Utilizing the Open Movie Database API for Predicting the Review Class of Movies

Johann Schaible[1], Zeljko Carevic[1], Oliver Hopt[1], and Benjamin Zapilko[1]

GESIS - Leibniz Institute for the Social Sciences, Cologne, Germany
{firstName.lastName}@gesis.org

Abstract. In this paper, we present our contribution to the Linked Data Mining Challenge 2015. Our approach predicts the review class of movies using external data from the Open Movie Database API (OMDb-API). We select specific features, such as movie ratings and box office, that are very likely to describe the quality of a movie. With RapidMiner we utilize these features and apply three basic classification algorithms to train and validate the prediction model using a 10-fold cross-validation. The results of our evaluation are interesting in a two-fold way: (i) few movie ratings from professional critics provide a higher accuracy (accuracy $\approx$ 0.94) than many ratings from users (accuracy $\approx$ 0.7), and (ii) the Decision Tree classifier (accuracy ≈ 0.83) outperforms Naive Bayes (accuracy ≈ 0.73), whereas k-NN is not suitable at all (accuracy ≈ 0.53).

1 Introduction

In the Linked Data Mining Challenge 2015[1] participants were asked to predict a movie's review class, i.e. one needs to identify whether a movie is labeled as *good* or as *bad*. The training set contains solely the movie title, its release date, its DBpedia[2] URI, as well as the actual label.

Instead of developing sophisticated data mining algorithms or adapting existing ones to the challenge task, we focus on selecting and calculating specific features out of particular data sets that can be used by state of the art classification algorithms to provide a statement about a movie's quality. In detail, we extend the training set with the publicly available data from the Open Movie Database API[3] (OMDb API) containing various movie ratings and box office information. With RapidMiner[4], we apply the Naive Bayes, the k-NN, and the Decision Tree classifier to train and evaluate the prediction model using a 10-fold cross-validation. Each feature is evaluated alone as well as in combination.

In the following section, we describe the utilized data set and our evaluation setup. We present and discuss our results in detail in Section 3.

[1] http://knowalod2015.informatik.uni-mannheim.de/en/linkeddataminingchallenge/
[2] http://de.dbpedia.org/
[3] http://www.omdbapi.com/
[4] https://rapidminer.com/

2 Our Approach

2.1 Extending the Data

The information retrieved from the OMDb API enables to provide a statement on a movie's quality. For example, it illustrates how many awards a movie has won or was nominated for, as well as movie ratings such as the IMDB[5] rating and several Rotten Tomatoes[6] ratings. The information also contains the Metacritic's[7] Metascore that is used as ground truth for the challenge. However, we did not make use of the Metascore for tuning the prediction model in any way. The API allows for querying the data source by various criteria, of which we used the movie title and release year. All its content is licensed under Creative Commons Attribution 4.0 International Public License.[8] Hence, we were allowed to publish the relevant parts of this data as Linked Data which was done by following the guidelines of Heath and Bizer [1]. To express the full semantics of the data, we had to define a few datatype properties of our own under the namespace *gmovies*: `http://lod.gesis.org/gmovies/`. Listing 1.1 illustrates an excerpt of the published data in turtle syntax for the movie "The Godfather".

```
<http://lod.gesis.org/gmovies/The_Godfather>
  a <http://dbpedia.org/ontology/Film>;
  dcterms:title "The_Godfather";
  owl:sameAs <http://dbpedia.org/resource/The_Godfather>;
  .
  .
  gmovies:numberOfAwards "52";
  gmovies:tomatoMeter "99";
  gmovies:tomatoFreshRatio "0.9879518072289156";
  gmovies:tomatoRottenRatio "0.012048192771084338";
  rdfs:seeAlso <http://www.omdbapi.com/?t=The+Godfather&y=1972&tomatoes=true>;
  foaf:page <www.imdb.com/title/tt0068646>.
```

Listing 1.1. An excerpt of the RDF representation of the data from the OMDb API for the movie "The Godfather" in Turtle syntax.

The title, release date, and DBpedia link are obtained from the data provided by the challengers. The various ratings, meters, and IMDB page are retrieved directly from the OMDb API. We also defined the further metrics *numberOfAwards*, *tomatoFreshRatio*, and *tomatoRottenRatio*. The number of awards counts the awards the movie has won or was nominated for. The other two ratios are based on the Tomatometer and are defined by the number of critics rating a movie as *fresh* or *rotten* divided by the total number of critics.

The RDF representation of the additional data for all movies contained in the challenge is published as LOD.[9] The example for the movie "Skyfall"[10] illustrates the several data type properties and the possibility to download the data as Turtle, as RDF/XML, or query it via a SPARQL endpoint.

[5] `http://www.imdb.com/`
[6] `http://www.rottentomatoes.com/`
[7] `http://www.metacritic.com/`
[8] `http://creativecommons.org/licenses/by/4.0/legalcode`
[9] `http://lod.gesis.org/gmovies/`
[10] `http://lod.gesis.org/pubby/page/gmovies/Skyfall`

2.2 Evaluation Setup

To train and evaluate the prediction model, we used the RapidMiner Studio (free edition). The Linked Open Data Extension[11] was used to query the previously defined RDF data from the OMDb API. Subsequently, the extended data set is forwarded to the RapidMiner process *X-Validation*, i.e. the built-in 10-fold cross-validation. Three different prediction models were trained and evaluated, based on Naive Bayes, k-NN, and a Decision Tree classifier. For better comparability of the additional features, we used a 3-NN classifier like it was provided by baseline of the challenge. For the Naive Bayes classifier we used the Laplace correction to prevent high influence of zero probabilities. Besides that, we used the RapidMiner's default setting for each classifier. Hereby, the RapidMiner's decision tree learner works similar to Quinlan's C4.5 [2] or CART [3] with a maximal depth of 20. The criterion determining the type of the tree is set to "gain_ratio".

The entire RapidMiner process as well as the XML test set including the predicted labels can be downloaded at the GESIS data repository service "datorium".[12]

3 Results and Discussion

Results. The results of the 10-fold cross-validation on the training data set is illustrated in Table 1. It shows the accuracy ACC of the combinations between the three classifiers and the various features, which is defined as follows:

$$ACC = \frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{True positive} + \sum \text{False positive} + \sum \text{True negative} + \sum \text{False negative}}$$

The 3-NN classifier did not reach a prediction accuracy of 60%, whereas the the Naive Bayes ($ACC \approx 0.73$) and the Decision Tree ($ACC \approx 0.83$) approaches performed quite well. Regarding the overall precision and recall when predicting the labels *good* and *bad*, the results are as follows: The 3-NN has a precision of $p = 0.51$ and a recall of $r = 0.99$ when predicting the label *good*. When predicting the label *bad*, it has a precision of $p = 0.64$ and a recall of $r = 0.1$. The Naive Bayes classifier has in both cases a precision of $p = 0.95$ and recall $r = 0.93$. Finally, RapidMiner's Decision Tree has a precision of $p = 0.86$ and a recall of $r = 0.92$ when predicting label *good*. When predicting label *bad*, its precision is $p = 0.9$ and recall $r = 0.84$.

Considering the different features, with an accuracy of about 90% the Tomatometer/rating scores provide a better accuracy than the user generated Tomato-scores or the IMDB score (accuracy between 70% and 80%). The box office information (ACC $\approx 50\%$) does not seem to provide an appropriate feature to predict a movie's review class at all. Winning awards or being nominated for awards indicates the label of a movie, but with about 65% it does not make a clear statement as the other features.

Discussion. The submitted configuration, which achieved an accuracy of $ACC_t = 0.97$ on the training data, achieved only an accuracy of $ACC_e = 0.95$ on the evaluation set. Such an overfitting is quite typical for Decision Tree algorithms [4].

[11] http://dws.informatik.uni-mannheim.de/en/research/rapidminer-lod-extension/

[12] http://dx.doi.org/10.7802/78

	Naive Bayes	3-NN	Decision Tree
Awards won or nominated for	0.66	0.52	0.69
Box Office information	0.49	0.59	0.5
IMDB Rating	0.76	0.52	0.87
IMDB Rating + number of votes	0.73	0.51	0.86
Tomatometer	0.91	0.51	0.94
Tomatorating	0.87	0.54	0.95
Tomato Fresh/Rotten Ratio	0.93	0.54	0.94
Tomato User Meter	0.71	0.51	0.81
Tomato User Rating	0.68	0.53	0.79
Tomato User Rating + number of reviews	0.68	0.51	0.78
All above features combined	0.94	0.52	0.96
Tomatometer + Tomatorating + Fresh/Rotten Ratio	0.95	0.51	**0.97**

Table 1. Results of the 10-fold cross-validation on the training data. The left column illustrates the different features used in the evaluation. The accuracy values for the three classifiers are presented in the three columns from the right. The value marked as bold represents the setup we have submitted to the challenge.

As the model is trained by maximizing its prediction performance, the number and performance of the provided features might lead to memorizing the training data. Thus, it decreases the prediction performance on new and previously unseen data. However, decision trees use the "divide and conquer" method, so they tend to perform well on a few highly relevant features, like in our use-case [4]. On the contrary, Naive Bayes takes features and values into account, which Decision Trees have already eliminated [4]. However, as we use only a small amount of features, the Naive Bayes approach cannot use this big advantage to outperform the Decision Tree algorithm. The k-NN classifier cannot use the additional features in a way the other classifiers can at all. Taking further looks on its predictions, we observed that the 3-NN classifier predicted the label *good* in over 95% of cases. This observation correlates with the precision and recall values of its predictions. The most probable reason for this is the dimensionality and a normalization of distance between single data values. In detail, we did not normalize the data, so that the distance measure might have been dominated by features with a large scale [4]. Thus, the various features did not play a similar role in determining the distance, so that no good prediction could be produced.

Regarding the various rating features, we observed that user generated critics, such as the IMDB score and the Tomato user rating/meter, provide a 10 to 20 percent lower prediction accuracy than the "official" critics like the Tomatometer. The reason for this, is that Metacritic's Metascore, which is used as ground truth, is a weighted average of scores from top critics.[13] These critics are 30-50 writers from the most recognizable journals in the movie industry. Similar to that, the Tomatometer reflects the percentage of up to 200 critics, who are involved in print or online publications and maintain a certain level of quality and consistency.[14] Thus, a high Tomatometer value reflects the

[13] http://www.metacritic.com/about-metascores

[14] http://www.rottentomatoes.com/about/

Metascore better than a high user-generated IMDB value. Regarding the other features, the number of winning or nominated awards provides a decent prediction accuracy as well. Distinguishing between a movie having no award nomination and a missing value might increase the accuracy, as currently the value for both is ”N/A“. Such data sparseness is also the reason for the low prediction accuracy using the box office information (missing values for about 50% of the movies). Using LOD sources like DBpedia or LinkedMDB posed the same problem. To generate good predictions, it is crucial for the additional data to be as less sparse as possible. DBpedia did not provide information, e.g. a movie's budget and gross income, for almost half of the movies in the challenge. One could interlink LOD sources with each other to overcome data sparseness. However, such a process is quite challenging, as first one needs to find LOD sources containing similar data, second one must be familiar with the source's schema, and third one might have to deal with different data formats, such as digits vs. text describing a movie's gross income , in order to apply classification algorithms.

In our evaluation we primarily used features that describe a movie's quality via some sort of rating. These features are likely to be very close to the metric defining the ground truth. Thus, using our approach is only possible if such features already exist for a given movie. Predicting a review class of a movie that is yet to come out, i.e. it does not have any ratings yet, will be impossible using our approach. However, more sophisticated mining algorithms might increase the prediction accuracy for several features that are already known before the rating of a movie. For example, some features could express the cast and crew reputation of a movie, e.g. awards of actors and directors. Using such features and sophisticated data mining algorithms might provide a quite decent prediction of the movie's quality.

4 Conclusion

The results of our evaluation show that using state of the art classifiers makes it possible to achieve a high prediction accuracy, if one uses various ratings that were generated by critics maintaining a certain level of quality and consistency. Furthermore, information such as award nominations are likely to provide adequate results, if the data is not too sparse. To follow this initiative, we will publish the generated data set as LOD and extend it with further data from other sources, e.g. links to persons and other relevant classes from the movies domain.

References

1. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web. Morgan & Claypool Publishers (2011)
2. Quinlan, J.R.: C4. 5: programs for machine learning. Elsevier (2014)
3. Lewis, R.J.: An introduction to classification and regression tree (cart) analysis. In: Annual Meeting of the Society for Academic Emergency Medicine in San Francisco. (2000) 1–14
4. Entezari-Maleki, R., Rezaei, A., Minaei-Bidgoli, B.: Comparison of classification methods based on the type of attributes and sample size. Journal of Convergence Information Technology **4**(3) (2009) 94–102