Data Science + Machine Learning

Building a Machine Learning model and using it for prediction

Taught by:

Paraskevi Zerva
Solutions Architect

Learning Objectives

- The Machine Learning Model Development Life Cycle
- Stardog's Machine Learning Features and Services for statistical predictions
- How to prepare training data for a Machine Learning model using Stardog
- Steps to train and evaluate a supervised-learning classification model and a Machine Learning regression model using Stardog
- How to build a similarity model using Stardog
- How to tune and optimize hyper-parameters to improve model performance
- Steps to prepare and develop your data in preparation for your model training

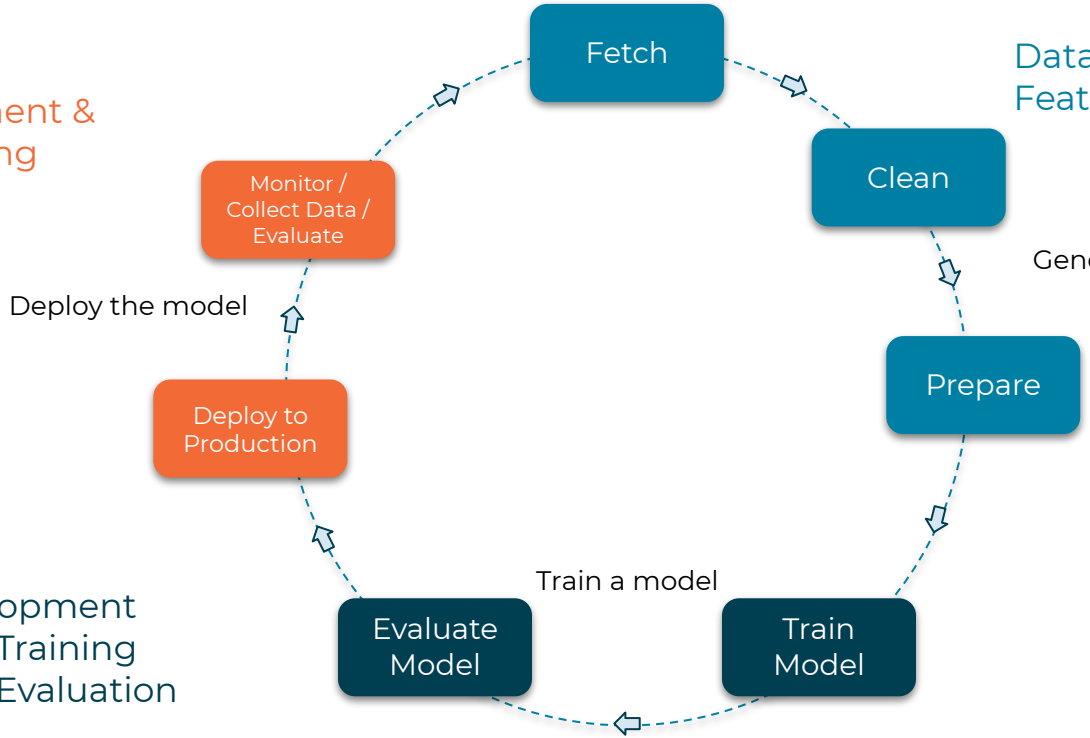
Machine Learning

STARDOG ACADEMY

Machine Learning

- Ability to learn without being explicitly programmed
- ML is based on statistical inference
- Adding statistical to the logical inference that Stardog performs
- Stardog focuses on predictive analytics
  - Predict nodes and edges in a (knowledge) graph
  - Extract patterns and make predictions over those patterns

ML Model Development Life Cycle



Data Exploration & Feature Selection/Engineering

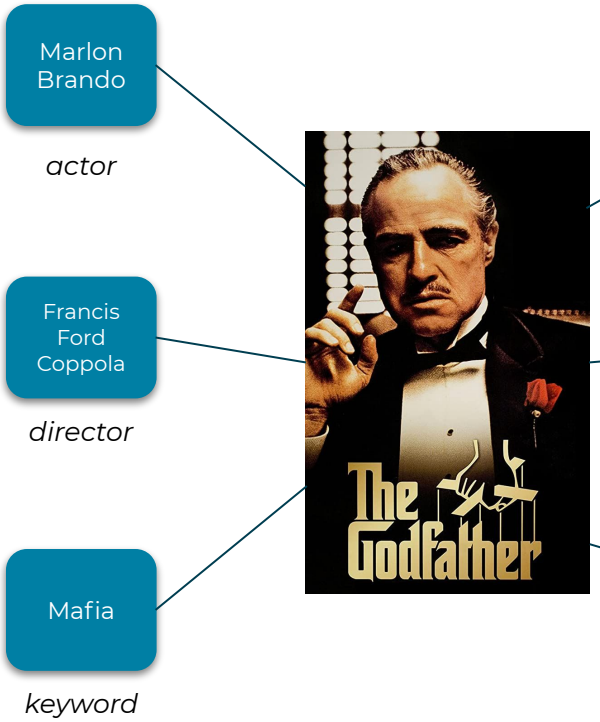
Model Deployment & Model Monitoring

Model Development
- Model Training
- Model Evaluation

Stardog Services in ML Model Development

- Explore and fetch data for preparation of training data
- Model Data & Feature Selection (Feature Engineering)
- Train (Learn) a Model
- Make Predictions
- Evaluate and Assess Model Quality

Stardog Machine Learning Services



Regression
Predicting numerical values

Classification
Predicting categorical values

Similarity
Predict similar nodes

ML Stardog Services Implementation

- Fixed set of algorithms and set of parameters
- Regression & Clustering
  - Classification (Binary & Multi-class classification)

    Vowpal Wabbit : an extremely efficient and scalable ML library

- Similarity Model

    Approximate Nearest Neighbor Search index based on Cluster Pruning technique

ML Model Approaches

- Supervised Training
- Unsupervised Training
- Enabling Machine Learning over Reasoning

Stardog Services by Example

A Supervised Learning Example

Supervised Learning

- Infer a function from labeled training data
- Training data is a set of instance examples
- Each example is a pair
  - An input object (typically a vector)
  - A desired output value
- Learn a model that can be used to make predictions of that data

Supervised Learning Model Lifecycle Steps

I. Prepare training data
II. Determine feature selection as input to the model
III. Train a model
IV. Evaluate the accuracy of the learned model

Walk-through Examples

- Use Movie Data that can be found under link
- Showcase a Classification Model Example
- Showcase a Regression Model Example

A Classification Model Example

STARDOG ACADEMY

Predict Movie Genres

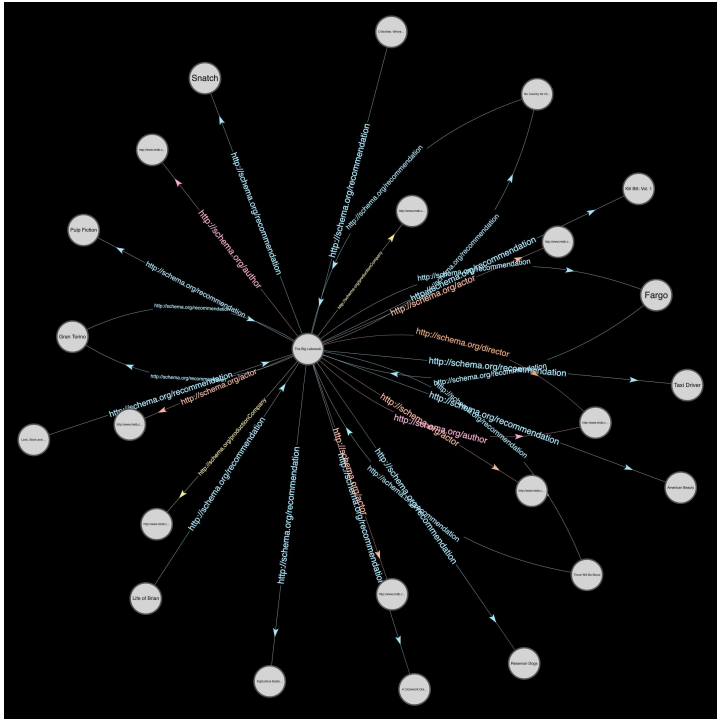
- Movies are linked to an enumeration of **genres**
- For a completely new movie, based on its **properties**, *determine what genre it belongs to*
- Classification problem

I. Prepare Training Data (IMDB movies)

- The dataset contains info about 6730 movies with varied degrees of detail
- A typical movie, identified by its IMDB ID, contains the following properties

```
t:tt0118715 :actor n:?ACTORnm0000422 , n:nm0000114 , n:nm0000313 , n:nm0000194 ;
    :description "\"The Dude\" Lebowski, mistaken for a millionaire Lebowski, ...." ;
    rdfs:label "The Big Lebowski" ;
    :boxOffice 17439163 ;
    :author n:nm0001053 , n:nm0001054 ;
    :director n:nm0001054 ;
    :genre "Crime" , "Comedy" , "Mystery" ;
    :contentRating "R" ;
    :copyrightYear 1998 ;
    :rating "8.2"^^xsd:float ;
    :productionCompany c:co0057311 , c:co0030612 ;
    :keyword "death" , "drug" , "nihilism" , "rug" , "white russian" ;
    :language "German" , "English" , "Spanish" , "Hebrew" ;
    :storyline "When \"The Dude\" Lebowski is mistaken for a millionaire Lebowski ... " ;
    :recommendation ... t:tt0075314 , t:tt0477348 , t:tt0116282 ;
    :metaCritic 69 .
```

Knowledge Graph for IMDB Movies



II. Feature Selection as ML Model Input

- Model input is a function (array of values)
- Treat SELECT query as such function for feature engineering
- Features represent (columns in relational db or) properties in a graph
- Results of a SELECT query to feed this as input back to the ML algorithm

III. Train Model

- Before Stardog can perform predictions we need to define what we need to predict: this task is called model training
- You provide the data and a target, and Stardog learns a model that can be used to predict the value of the target given some other, probably unseen, data
- Training a model in Stardog can be expressed in SPARQL using INSERT clause
- WHERE clause selects the data we are interested in (features)
- Special graph spa:model, is used to specify the parameters of the training

Specify Type of ML

- Classification

  spa:ClassificationModel : if we are interested in predicting a categorical value that has a limited set of possible values (e.g., genre of a movie)

- Regression

  spa:RegressionModel : if we predict a numerical value that can naturally have an unlimited set of values (e.g., box office of a movie)

- Similarity

  spa:SimilarityModel : if we want to predict the degree of similarity between two objects (e.g., most similar movies)

Train Genre Prediction Model

- Our model will be trained to predict the value of ?genre (spa:predict)
- Based on the values of ?director , ?year, ?studio (spa:arguments) that will be used as input features
- Type of learning specified as a classification model (spa:ClassificationModel)

```
PREFIX spa: <tag:stardog:api:analytics:>

INSERT {
   GRAPH spa:model {
       :GenreModel a spa:ClassificationModel ;
                   spa:arguments (?director ?year ?studio) ;
                   spa:predict ?genre .
   }
}
WHERE {
   ?movie :director ?director ;
          :copyrightYear ?year ;
          :productionCompany ?studio ;
          :genre ?genre . }
```

Predict Movie Genre

- Now that we have trained a model, we can use it for prediction as part of query answering
- We select a movie's properties and use as arguments to the model Stardog learned
- predictedGenre variable is predicted by the model, based on the values of the arguments
- Query answering proceeds as if the predicted value were present in the graph

```
SELECT ?predictedGenre {
  GRAPH spa:model {
      :GenreModel spa:arguments (?director ?year ?studio) ;
                  spa:predict ?predictedGenre .
  }
  ?movie rdfs:label "The Godfather" ;
         :director ?director ;
         :copyrightYear ?year }
```

```
+----------------+
| predictedGenre |
+----------------+
| "Drama"        |
+----------------+
```

IV. Evaluate Accuracy of the Model

Percentage of times correct prediction is made

```
SUM(IF(?a = ?b, 1, 0)) / COUNT(*)
```

```
SELECT ?accuracy {
    graph spa:model {
        :GenreModel spa:evaluationScore ?accuracy
    }
}
```

```
+----------------------+
|       accuracy       |
+----------------------+
| 2.965268844867507E-1 |
+----------------------+
```

Prediction Confidence

- Besides predicting the most probable value for a property, you will be interested to know the confidence of that prediction.
- By providing the spa:confidence property, you can get confidence levels for all the possible predictions.
- These values can be interpreted as the probability of the given prediction being the correct one and are useful for tasks like ranking and multi-label classification.

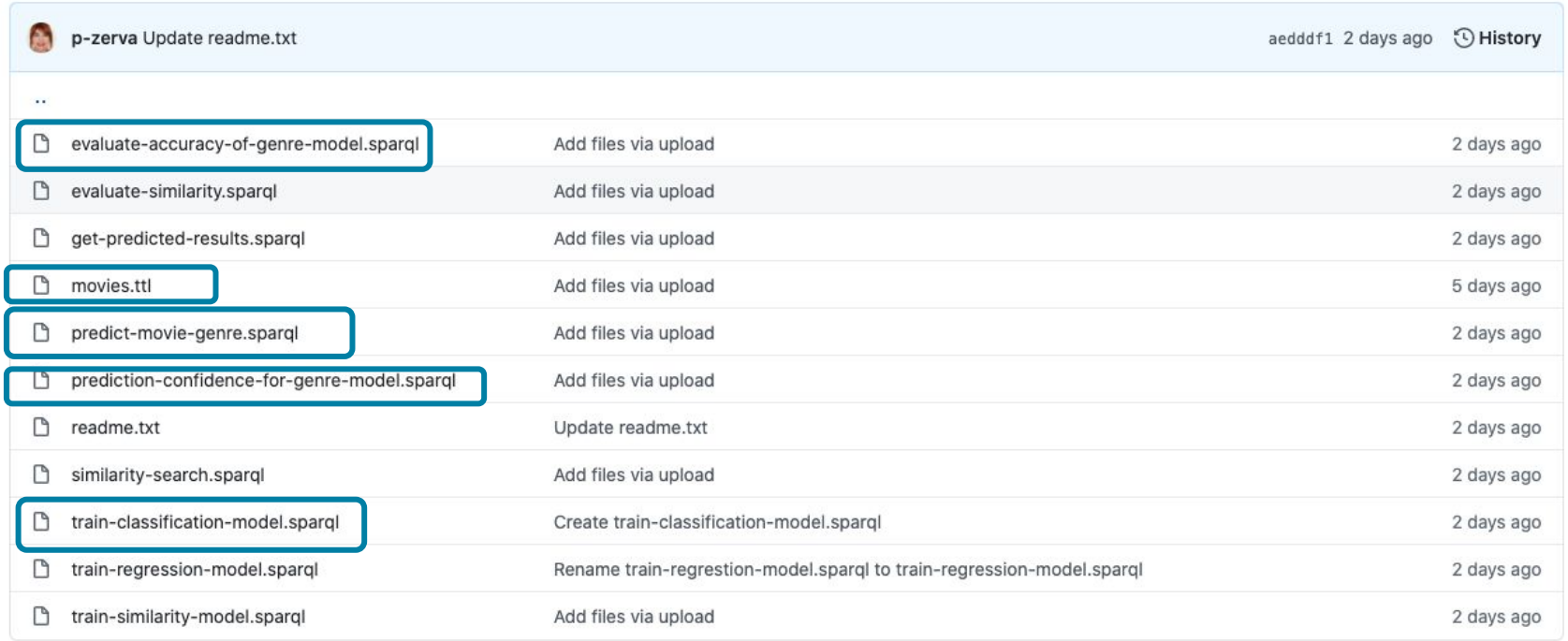
Retrieve Prediction Confidence for Genre Model

```
prefix spa: <tag:stardog:api:analytics:>

SELECT ?predictedGenre ?confidence {
  graph spa:model {
      :GenreModel spa:arguments (?director ?year ?studio) ;
                  spa:predict ?predictedGenre ;
                  spa:confidence ?confidence
  }

  ?movie rdfs:label "The Godfather" ;
         :director ?director ;
         :copyrightYear ?year
}
ORDER BY DESC(?confidence)
LIMIT 5
```

```
+----------------+-----------------------+
| predictedGenre |      confidence       |
+----------------+-----------------------+
| "Drama"        | 2.9611116647720337E-1 |
| "Crime"        | 1.123814657330513E-1  |
| "Fantasy"      | 1.1185823380947113E-1 |
+----------------+-----------------------+

Query returned 3 results in 00:00:00.095
```

Demo

Access ML Stardog Repository



p-zerva Update readme.txt		aedddf1 2 days ago History
..		
evaluate-accuracy-of-genre-model.sparql	Add files via upload	2 days ago
evaluate-similarity.sparql	Add files via upload	2 days ago
get-predicted-results.sparql	Add files via upload	2 days ago
movies.ttl	Add files via upload	5 days ago
predict-movie-genre.sparql	Add files via upload	2 days ago
prediction-confidence-for-genre-model.sparql	Add files via upload	2 days ago
readme.txt	Update readme.txt	2 days ago
similarity-search.sparql	Add files via upload	2 days ago
train-classification-model.sparql	Create train-classification-model.sparql	2 days ago
train-regression-model.sparql	Rename train-regrestion-model.sparql to train-regression-model.sparql	2 days ago
train-similarity-model.sparql	Add files via upload	2 days ago

[Link to Stardog Learning Repository]

A Regression Model Example

STARDOG ACADEMY

Predict Average User Rating

- Average user rating given by IMDB users
- Rating is a number between 1 and 10
- Regression problem

I. Prepare Training Data & II. Feature Selection

```
........
WHERE {
    SELECT
    (spa:set(?genre) as ?genres)
    ?contentRating
    ?storyline
    ?metaCritic
    ?rating
    {
        ?movie  :rating ?rating ;
                :genre ?genre ;
                :contentRating ?contentRating ;
                :storyline ?storyline .
         OPTIONAL {
            ?movie  :metaCritic ?metaCritic .
        }
     }
     GROUP BY ?movie ?rating ?contentRating ?storyline ?metaCritic }
```

III. Train Model

```
INSERT {
    graph spa:model {
        :r1 a spa:RegressionModel ;
            spa:arguments (?genres ?contentRating ?storyline ?metaCritic) ;
            spa:predict ?rating ;
            spa:crossValidation 100 ;
            spa:evaluationMetric spa:mae ;
            spa:overwrite True .
    }
}
       WHERE {
           SELECT
           (spa:set(?genre) as ?genres)
           ?contentRating
           ?storyline
           ?metaCritic
           ?rating
           { ?movie  :rating ?rating ;
                     :genre ?genre ;
                     :contentRating ?contentRating ;
                     :storyline ?storyline .
              OPTIONAL {
                  ?movie  :metaCritic ?metaCritic .
              }
            }
            GROUP BY ?movie ?rating ?contentRating ?storyline ?metaCritic }
```

Set Operator 1/2

- Due to the nature of relational query languages like SPARQL, results are returned for all the combinations between the values of the selected variables.
- In order to properly model relational domains like this, we introduced a special aggregate operator, set.
- Used in conjunction with GROUP BY, we can easily model this kind of data as a single result per individual.

Set Operator 2/2

```
........
WHERE {
    SELECT
    (spa:set(?genre) as ?genres)
    ?contentRating
    ?storyline
    ?metaCritic
    ?rating
    {
        ?movie  :rating ?rating ;
                :genre ?genre ;
                :contentRating ?contentRating ;
                :storyline ?storyline .
         OPTIONAL {
            ?movie  :metaCritic ?metaCritic .
        }
     }
     GROUP BY ?movie ?rating ?contentRating ?storyline ?metaCritic }
```

III. Train Model with Validation

```
prefix agg: <urn:aggregate>
prefix spa: <tag:stardog:api:analytics:>

INSERT {
    graph spa:model {
        :RatingModel a spa:RegressionModel ;
            spa:arguments (?genres ?contentRating
?storyline ?metaCritic) ;
            spa:predict ?rating ;
            spa:crossValidation 100 ;
            spa:evaluationMetric spa:mae .
    }
}
....
```

The default automatic evaluation technique of measuring the accuracy of the model on the same data as that of the training might be prone to overfitting.

The most accurate measure we can have is testing on data that the model has never seen before.

We provide a spa:crossValidation property, which will automatically apply K-Fold cross validation on the training data, with the number of folds given as an argument.

In this case, we will be using 100-fold cross validation, using the mean absolute error as score.

Predict Ratings

```
./stardog query movies predicted_ratings.sparql
+----------------------------------------+-----------------+------------------------+
|                 title                  |      rating     |    predictedRating     |
+----------------------------------------+-----------------+------------------------+
| "Independence Day"                     | "6.9"^^xsd:float | "6.866226"^^xsd:float  |
| "Raging Bull"                          | "8.3"^^xsd:float | "7.9724197"^^xsd:float |
| "Star Trek V: The Final Frontier"      | "5.4"^^xsd:float | "5.411457"^^xsd:float  |
| "The Handmaiden"                       | "8.1"^^xsd:float | "7.9376"^^xsd:float    |
| "Harry Potter and the Sorcerer's Stone" | "7.5"^^xsd:float | "7.239612"^^xsd:float  |
| "The Princess Bride"                   | "8.1"^^xsd:float | "7.8432794"^^xsd:float |
| "X-Men: First Class"                   | "7.8"^^xsd:float | "7.4362893"^^xsd:float |
| "Central Intelligence"                 | "6.4"^^xsd:float | "6.092653"^^xsd:float  |
| "L.A. Confidential"                    | "8.3"^^xsd:float | "7.9753857"^^xsd:float |
| "Phantasm II"                          | "6.5"^^xsd:float | "6.4561276"^^xsd:float |
| "The Godfather"                        | "9.2"^^xsd:float | "8.723323"^^xsd:float  |
| "Chinatown"                            | "8.2"^^xsd:float | "7.79008"^^xsd:float   |
| "The Conjuring 2"                      | "7.5"^^xsd:float | "7.10889"^^xsd:float   |
| "The Curse of the Jade Scorpion"       | "6.8"^^xsd:float | "6.4890537"^^xsd:float |
...
```

IV. Evaluate a Regression Model

Three different measures:

- Mean absolute error: how far away is the prediction from the real target number

  spa:mae(?originalValue, ?predictedValue)

- Mean square error: how much is the squared difference between prediction and the target number

  spa:mse(?originalValue, ?predictedValue)

- Root mean square error: the square root of the mean square error:

  spa:rmse(?originalValue, ?predictedValue)

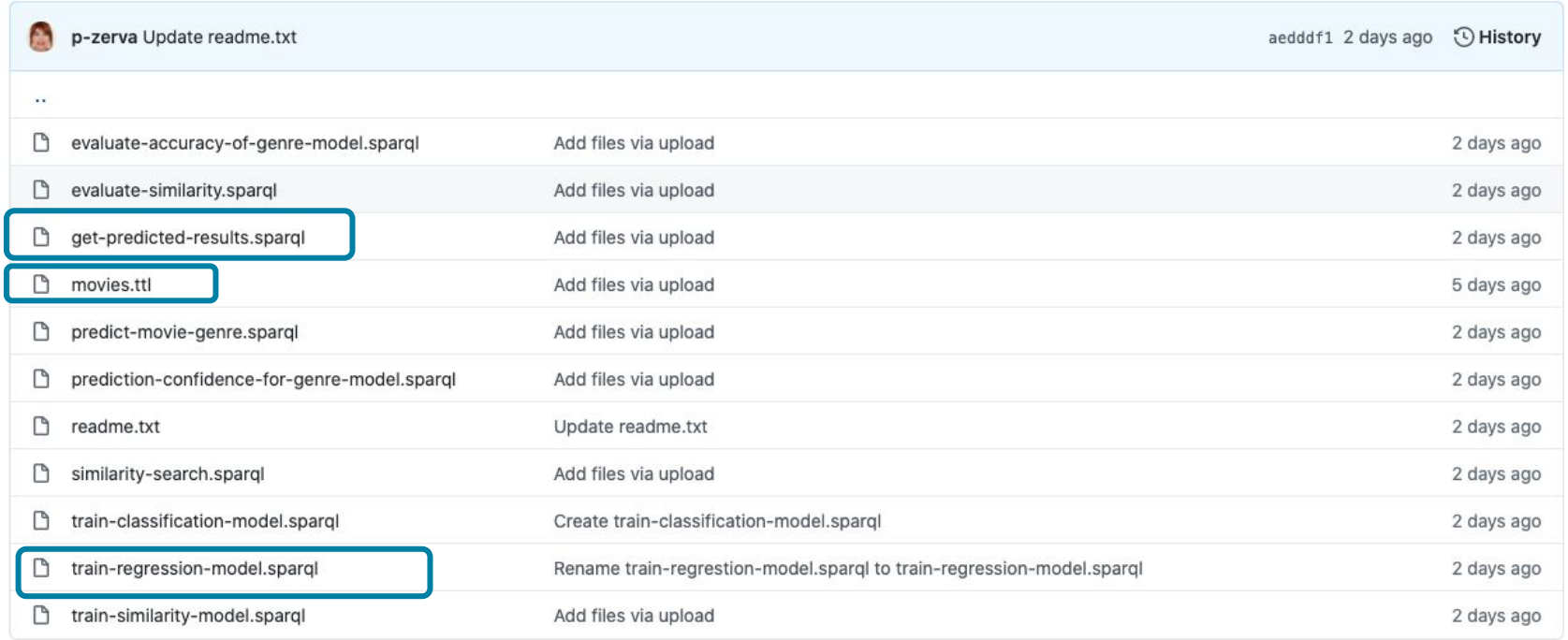
Note : By default, **spa:accuracy** is used for classification problems, and **spa:mae** for regression. This metric can be changed during model learning, by setting the **spa:evaluationMetric** argument.

Evaluation Measure Example

```
prefix agg: <urn:aggregate>
prefix spa: <tag:stardog:api:analytics:>

INSERT {
    graph spa:model {
        :RatingModel a spa:RegressionModel ;
            spa:arguments (?genres ?contentRating
?storyline ?metaCritic) ;
            spa:predict ?rating ;
            spa:crossValidation 100 ;
            spa:evaluationMetric spa:mae .
    }
}
...
```

Demo

Access ML Stardog Repository



p-zerva Update readme.txt — aedddf1 2 days ago — History

..		
evaluate-accuracy-of-genre-model.sparql	Add files via upload	2 days ago
evaluate-similarity.sparql	Add files via upload	2 days ago
get-predicted-results.sparql	Add files via upload	2 days ago
movies.ttl	Add files via upload	5 days ago
predict-movie-genre.sparql	Add files via upload	2 days ago
prediction-confidence-for-genre-model.sparql	Add files via upload	2 days ago
readme.txt	Update readme.txt	2 days ago
similarity-search.sparql	Add files via upload	2 days ago
train-classification-model.sparql	Create train-classification-model.sparql	2 days ago
train-regression-model.sparql	Rename train-regrestion-model.sparql to train-regression-model.sparql	2 days ago
train-similarity-model.sparql	Add files via upload	2 days ago

[Link to Stardog Learning Repository]

Unsupervised Learning

Unsupervised Learning Steps

- Similar steps to Supervised Learning but no data preparation is required
- Prepare training data is not required
- Determine input features
- Train a model using the data and the selected features
- Evaluate the accuracy of the learned model

Similarity Model

Predict Movie Similarity

- Find similar movies based on the properties of movies
- This model will find similar movies based on their genres, directors, authors, producers, and MetaCritic score.
- Clustering problem

I) Prepare Training Data & II)Feature Selection

```
... WHERE {
    SELECT
    (spa:set(?genre) as ?genres)
    (spa:set(?director) as ?directors)
    (spa:set(?author) as ?authors)
    (spa:set(?producer) as ?producers)
    ?metaCritic
    ?movie
    {
        ?movie  :genre ?genre ;
                :director ?director ;
                :author ?author .

        OPTIONAL {
            ?movie  :productionCompany ?producer .
        }

        OPTIONAL {
            ?movie  :metaCritic ?metaCritic .
        }
}
    GROUP BY ?movie ?metaCritic }
```

III) Train Model

- The underlying algorithm is based on cluster pruning, an approximate search algorithm which groups items based on their similarity to speed up query performance.
- This number should be increased with datasets containing many near-duplicate items.

```
INSERT {
    graph spa:model {
        :s1 a spa:SimilarityModel ;
            spa:arguments (?genres ?directors ?authors ?producers ?metaCritic) ;
            spa:predict ?movie ;
            spa:overwrite True .
    }
  }
```

IV. Evaluate Model (1/2)

- During prediction, there are two parameters available:
  - spa:limit : restricts the number of top N items to return; by default, it returns only the top item, or all items if using spa:confidence.
  - spa:clusters, which sets the number of similarity clusters used during the search, with a default value of 1. Larger numbers will increase recall, at the expense of slower query time.

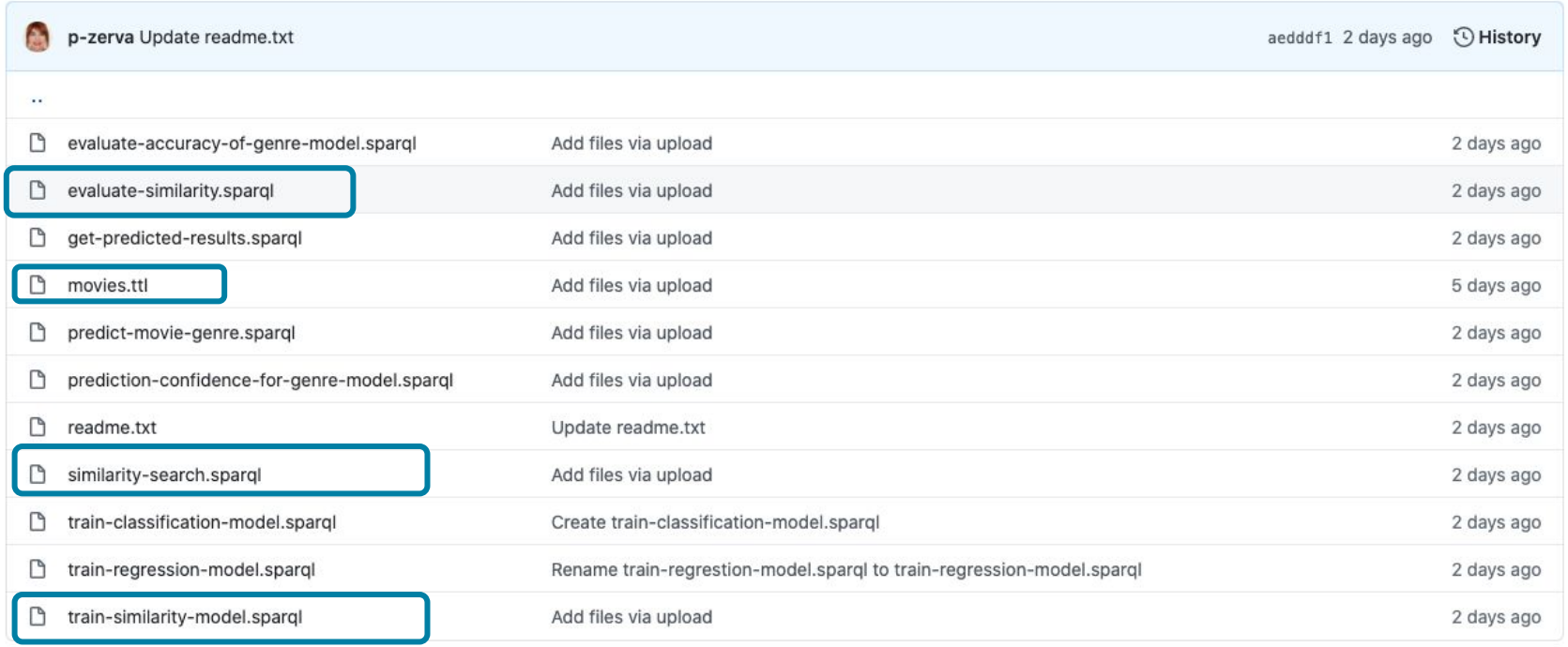
Evaluate Model (2/2)

- For example, the following query will return the 3 top most similar items and their confidence scores restricting the search to 10 clusters.

```
SELECT * WHERE {
graph spa:model {
:myModel  spa:parameters [
                spa:limit 3 ;
                spa:clusters 10 . ] ;
              spa:confidence ?confidence ;
              spa:arguments (?director ?year ?studio) ;
              spa:predict ?similar .
}
} ...
```

Demo

Access ML Stardog Repository



p-zerva Update readme.txt		aedddf1 2 days ago	History
..			
evaluate-accuracy-of-genre-model.sparql	Add files via upload	2 days ago	
evaluate-similarity.sparql	Add files via upload	2 days ago	
get-predicted-results.sparql	Add files via upload	2 days ago	
movies.ttl	Add files via upload	5 days ago	
predict-movie-genre.sparql	Add files via upload	2 days ago	
prediction-confidence-for-genre-model.sparql	Add files via upload	2 days ago	
readme.txt	Update readme.txt	2 days ago	
similarity-search.sparql	Add files via upload	2 days ago	
train-classification-model.sparql	Create train-classification-model.sparql	2 days ago	
train-regression-model.sparql	Rename train-regrestion-model.sparql to train-regression-model.sparql	2 days ago	
train-similarity-model.sparql	Add files via upload	2 days ago	

[Link to Stardog Learning Repository]

Additional Features

Model Data (1/2)

- Modeling data with correct datatypes can increase model quality
- Datatypes in Stardog can be modified at query level (training time)
- Stardog does special treatment on values of the following types:
  - **Numbers**, such as xsd:int, xsd:short, xsd:byte, xsd:float, and xsd:double, are treated internally as weights and properly model the difference between values
  - **Strings**, xsd:string and rdf:langString, are tokenized and used in a bag-of-words fashion
  - **Sets**, created with the spa:set operator, are interpreted as a bag-of-words of categorical features
  - **Booleans**, xsd:boolean, are modeled as binary features
  - Everything else is modeled as categorical features

Model Data (2/2)

- Setting the correct data type for the target variable, given through spa:predict, is extremely important:
  - with regression, make sure values are numeric
  - with classification, individuals of the same class should have consistent data types and values
  - with similarity, use values that uniquely identify an object, e.g., an IRI

Hyperparameter Optimization (1/2)

- Finding the best parameters for a model is a time consuming, laborious, process.
- Stardog helps to ease the pain by performing an exhaustive search through a manually specified subset of parameter values.

```
INSERT { graph spa:model {
    :myModel  a spa:ClassificationModel ;
                spa:parameters [
                spa:learning_rate (0.1 1 10) ;
                spa:hash ('all' 'strings')
              ] ;
              spa:arguments (?director ?year ?studio) ;
              spa:predict ?genre . } } ...
```

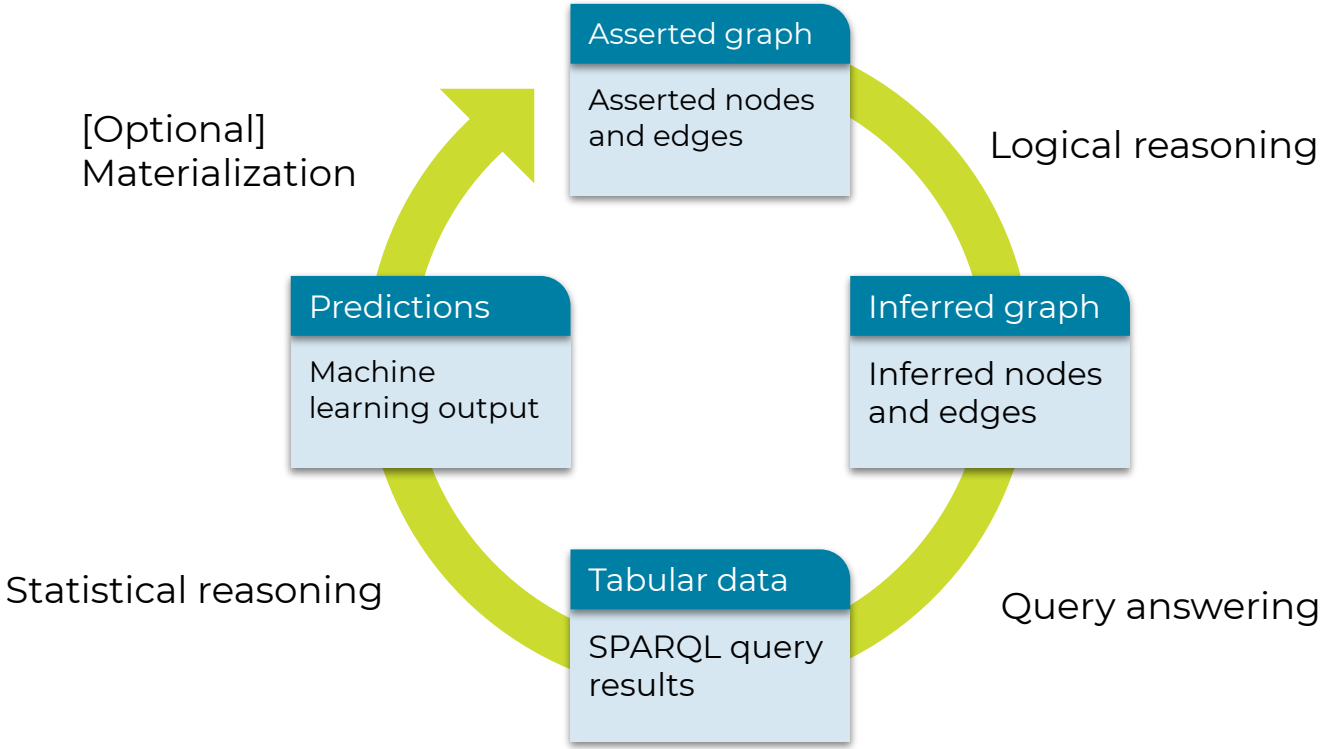
Hyperparameter Optimization (2/2)

- All possible sets of parameter configurations that can be built from the given values spa:learning_rate 0.1 ; spa:hash 'all', spa:learning_rate 1 ; spa:hash 'all', and so on will be evaluated.
- The best configuration will be chosen, and its model will be saved in the database.
- Afterwards, parameters are available for querying, just like any other model metadata.

```
prefix spa: <tag:stardog:api:analytics:>

SELECT * WHERE {
    graph spa:model {
    :myModel  spa:parameters
              [ ?parameter ?value ]
    }
}
```

```
+-------------------+-------+
|     parameter     | value |
+-------------------+-------+
| spa:hash          | "all" |
| spa:learning_rate | 1     |
+-------------------+-------+
```

Reasoning Flow



Learning Objectives

- The Machine Learning Model Development Life Cycle
- Stardog's Machine Learning Features and Services for statistical predictions
- How to prepare training data for a Machine Learning model using Stardog
- Steps to train and evaluate a supervised-learning classification model and a Machine Learning regression model using Stardog
- How to build a similarity model using Stardog
- How to tune and optimize hyper-parameters to improve model performance
- Steps to prepare and develop your data in preparation for your model training

Thank you