Recommender Systems class

Evaluation and testing

Evaluation measures

Regression

- MSE
- RMSE
- MAE
- MAPE (MRE)
- TRE

Classification

- Sensitivity/Recall/True Positive Rate
- Precision
- Accuracy
- F1 score

Ranking

- HR@n
- NDCG@n
- MAP@n

Regression evaluation measures

- MAE – Mean Absolute Error

$$\frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

- MSE – Mean Squared Error

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

- RMSE – Root Mean Squared Error

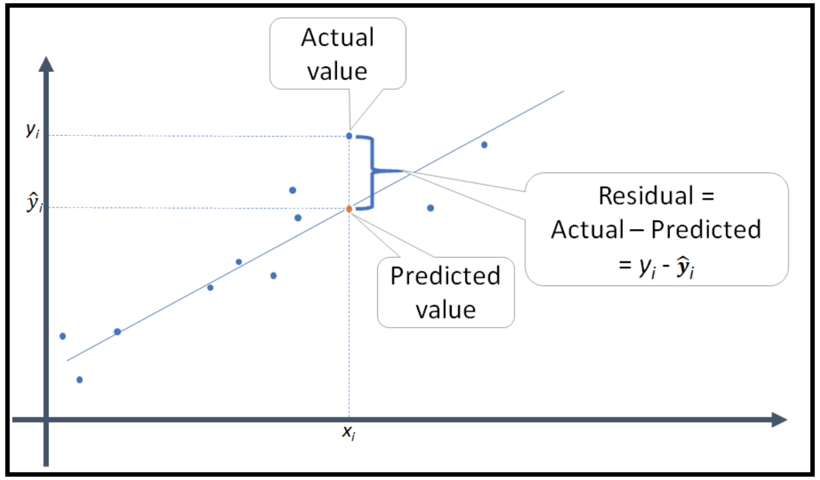
$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

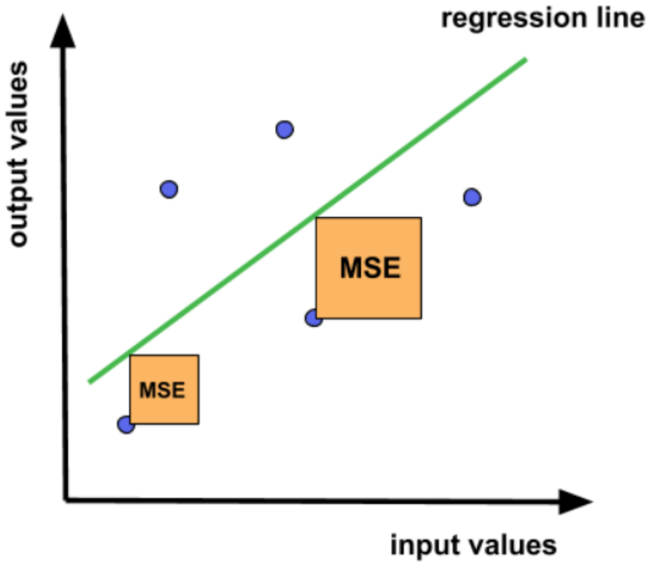
- MAPE (MRE) – Mean Absolute Percentage Error (Mean Relative Error)

$$\frac{1}{n}\sum_{i=1}^{n}\frac{|\hat{y}_i - y_i|}{|y_i|}$$

- TRE – Total Relative Error

$$\frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{\sum_{i=1}^{n}|y_i|}$$





Classification evaluation measures

- Sensitivity/Recall/True Positive Rate

$$\frac{TP}{TP + FN}$$

- Precision

$$\frac{TP}{TP + FP}$$

- Accuracy

$$\frac{TP + TN}{P + N}$$

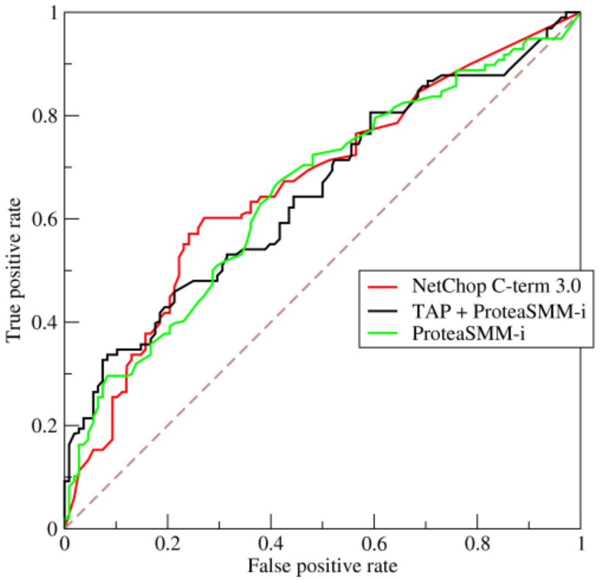
- F1 score

$$\frac{2}{\frac{1}{Sensitivity} + \frac{1}{Precision}}$$

Close to one – good
Close to zero – bad

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

ROC curve



Ranking evaluation measures

- HR@n – Hit Ratio

$$\frac{1}{|U|}\sum_{u\in U}\sum_{k=1}^{n} 1_{D_u}(r_{u,k})$$

- NDCG@n – Normalized Discounted Cumulative Gain

$$\frac{1}{|U|}\sum_{u\in U}\sum_{k=1}^{n} \frac{1_{D_u}(r_{u,k})}{\log_2(1+k)}$$

- MAP@n – Mean Average Precision

$$\frac{1}{|U|}\sum_{u\in U}\frac{1}{|D_u|}\sum_{k=1}^{n} \frac{\mathrm{HR@}k}{k} 1_{D_u}(r_{u,k})$$

Position	Movie	Score
1	Rocky	0.98
2	Interstellar	0.86
3	Shrek	0.83
4	Shawshank Redemption	0.75
5	Lion King	0.69
6	Star Wars	0.61
7	Apocalypto	0.55

$$\mathrm{HR@7} = 1+0+1+1+0+0+1 = 4$$

$$\mathrm{NDCG@7} = \frac{1}{\log_2 2}+\frac{0}{\log_2 3}+\frac{1}{\log_2 4}+\frac{1}{\log_2 5}+\frac{0}{\log_2 6}+\frac{0}{\log_2 7}+\frac{1}{\log_2 2} = 2.26401$$

$$\mathrm{MAP@7} = \left(\frac{1}{1}\cdot 1+\frac{1}{2}\cdot 0+\frac{2}{3}\cdot 1+\frac{3}{4}\cdot 1+\frac{3}{5}\cdot 0+\frac{3}{6}\cdot 0+\frac{4}{7}\cdot 1\right)/4 = 0.7470238095$$

D_u - items user u actually interacted with
$r_{u,k}$ - k-th recommendation for user u

Other evaluation measures

Coverage

- The percentage of all available items in the first n recommendations for all users

Novelty

- Evaluates the likelihood that the user was not aware of the recommended items

Serendipity

- Should measure the surprise effect in recommendations

Diversity

- The first n recommendations should be diverse enough so that if the user does not like the first item he/she might still like the other recommendations

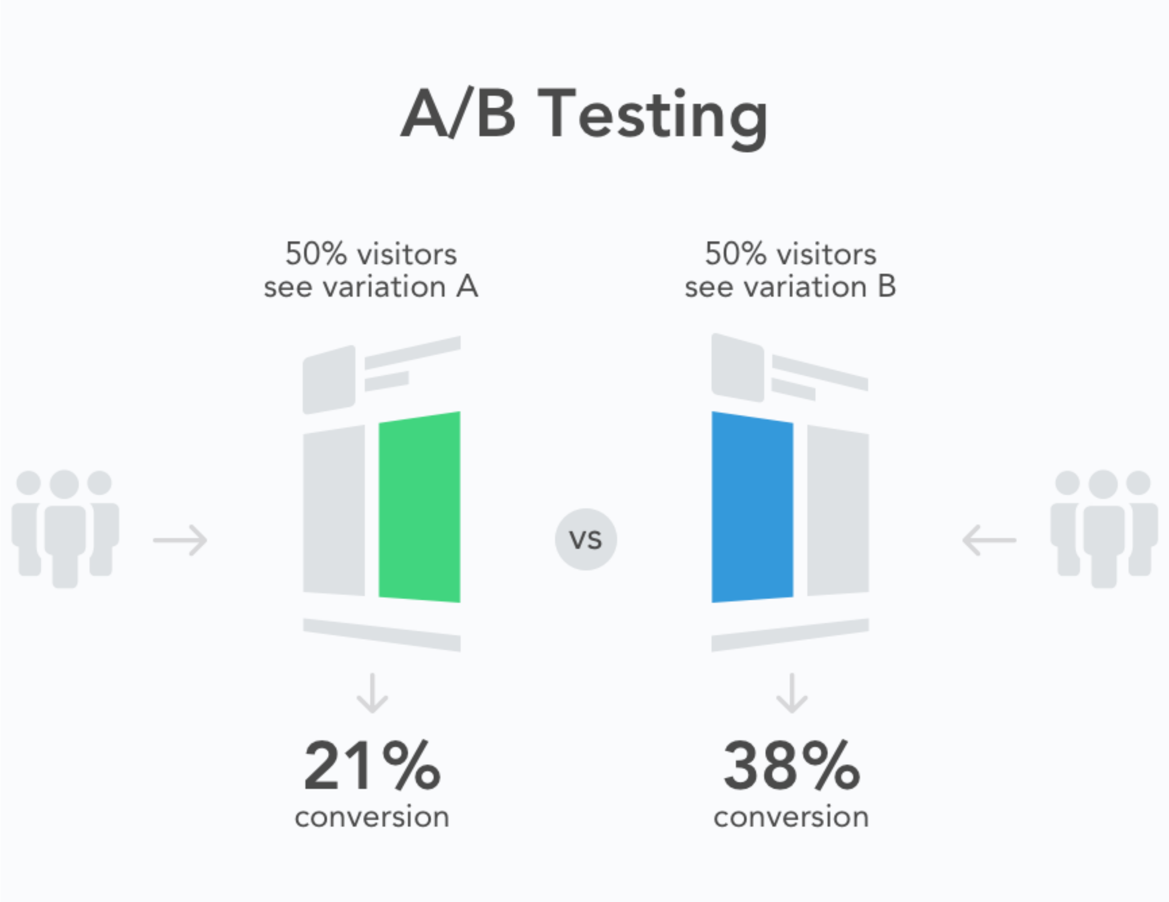
Testing schemes

- Online
  - A/B tests
  - Counterfactuals

- Offline
  - Train-test split
  - Train-validation-test split
  - Cross-validation
  - Leave-one-out
  - Simulation

A/B tests

- Assumes that a model/algorithm is already used in production
- Split all cases when the algorithm is used into two groups: A and B
  (typically 50%-50%)
- After a predefined time gather and compare results of both models/algorithms
- Examples:
  - Two recommenders on YouTube – users are randomly split in half
  - Two trading algorithms – budget is split in half
  - Two website versions are presented to users – cookies are randomly assigned to both groups



Train-test split

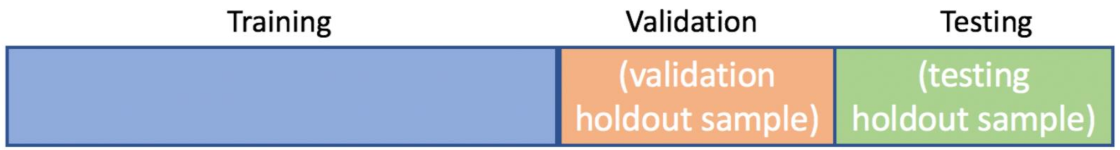
- Divide the dataset into two parts:
  - training set
  - test set
- Train the model on the training set
- Evaluate the model on the test set

+ Good when the dataset is large and training is expensive

- A single dataset split may not properly reflect model's ability to generalize

Train-validation-test split

- Divide the dataset into three parts:
  - training set
  - validation set
  - test set
- Train the model on the training set with many sets of hyperparameters
- Evaluate all trained models on the validation set
- Choose hyperparameters which give the best result
- Evaluate the final model on the test set to check for overfitting



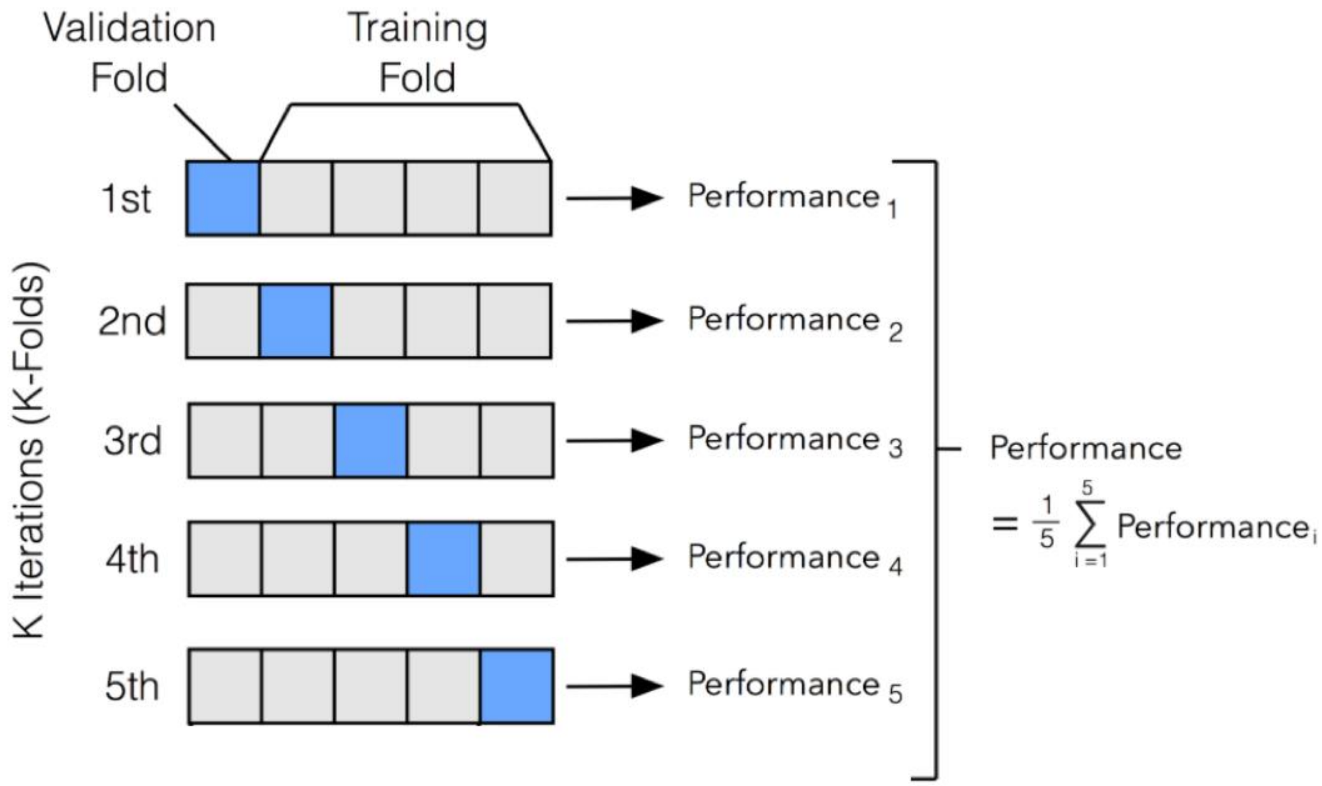
\+ Good when the dataset is large and training is expensive

\+ Allows for a proper hyperparameter tuning

\- A single dataset split may not properly reflect model's ability to generalize

Cross-validation

- Divide the dataset D into K equal-sized parts
- Train and evaluate the model K times in the following way:
    - Choose the i-th dataset part D_i
    - Train the model on $D \backslash D_i$
    - Evaluate the model on D_i
- Gather all K results and aggregate them into a single measure (for instance by taking an average)

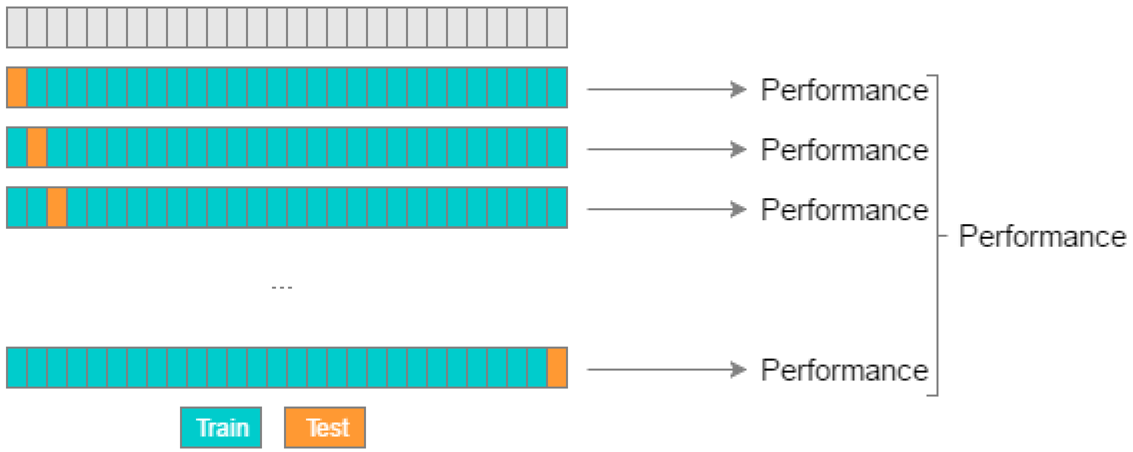


\+ Allows to test the model on the entire dataset removing the risk of choosing an atypical split

\- When K is large and model training is expensive, this method requires a long processing time

Leave-one-out

- Leave-one-out is an extreme case of cross-validation where K is equal to the number of elements in the dataset
- Train and evaluate the model K times in the following way:
  - Choose the i-th element $y_i \in D$
  - Train the model on $D \setminus \{y_i\}$
  - Evaluate the model on y_i
- Gather all results and aggregate them into a single measure



+ Allows to test the model on the entire dataset removing the risk of choosing an atypical split
+ Good when the training set is small
+ Good when the typical proportion of the production training dataset size is large compared to the number of predictions to be made

- For even moderately complex models requires a long processing time

Explicit feedback vs implicit feedback testing

Explicit feedback, e.g. ratings

- Treat the recommender as a typical regressor and use regression measures to evaluate it
- Generate prediction for every pair user-item in the test set
- You can simplify the testing scheme by generating a single prediction at a time
- When serving recommend items with the highest predictions first

Implicit feedback, e.g. binary indicators if there was an interaction or not

- Generate a set of recommendations for each user in the test set
  (typically by assigning a score to every item in the set of items the user have not interacted with)
- Use ranking evaluation measures to take positions into account