Course: Machine Learning

Lab 3: Working with Recommendation Models

Overview

In this lab, you will construct and evaluate a recommender using a sample of user movie rating data. The goal of the recommender is to provide users with recommendations for movies to watch based on their previous movie ratings and movie ratings of other viewers.

What You'll Need

To complete this lab, you will need the following:

- An Azure ML account

- A web browser and Internet connection

Creating a Recommender

In this exercise, you will create a recommender using the Matchbox Recommender provided with Azure ML.

Create a Data Flow to Prepare the Movie Data

The movie data set needs to be transformed so that it is suitable for the recommender algorithm.

1. If you have not already done so, open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.

2. Create a new blank experiment, and give it the title **Movie Recommendation**.

3. Search for the **Movie Ratings** dataset and drag it onto the blank canvas.

4. Search for the **Edit Metadata** module and drag it onto the canvas.

5. Connect the output of the **Movie Ratings** dataset to the **Dataset** input of the **Edit Metadata**.

6. Configure the properties of the **Edit Metadata** to ensure that the **Rating** column is of **Integer** type as required by the **Matchbox Recommender** module:

   - **Column selector:** Rating

   - **Data type:** Integer

   - **Categorical:** Unchanged

   - **Fields:** Unchanged

   - **New column names:** blank

7. Search for the **IMDB Movie Titles** dataset and drag it onto the canvas. This dataset maps numeric movie IDs to human readable movie titles.

8. Search for the **Join Data** module and drag it onto the canvas.

9. Connect the **Results dataset** output of the **Edit Metadata** to the **Dataset1** (left) input of the **Join Data** module.

10. Connect the output of the **IMDB Movie Titles** dataset to the **Dataset2** (right) input of the **Join Data** module.

11. Configure the properties of the **Join Data** module as follows:

    - **Column Selector for L (left):** MovieId

    - **Column Selector for R (right):** Movie ID

    - **Match case:** Checked

    - **Join type:** Inner Join

    - **Keep right key column:** Unchecked

    Your experiment should look like this.
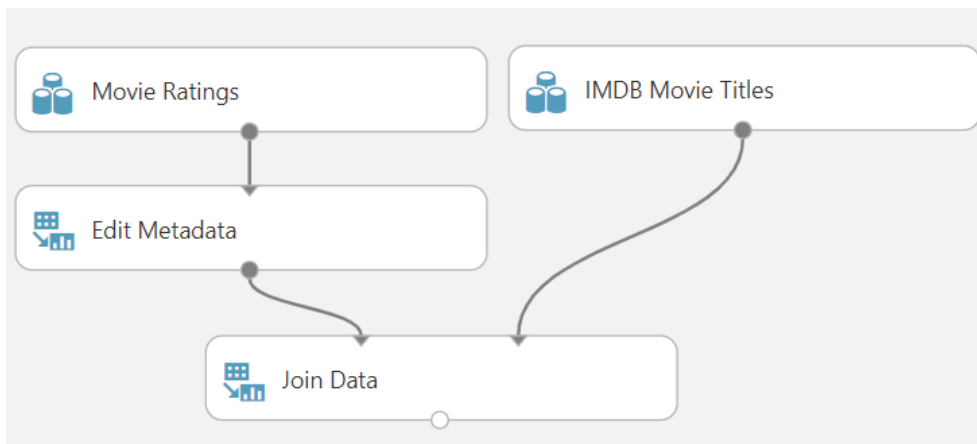
12. Search for the **Select Columns in Dataset** module and drag it onto the canvas.

13. Connect the **Results dataset** output of the **Join Data** module to the **Dataset input** of the **Select Columns in Dataset** module.

14. Configure the **Column Selector** of the **Select Columns in Dataset** module. Select the **Allow duplicates and preserve column order in selection** box, and then select the following columns in the order shown below (the order is important):

    - **UserId**

    - **Movie Name**
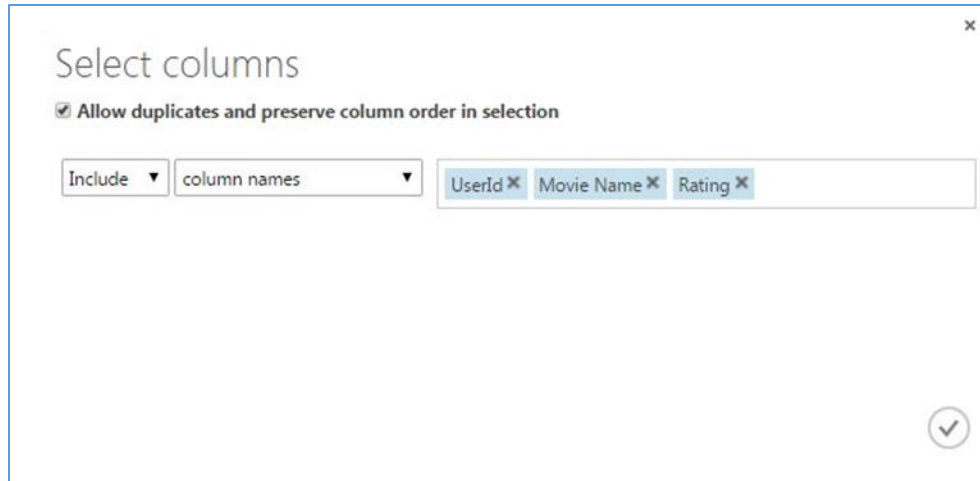
    - **Rating**

Note: Duplicates are common in rating data. A user may rate the same item multiple times. These duplicates should be removed to prevent them from biasing the results of the recommendation calculation.

15. Search for the **Remove Duplicate Rows** module and drag it onto the canvas.

16. Connect the **Results dataset** output of the **Select Columns in Dataset** module to the input of the **Remove Duplicate Rows** module.

17. Configure the properties of the **Remove Duplicate Rows** module as follows:

- **Column Selector**: UserId, Movie Name

- **Retain first duplicate row**: checked

18. Verify that your experiment resembles the following image.
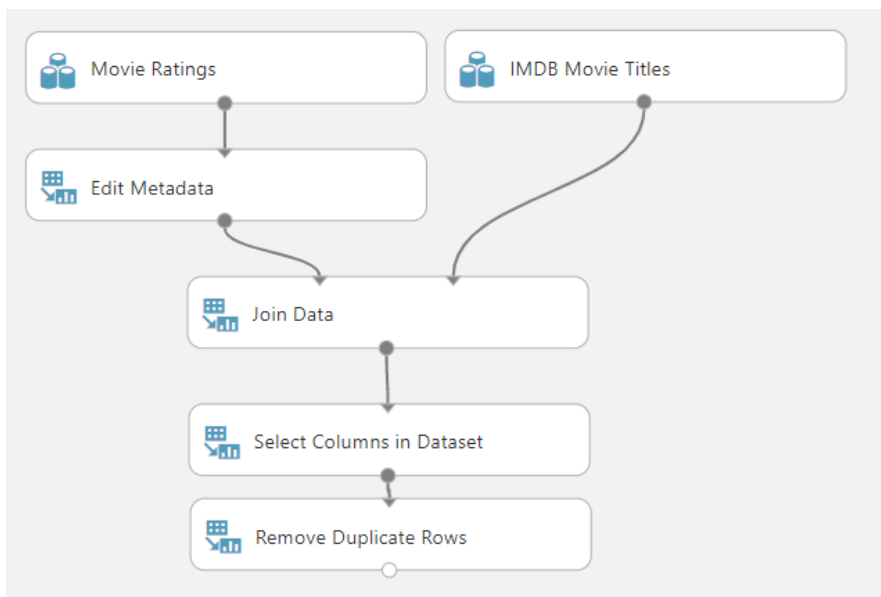
19. Save and run the experiment. When the experiment has finished, visualize the output of the **Remove Duplicate Rows** module. Verify that the results have 227468 rows and 3 columns in the correct order as shown in the following image.

rows	columns		
227468	3		

	UserId	Movie Name	Rating
view as			
	1	The Godfather (1972)	10
	1	Heat (1995)	10
	2	Life of Pi (2012)	8
	2	Dallas Buyers Club (2013)	7
	2	World Z (2013)	8
	2	Lone Survivor (2013)	7
	2	August Osage County (2013)	7
	2	Disconnect (2012)	8
	2	Gravity (2013)	8
	2	Captain Phillips (2013)Biography	8
	2	Intouchables (2011)	8
	2	Her (2013)	10
	2	All Is Lost (2013)	7
	2	12 Years a Slave (2013)	10

20. Close the visualization.

21. Can you draw any conclusions from the results?

Train a Recommender

Now that the data is prepared, you can train a recommender.

1. Search for the **Split Data** module and drag it onto the canvas.

2. Connect the **Results dataset** output of the **Remove Duplicate Rows** module to the input of the **Split Data** module.

3. On the **Properties** pane of the **Split Data** module, configure the properties as follows:

    *   **Splitting mode**: Recommender Split

    *   **Fraction of training-only users**: 0.75

    *   **Fraction of test user ratings for training**: 0.25

    *   **Fraction of cold users**: 0

    *   **Fraction of cold items**: 0

    *   **Fraction of ignored users**: 0

    *   **Fraction of ignored items**: 0

    *   **Remove occasionally produced cold items**: unchecked

- **Random seed for Recommender**: 5432

4. Search for the **Train Matchbox Recommender** module and drag it onto the canvas.

5. Connect the **Results dataset1** (left) output of the **Split Data** module to the **Training dataset of useritem-rating triples** (left) input of the **Train Matchbox Recommender** module.

6. On the **Properties** pane for the **Train Matchbox Recommender** module, configure the properties as follows:

   - **Number of traits:** 20

   - **Number of recommendation algorithm iterations:** 10

   - **Number of training batches:** 4

Evaluating a Recommender

Evaluating recommenders is difficult. First, there are no marked cases with ground truth. A rating is someone's opinion at some point in time in the past. Second, we have no idea why a particular person does or does not like something. A given person may be a friend of a restaurant's proprietor, or they may like any movie starring a certain actor or actress, or they may have some other reason entirely.

To overcome these issues, data scientists evaluate a recommender using several methods and metrics. A recommender that exhibits reasonable performance by several measures is likely to work well in practice. Metrics used to evaluate recommenders include:

- Item Recommendations

- Related Items

- Rating Prediction

- Related Users

Evaluate by Item Recommendation

Item Recommendations compute recommendations and evaluate the results using the user's item ratings. Results are evaluated by averaging normalized discounted cumulative gain (NDCG) over the chosen items. An ideal result has a value of 1.0.

1. Search for the **Score Matchbox Recommender** module and drag it onto the canvas.

2. Connect the **Trained Matchbox recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox recommender** (left) input of the **Score Matchbox Recommender** module.

3. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Dataset to score** (second from left) input to the **Score Matchbox Recommender** module.

4. On the **Properties** pane for the **Score Matchbox Recommender** module, ensure that the following properties are specified:

   - **Recommender prediction kind:** Item Recommendation

   - **Recommended item selection:** From Rated Items (for model evaluation)

- **Maximum number of items to recommend to a user:** 5

- **Minimum size of the recommendation pool for a single user:** 2

5. Search for the **Evaluate Recommender** module and drag it onto the canvas.

6. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Test dataset** (left) input of the **Evaluate Recommender** module.

7. Connect the **Scored dataset** (right) output of the **Score Matchbox Recommender** module to the **Scored dataset** (right) input of the **Evaluate Recommender** module.

8. On the **Properties** pane of the **Evaluate Recommender** module, verify that the properties are set as follows:

- **Minimum number of items that the query user and the related user must have rated in common:** 2

- **Minimum number of users that the query item and the related item must have been rated by in common:** 2

9. Ensure that you experiment from the **Remove Duplicate Rows** module onwards looks like this.
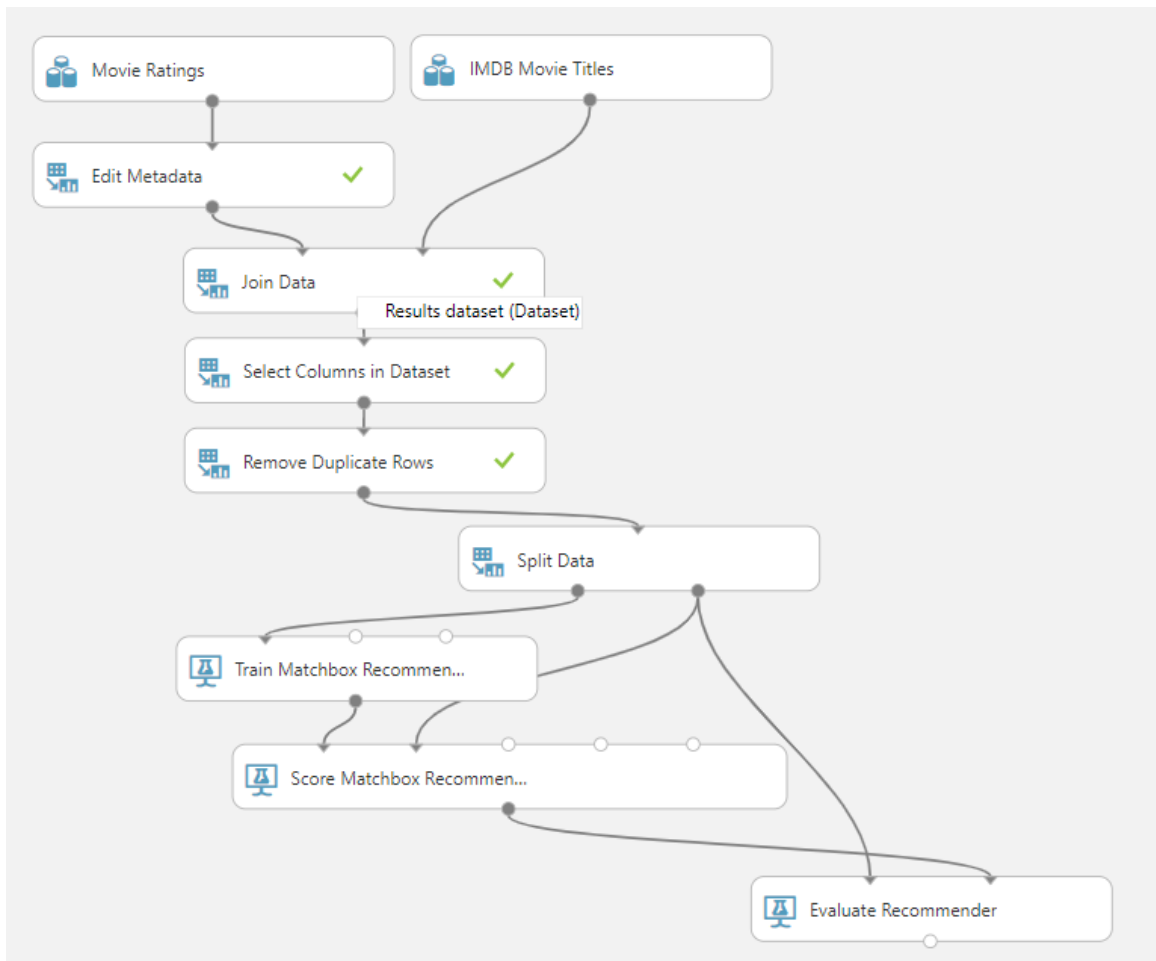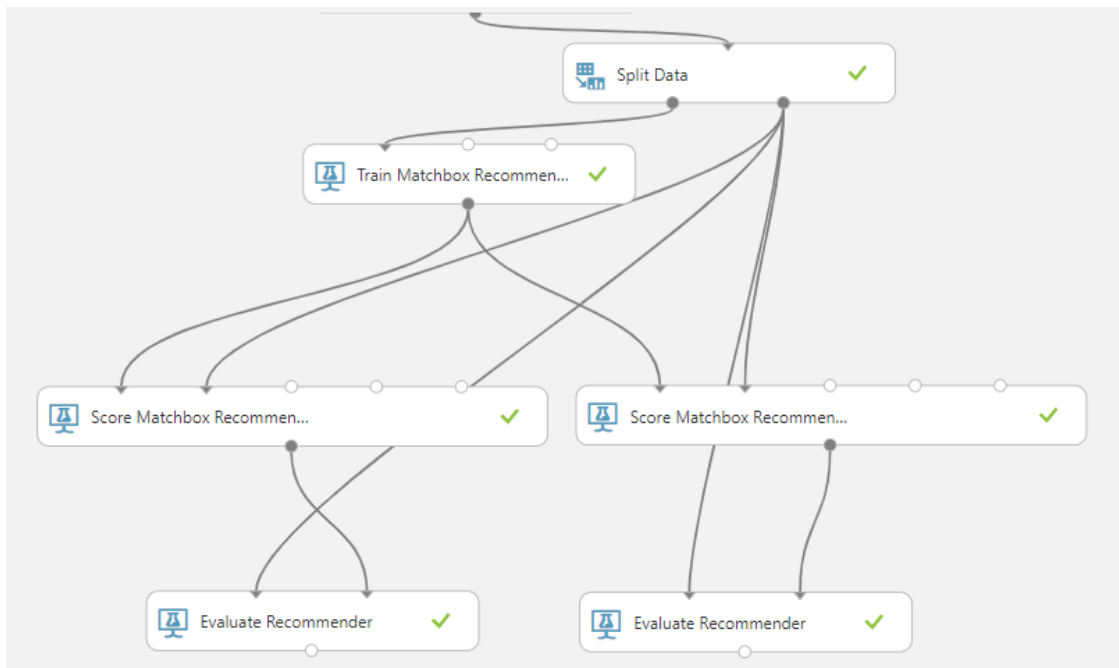

10. Save and run the experiment.

11. When the experiment has finished, visualize the output from the **Evaluate Recommender** module. Note that the **NDCG** is about 0.95. This is an encouraging result, not too far from the ideal.

12. Can you draw any conclusions from the results?

Evaluate by Related Items

For pairs of related items that a group of users has rated, you can predict user rating for one item based on the rating of the other items. Results are evaluated by the similarity of the ratings using both L1 (Manhattan) and L2 (Euclidian) average normalized discounted cumulative gain (NDCG) averaged over all the pairs selected. These metrics are referred to as the L1 similarity NDCG and L2 similarity NDCG. In both cases, an ideal recommender will give a value of 0.0, perfect similarity if the ratings are identical in all cases.

1. Copy the **Score Matchbox Recommender** module and the **Evaluate Recommender** module.

2. Paste these modules onto the canvas and drag them to one side.

3. Connect the **Trained Matchbox recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox recommender** (left most) input of the new **Score Matchbox Recommender** module.

4. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Dataset to score** (second from left) input to the new **Score Matchbox Recommender** module.

5. On the **Properties** pane of the new **Score Matchbox Recommender** module, configure the following properties:

   - **Recommender prediction kind:** Related Items

   - **Related item selection:** From Rated Items (for model evaluation)

   - **Maximum number of related items to find for an item:** 5

   - **Minimum number of users that the query item and the related item must have been rated by in common:** 2

   - **Minimum size of the related item pool for a single user:** 2

6. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Test dataset** (left hand) input of the new **Evaluate Recommender** module.

7. Ensure the **Scored dataset** (right) output of the new **Score Matchbox Recommender** module is connected to the **Scored dataset** (right hand) input of the new **Evaluate Recommender** module.

8. On the **Properties** pane of the new **Evaluate Recommender** module, configure the parameters as follows:

   - **Minimum number of items that the query user and the related user must have rated in common:** 2

   - **Minimum number of users that the query item and the related item must have been rated by in common:** 2

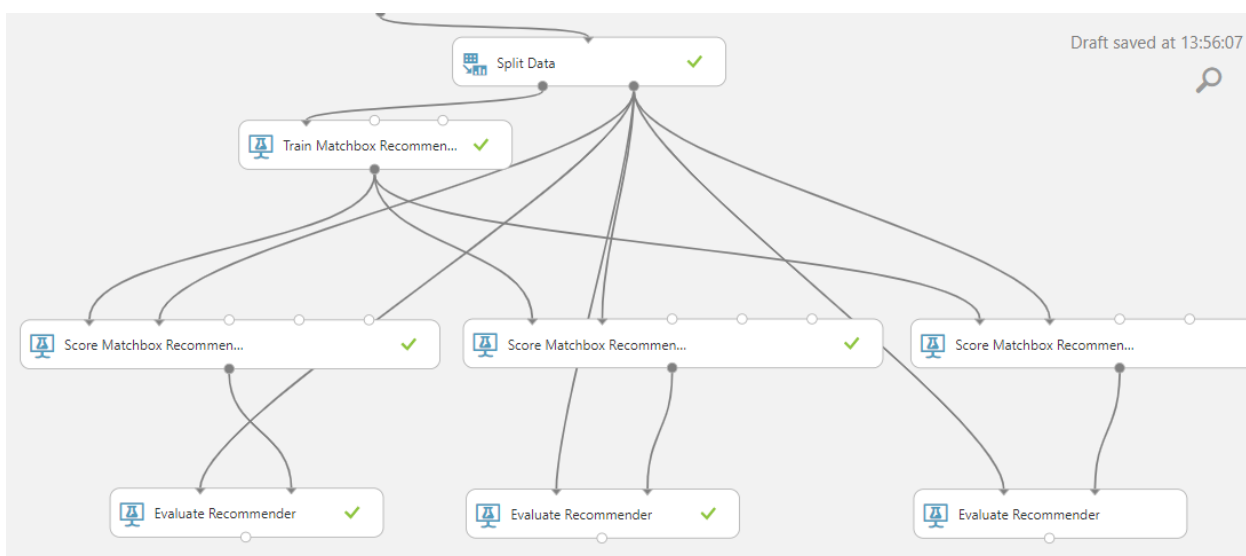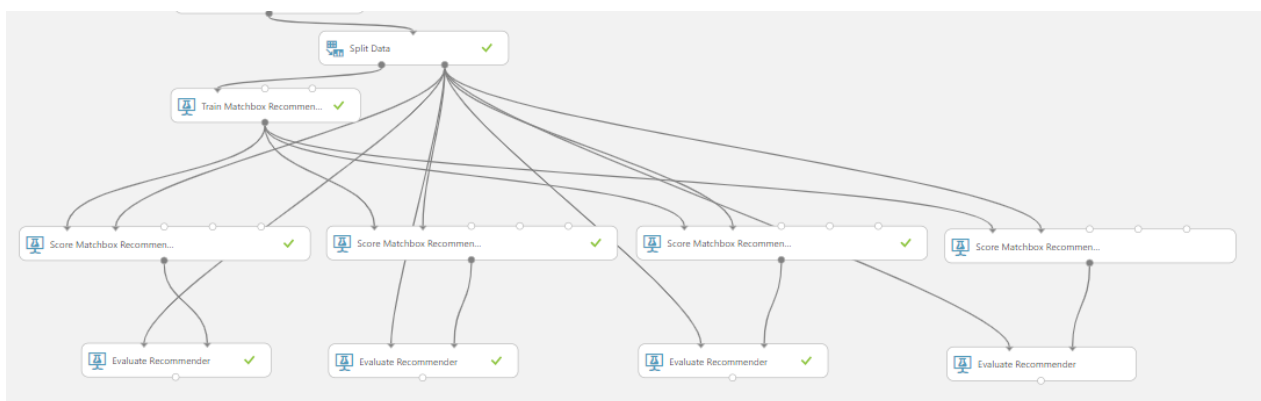9. Ensure that your experiment resembles the following figure.

10. Save and run the experiment.

11. When the experiment has finished, visualize the output form the **Evaluate Recommender** module. Note that the **L1 Sim NDCG** is about 0.71 and the **L2 Sim NDCG** is about 0.70. These values are quite good, considering the ratings are on a 0 to 10 scale.

12. Can you draw any conclusions from the results?

Evaluate by Rating Predictions

Rating Prediction compares predicted rating values to actual rating values using mean absolute error (MAE) and root mean square error (RMSE). Ideal results are 0.0 in both cases, if ratings are predicted perfectly.

1. Copy the newest **Score Matchbox Recommender** module and the newest **Evaluate Recommender** module.

2. Paste these modules onto the canvas and drag them to one side.

3. Connect the **Trained Matchbox recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox recommender** (left most) input of the newest **Score Matchbox Recommender** module.

4. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Dataset to score** (second from left) input to the newest **Score Matchbox Recommender** module.

5. On the **Properties** pane of the newest **Score Matchbox Recommender** module, set the **Recommender prediction kind** property to **Rating Prediction**.

6. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Test dataset** (left hand) input of the newest **Evaluate Recommender** module.

7. Ensure that the **Scored dataset** (right) output of the newest **Score Matchbox Recommender** module is connected to the **Scored dataset** (right hand) input of the newest **Evaluate Recommender** module.

8. On the **Properties** pane of the newest **Evaluate Recommender** module, configure the parameters as follows:

   - **Minimum number of items that the query user and the related user must have rated in common:** 2

   - **Minimum number of users that the query item and the related item must have been rated by in common:** 2

9. Ensure that you experiment resembles the following image.

10. Save and run the experiment.

11. When the experiment has finished, visualize the output form the **Evaluate Recommender** module. Note that the **MAE** is about 1.3 and the **RMSE** is about 1.9. Given that the ratings are on a 0 to 10 scale, these errors are relatively small.

12. Can you draw any conclusions from the results?

Evaluate by Related Users

For pairs of users who have rated the same items, you can predict ratings of one user based on the ratings of the other user. Results are evaluated by the similarity of the ratings using both L1 (Manhattan) and L2 (Euclidian) average normalized discounted cumulative gain (NDCG) averaged over all the pairs selected. These metrics are referred to as the L1 similarity NDCG and L2 similarity NDCG. In both cases, an ideal recommender will give a value of 0.0, perfect similarity if the ratings are identical in all cases.

1. Copy the newest **Score Matchbox Recommender** module and the newest **Evaluate Recommender** module.

2. Paste these modules onto the canvas and drag them to one side.

3. Connect the **Trained Matchbox recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox recommender** (left) input of the newest **Score Matchbox Recommender** module.

4. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Dataset to score** (second from left) input to the newest **Score Matchbox Recommender** module.

5. On the **Properties** pane of the newest **Score Matchbox Recommender** module, configure the following properties:

   - **Recommender prediction kind:** Related Users

   - **Related user selection:** From Users That Rated Items (for model evaluation)

   - **Maximum number of related Users to find for a User:** 5

   - **Minimum number of items that the query user and the related user must have rated in common:** 2

   - **Minimum size of the related user pool for a single user:** 2

6. Connect the **Results dataset2** (right) output of the **Split Data** module to the **Test dataset** (left hand) input of the newest **Evaluate Recommender** module.

7. Ensure the **Scored dataset** (right) output of the newest **Score Matchbox Recommender** module is connected to the **Scored dataset** (right) input of the newest **Evaluate Recommender** module.

8. On the **Properties** pane of the newest **Evaluate Recommender** module, configure the parameters as follows:

   - **Minimum number of items that the query user and the related user must have rated in common:** 2

   - **Minimum number of users that the query item and the related item must have been rated by in common:** 2

9. Ensure that your experiment resembles the following image.

10. Save and run the experiment.

11. When the experiment has finished, visualize the output form the **Evaluate Recommender** module. Note that the **L1 Sim NDCG** is about 0.58 and the **L1 Sim NDCG** is about 0.56. These values are quite good, considering the ratings are on a 0 to 10 scale.

12. Can you draw any conclusions from the results?

Computing Movie Recommendations

You have constructed the recommender and tested using four different metrics. Overall, the results are good. You will now compute a set of movie recommendations for a set of users.

Compute Recommendations

1. Search for the **Remove Duplicate Rows** module, and drag a new one onto the canvas.

2. Connect the output of the existing **Remove Duplicate Rows** module to the input of the new **Remove Duplicate Rows** module.

3. In the **Properties** pane of the new **Remove Duplicate Rows** module, set the following:

    - **Column selector:** UserId

    - **Retain first duplicate row:** checked

4. Search for the **Partition and Sample** module, and drag it onto the canvas.

5. Connect the output of the new **Remove Duplicate Rows** module to the input of the **Partition and Sample** module.

6. On the **Properties** pane of the **Partition and Sample** module, set the following parameters:

    - **Partition or sample mode:** Head

    - **Number of rows to select:** 100

7. Search for the **Select Columns in Dataset** module, and drag it onto the canvas.

8. Launch the **column selector** of the **Select Columns in Dataset** module and select only the **UserId** columns.

9. Search for the **Score Matchbox Recommender** module, and drag a new one onto the canvas.

10. Connect the **Trained Matchbox Recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox Recommender** input of the **Score Matchbox Recommender** module.

11. Connect the output of the newest **Select Columns in Dataset** module to the **Dataset to Score** (second from the left) input of the **Score Matchbox Recommender** module.

12. On the **Properties** pane of the **Score Matchbox Recommender** module, set the following parameters:

    - **Recommender prediction kind:** Item Recommendation

    - **Recommended item selection:** From All Items

    - **Maximum number of items to recommend to a user:** 3

13. Ensure that the new parts of your experiment resemble the lower-right part of this diagram.
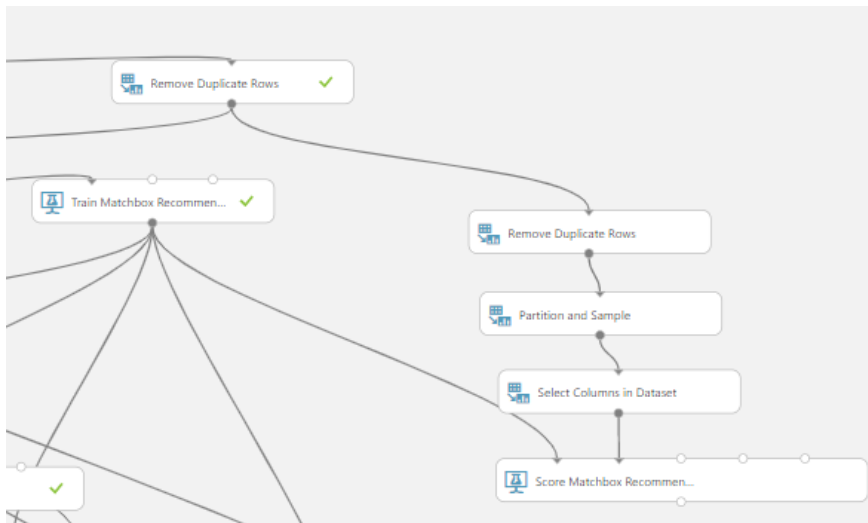
14. Save and run the experiment.

15. When the experiment has finished running, visualize the output of the **Score Matchbox Recommender** module. Examine the output.


There are 100 rows, one for each of the users. Each row contains three movie recommendations for each user.

16. Can you draw any conclusions from the results?

Summary

In this lab, you have constructed and evaluated a movie recommender. Specifically, you:

- Evaluated the movie recommender using four different metrics.

- Determined that the performance of the recommender was reasonably good based on these metrics.

- Used the recommender to create movie recommendations for 100 users.