A Linked Data-Based Decision Tree Classifier to Review Movies

Suad Aldarra[1] and Emir Muñoz[1,2]

[1] Fujitsu Ireland Limited
[2] Insight Centre for Data Analytics, National University of Ireland, Galway
E-mail: {Suad.AlDarra, Emir.Munoz}@ie.fujitsu.com

Abstract. In this paper, we describe our contribution to the 2015 Linked Data Mining Challenge. The proposed task is concerned with the prediction of review of movies as "good" or "bad", as does Metacritic website based on critics' reviews. First we describe the sources used to build the training data. Although, several sources provide data about movies on the Web in different formats including RDF, data from HTML pages had to be gathered to fulfill some of our features. We then describe our experiment training a decision tree model on 241 features derived from our RDF knowledge base, achieving an accuracy of 0.94.

1 Introduction

In this paper we describe the method used in our submission to the 2015 Linked Data Mining Challenge[1] at the Know@LOD Workshop. The challenge propose the task of predict whether a movie is "good" or "bad" based on the value of its RDF properties. These labels are as the ones used in the Metacritic[2] website, based on critics' reviews submitted to their system. Metacritic originally use three categories based on the critics: *positive*, *negative*, and *mixed*; according to a score ranging from 0 to 100. For simplicity, in this challenge, only two classes are required, and movies with score above 60 are regarded as "good", while movies with score less than 40 are regarded as "bad". To achieve this goal we learn a Decision Tree classifier [1], which can efficiently assign a binary label to incoming unlabeled/unseen movies.

To design our classifier, we solved two main challenges: 1) the collection/-transformation of relevant data about movies, and 2) the design of features from RDF data to train our classifier. We address the two challenges in this work with an estimated 70-30% effort, respectively. First, we collect data from several sources, including HTML pages, and convert it to RDF. Second, we use SPARQL queries to generate suitable data format for the learning process.

In the remaining of this paper, we describe how we address both challenges. We describe the construction of our RDF knowledge base, feature extraction, and experiment to learn the decision tree with its corresponding evaluation.

[1] http://knowalod2015.informatik.uni-mannheim.de/en/linkeddataminingchallenge/
[2] http://www.metacritic.com/

2 RDF knowledge base construction

The provided data comprises 2,000 movies along with their name, release date, DBpedia URI, class (good/bad), and ID. From the data, 80% (1,600) is used during the training step, and 20% (400) during the testing step. The DBpedia URIs are used to access the LOD cloud for collecting further data about movies. Although, several LOD datasets contain relevant data for this task, namely, DBpedia[3], LinkedMDB[4], Freebase[5], none of them contain high quality, complete, and up-to-date data in one place. Thus, we were forced to build our own RDF knowledge base, gathering facts from different RDF sources plus other (semi-/un-)structured data sources. The final list of sources included in our knowledge base comprises: IMDB[6], OMDB[7], Metacritic, Freebase, and DBpedia.

We start retrieving `dcterms:subject` values for a movie from DBpedia.We use DBpedia `sameAs` links to Freebase to get a movie's IMDB ID. Movies data (*e.g.*, year, release, genre, director, starring, MPAA rating) were collected from OMDB in JSON format and then converted into RDF programmatically. We queried OMDB using the movie's IMDB ID instead of the movie title provided, since the search was more accurate in most cases. We retrieved data about actors and directors from Freebase using OpenRefine[8]. Thus, we could collect personal information about actors and directors, such as, genre, nationality, date of birth, IMDB ID, among others. Other information was extracted from IMDB: actors, directors and movies awards, movies budget, gross, common languages and countries. For each movie, we also extracted its IMDB keywords, which are later used to determine common keywords among good and bad movies.

Finally, for each movie we collected textual critics' reviews from Metacritic website and applied an existing API for sentiment analysis using NLTK[9], which returns either a positive, negative or neutral sentiment label for a given text.

Our resulting RDF knowledge base comprises 338,140 RDF triples that are accessed using SPARQL queries to generate our set of features to train a decision tree model. (All data in RDF, decision tree model and diagram, and feature vectors are available at `https://github.com/emir-munoz/ldmc2015`.)

3 Experiment

In the following we present our experiment set up to train and evaluate the proposed decision tree. Figure 1 shows a flow diagram of the data and processes involved. In order to train a decision tree classifier, we first define a set of features to be extracted from our RDF knowledge base (Movies DB). Movies DB is stored in a Virtuoso Server running on a CentOS Linux virtual machine (with 4.0 GHz CPU and 7.5 GB of RAM), and queried via HTTP.

[3] `http://dbpedia.org/`
[4] `http://www.linkedmdb.org/`
[5] `http://www.freebase.com/`
[6] `http://www.imdb.com/`
[7] `http://www.omdbapi.com/`
[8] `http://openrefine.org/`
[9] `http://text-processing.com/docs/sentiment.html`

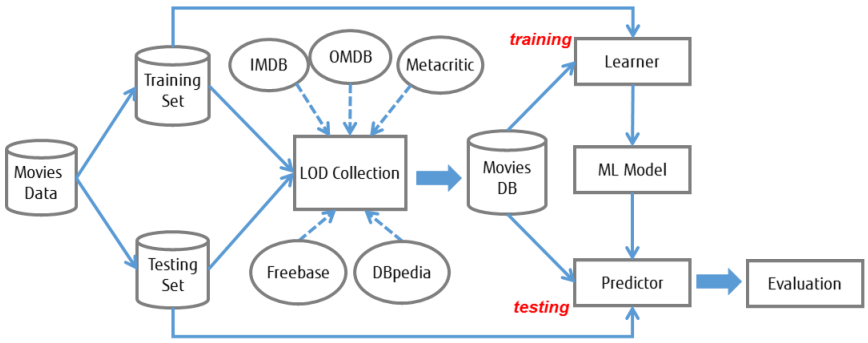

Fig. 1: System architecture with training and evaluation parts.

3.1 Feature set

Once the RDF KB was finished, we defined a set of 241 features. Our features contain mixed continuous (numerical) and dichotomous (categorical) types that can be handled by C4.5 algorithm [2]. The following list summarize the features used in this work. ($\star$ = feature considers the release/record date of the movie.)

- `dcterms:subject` values
- genres of a movie
- countries of a movie
- languages of a movie
- MPAA rating
- # of directors' Oscar/Golden Globe awards won/nominated ($\star$)
- # of actors' Oscar/Golden Globe awards won/nominated ($\star$)
- runtime
- release week/weekend day
- # of bad/good/neutral/mostly-good/mostly-bad keywords
- # of female/male actors
- # directors younger than 30 ($\star$)
- # directors between 30 and 50 ($\star$)
- # directors older than 50 ($\star$)
- # actors younger than 30 ($\star$)
- # actors between 30 and 50 ($\star$)
- # actors older than 50 ($\star$)
- is the movie from a common country?
- is the movie in a common language?
- low or high amount of budget?
- is the gross higher than the budget?
- % of positive critics' reviews
- % of negative critics' reviews
- % of neutral critics' reviews
- is the movie based on a book?
- is the movie a sequel?
- is the movie an independent film?

Features are extracted from the data using `SELECT` and `ASK` SPARQL queries. For instance, the query on the right, get the age value for each actor involved in the movie "Amores Perros". These values are then used to generate three of our features. A similar query is performed to get the age for directors.

```
SELECT ?age WHERE {
  dbr:Amores_perros rdf:type dbo:Film .
  dbr:Amores_perros dbp:recorded ?recorded .
  dbr:Amores_perros dbp:starring ?actor .
  ?actor dbp:dateOfBirth ?dob .
  BIND (?recorded - YEAR(?dob) AS ?age)
}
```

3.2 Learning process

After all features are extracted for both train and test sets, we use the J48 classifier, a Weka implementation for C4.5 algorithm. The decision tree settings consider pruning of the tree, and a confidence factor equals to 0.25.

Figure 2a reports the confusion matrix resulting from a 10-fold cross-validation over the train data.

		Prediction	
		good	bad
Gold	good	$tp = 752$	$fn = 49$
	bad	$fp = 48$	$tn = 751$

(a) Confusion matrix

$$Acc = P(\text{correct}) = \frac{tp + tn}{tp + fp + fn + tn}$$

(b) Accuracy metric

Fig. 2: Model evaluation.

Using equation in Figure 2b to compute the accuracy of our system, we achieve Acc =0.94 on the train set. The challenge system reports an Acc =0.9175 for our submission on the test set.

4 Conclusion

We have described our submission to the 2015 Linked Data Mining Challenge, presenting a decision tree classifier to solve the prediction problem of review of movies. We trained this decision tree on 1,600 examples, with input features extracted from a built-in RDF knowledge base using SPARQL queries.

In order to reduce the features space, feature aggregation was applied over actors, directors, and critics' reviews. The sentiment analysis over critics' reviews generate the attributes with higher information gain [3]. Negative critics have an information gain of 0.71886 bits, thus, selected as root of the decision tree. Experiments removing all sentiment features from the training show that accuracy is reduced by *ca.* 9%. While removing positive or negative does not affect the accuracy severely. That shows the relevance of sentiment analysis-based features for this task, which are directly related to the taste of users.

Movie keywords are the next features with higher information gain, and their analysis provide interesting insights to be considered by writers and directors: a) *bad* movies are based on video games, with someone critically bashed, using a taser, pepper spray, or hanged upside down, with dark heroine involved; and b) *good* movies include family relationships, frustration, crying, melancholy, very little dialogue, and some sins with moral ambiguity—yes, people like drama.

Acknowledgments. This work has been supported by KI2NA project funded by Fujitsu Laboratories Limited and Insight Centre for Data Analytics at NUI Galway.

References

1. Quinlan, J.: Simplifying decision trees. International Journal of Man-Machine Studies **27**(3) (1987) 221 – 234
2. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques. 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2011)
3. Basuroy, S., Chatterjee, S., Ravid, A.S.: How Critical Are Critical Reviews? The Box Office Effects of Film Critics, Star Power, and Budgets. Journal of Marketing **67**(4) (October 2003) 103–117