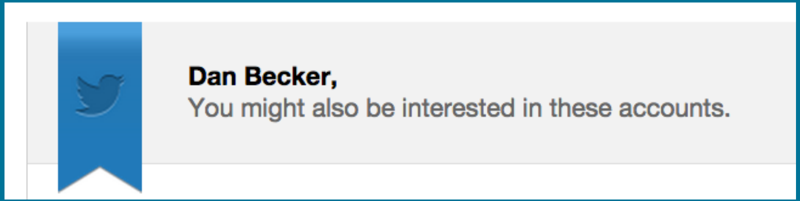
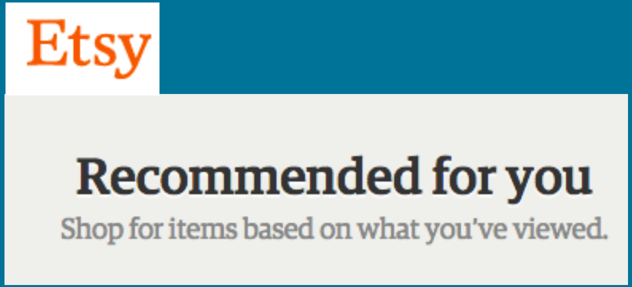
Recommender Systems

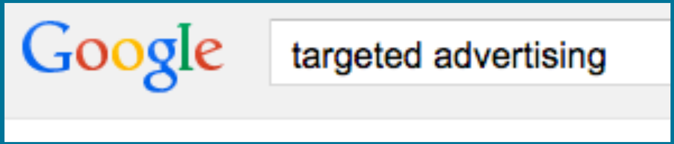
Dan Becker

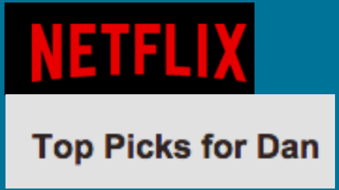
Recommenders Everywhere











Answer This

What You'll Learn

How to build, evaluate, deploy and improve models that make recommendations to users.

Example Data

Item

User

	A	B	C	D	…
Al	1	?	2	?	
Bob	?	2	3	4	
Cat	3	?	1	5	
Dan	?	2	?	?	
Ed	2	?	?	1	
…					

Binary Data

Item

User

	A	B	C	D	...
Al	0	1	0	1	
Bob	0	0	1	0	
Cat	0	1	1	1	
Dan	1	0	0	1	
Ed	0	1	0	0	
...					

Most Popular Techniques

Popularity

- Most viewed, most well-liked, etc
- Not customized to users

User-User

- Recommend items liked by users with similar ratings
- Computational challenges when more users than items

Item-Item

- Recommend items similar to what the current user rated favorably
- Covered in this lecture

Matrix Factorization Methods

- Estimate each user's underlying preferences for product characteristics
- Covered in next lectures

User Ratings as Vectors

Item

User

	A	B	C	D	...
Al	1	?	2	?	
Bob	?	2	3	4	
Cat	3	?	1	5	

Item Ratings as Vectors

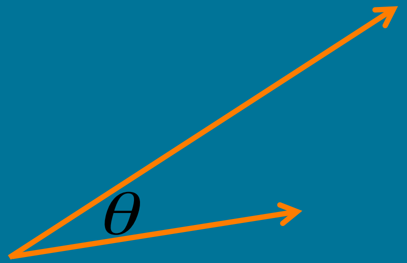
Item

User

	A	B	C	D
Al	1	?	2	?
Bob	?	2	3	4
Cat	3	?	1	5
Dan	?	2	?	?
Ed	2	?	?	1
…				

Measuring Similarity

Metric Name	Useful For	Formula
Cosine	Numeric Ratings	$\frac{A \bullet B}{[A][B]}$
Jacard	Binary / Event Data	$\frac{A \cap B}{A \cup B}$



Measuring Similarity

Metric Name	Useful For	Formula
Cosine	Numeric Ratings	$\frac{A \bullet B}{[A][B]}$
Jacard	Binary / Event Data	$\frac{A \cap B}{A \cup B}$

Cosine Sim

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_i)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_j)^2}}.$$

Adjusted Cosine Sim

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_u)^2}}.$$

Item-Item Recommendation Algorithm

In The Middle of the Night

- Compute similarity metric for all pairs of items
- Create “neighborhood” of similar items for each item to rate

At Request Time

- Predict scores for candidate items

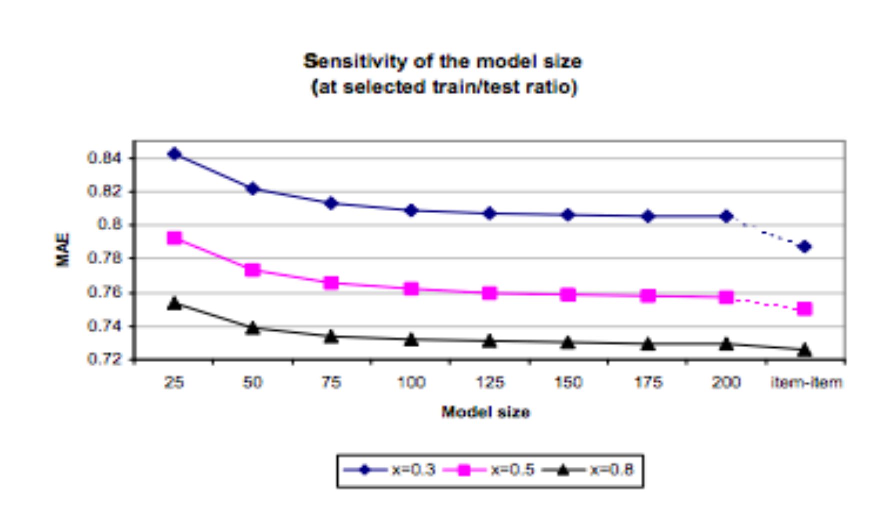
Item-Item Recommendation Algorithm

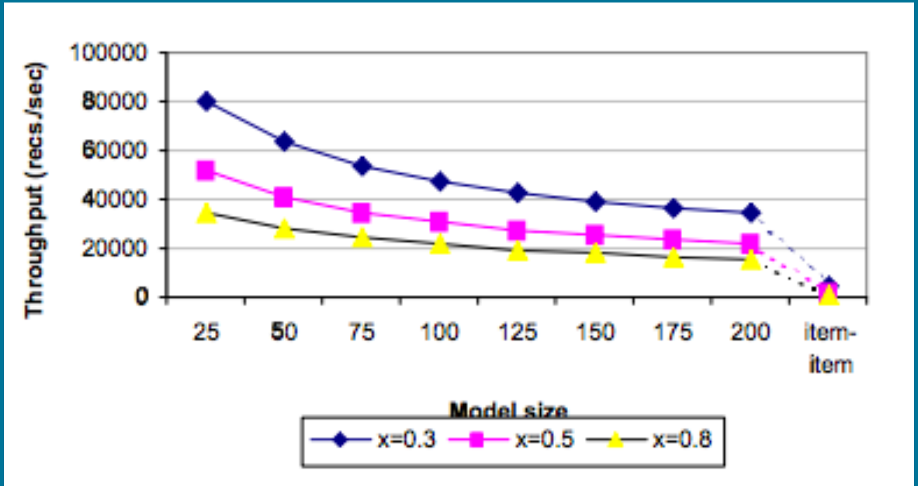
At Request Time

- Predict scores for candidate items

$$P_{u,i} = \frac{\sum_{\text{all similar items, N}} (s_{i,N} * R_{u,N})}{\sum_{\text{all similar items, N}} (|s_{i,N}|)}$$

Item-Item Recommendation Algorithm



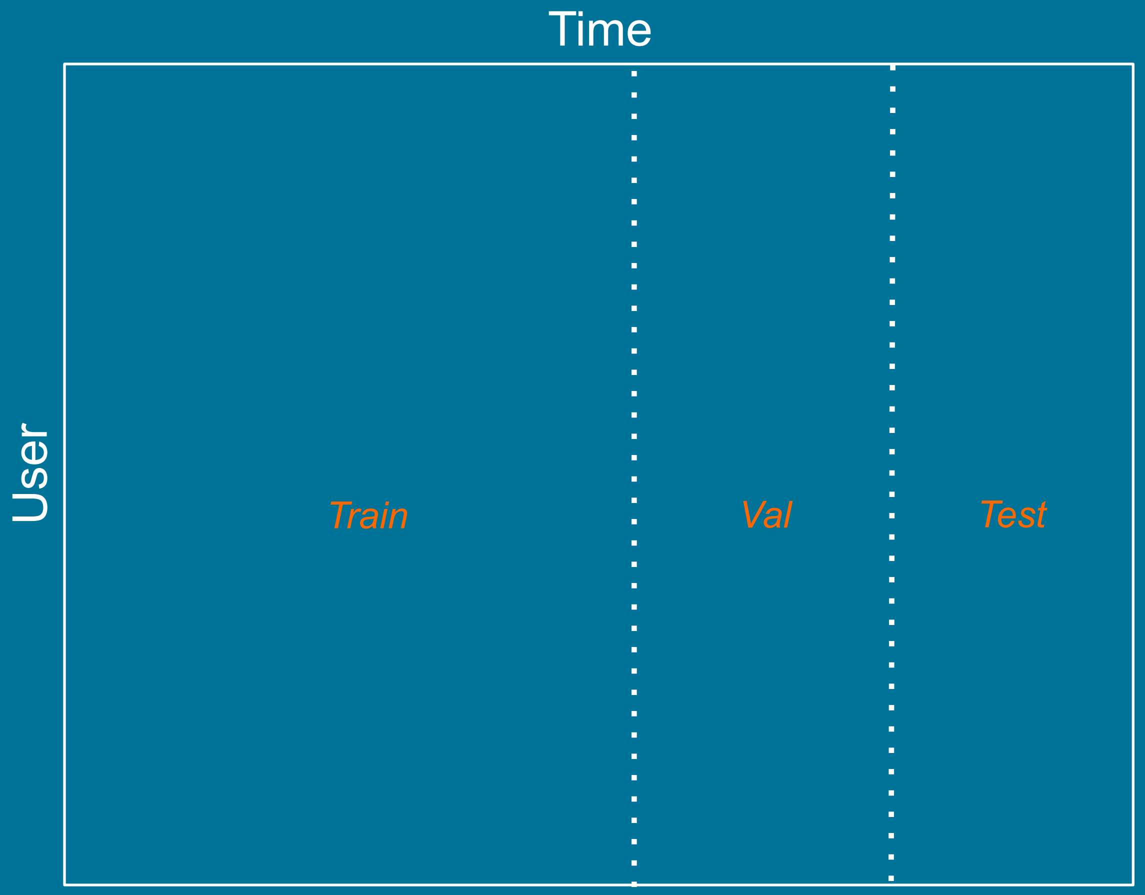


Why Item-Item rather Than User-User

Caching	• Item correlations more stable
Similarity Calculation (Speed)	• Lower algorithmic complexity
Similarity Calculation (Quality)	• User-pairs with only 1 overlapping rating

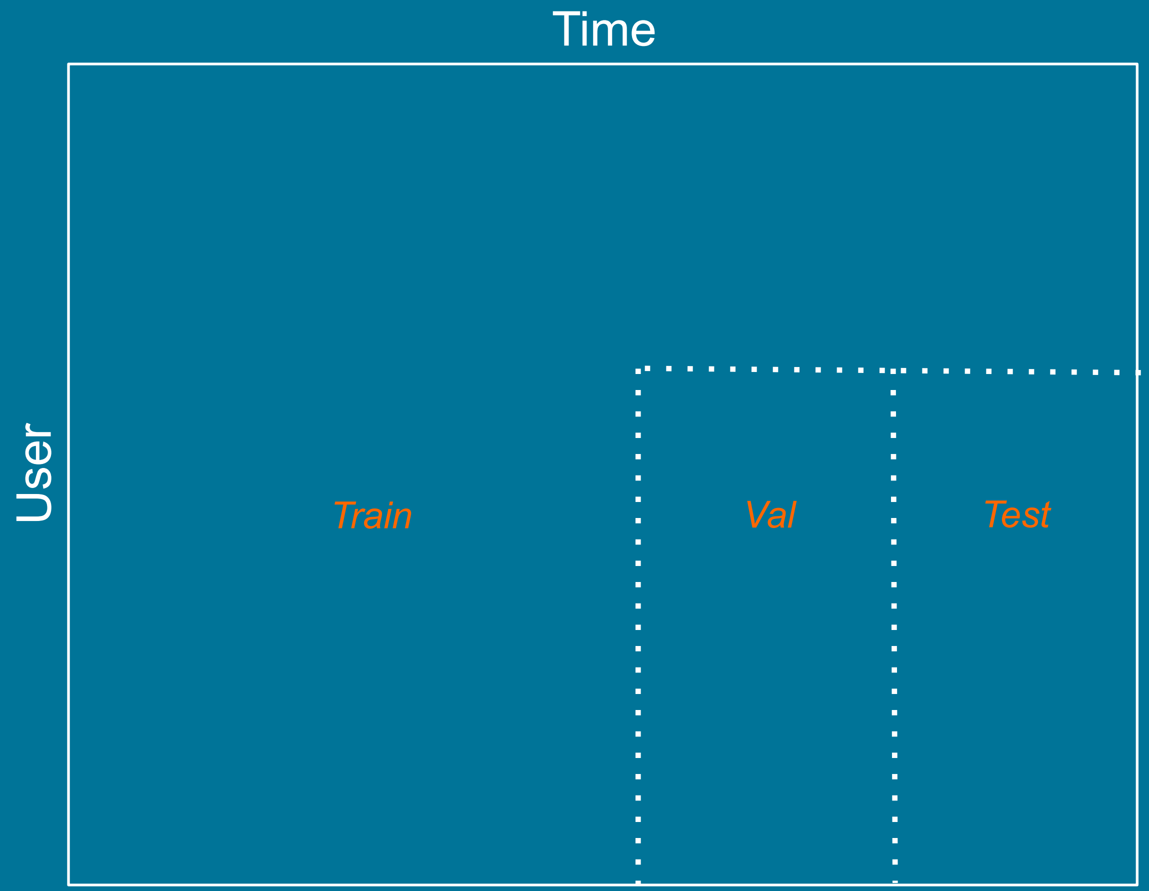
Validation Strategies

Simple Time-Based



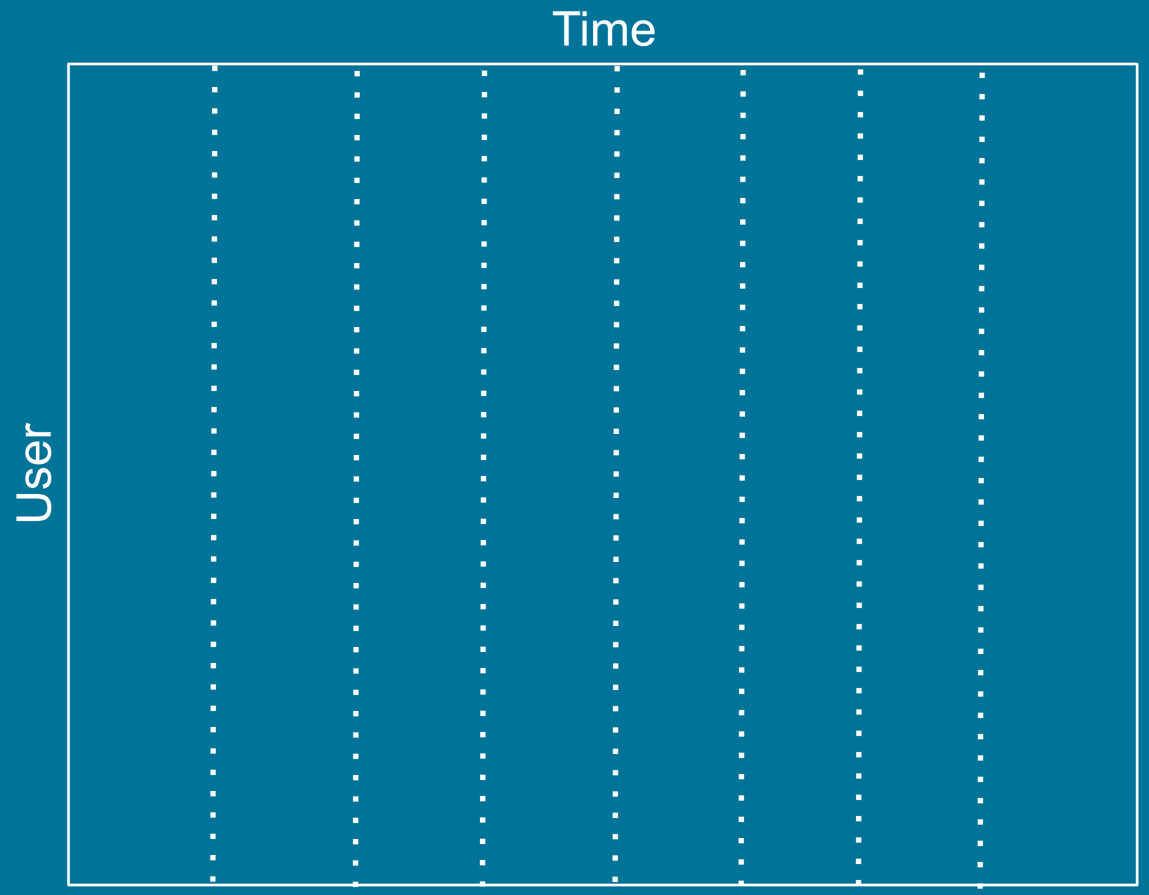
Validation Strategies

More Complex Time-Based

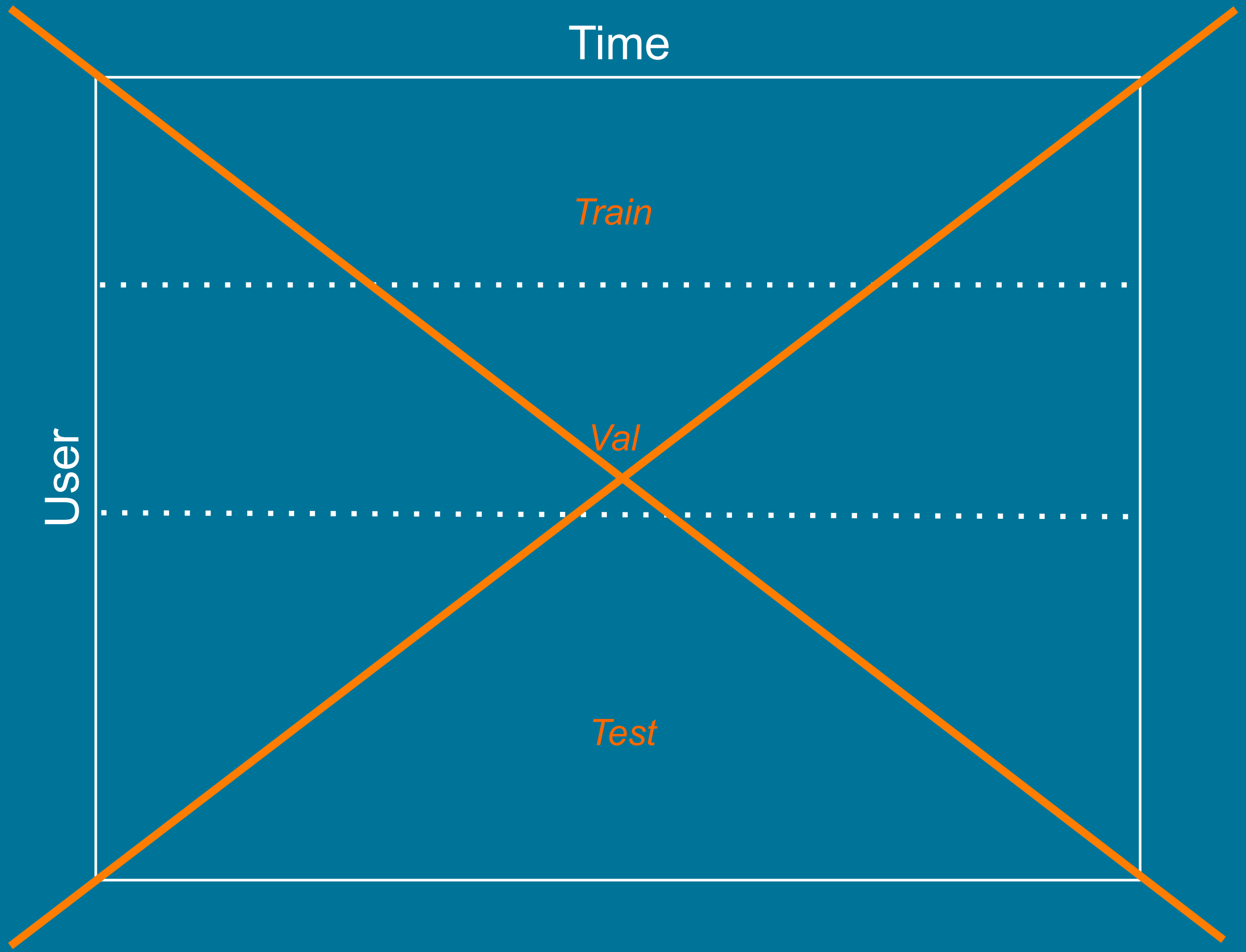


Validation Strategies

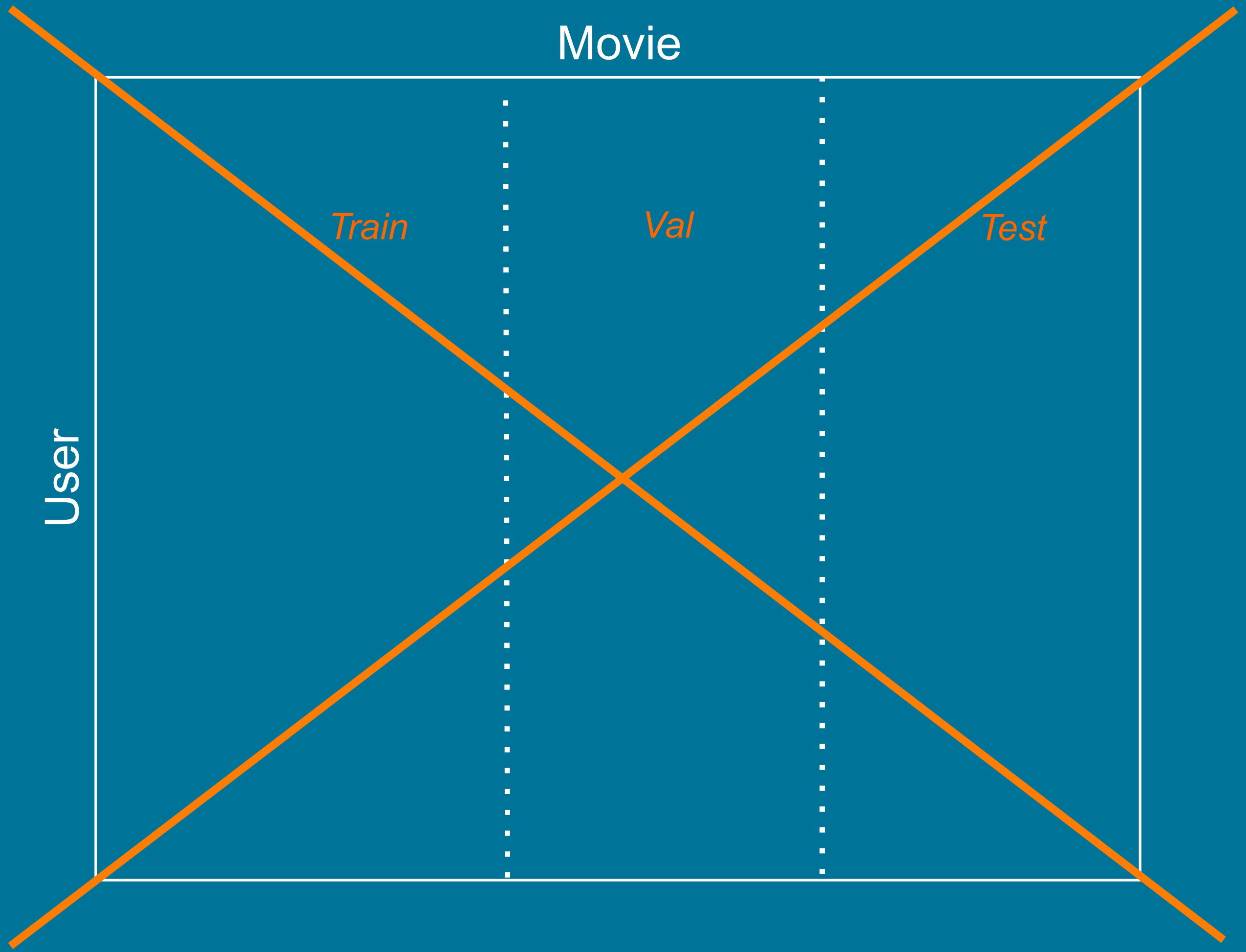
K-Fold



Thought Experiment: Why Doesn't This Work



Another Bad Validation Strategy



Validation Strategies

Purely Random Holdout (Optionally K-fold)