CS 839 Project-3

by Vishal Agarwal, Roshan Lal and Pratham Desai

Data Description

We extracted movie information from the following two sources:

- Source-1: www.imdb.com/ (Internet Movie Database)
    - Number of tuples from imdb: 7846
- Source-2: https://www.allmovie.com/
    - Number of tuples from allmovie: 7258

Blocking Technique

In order for a tuple-pair to pass through the blocker, it should satisfy **all** of the following conditions:

1. Movie name - overlap should be at least one; whitespace separated word level tokens; excluding stop words
2. Director name - overlap should be at least one; whitespace separated word level tokens
3. Year of release - absolute difference between year of release should be less than or equal to two years

Sample

We labelled 1500 tuple-pairs. 1000 were used for training and 500 were used for testing.
|G| = 1500 tuple-pairs; |I| = 1000 tuple-pairs; |J| = 500 tuple-pairs

Training

We performed cross-validation within *set I* with k=2. The following table describes Precision, Recall and F1 scores for different classifiers.

Classifier	Precision	Recall	F1
Decision Tree	0.987865	0.979937	0.983885
Random Forest	0.986043	0.983725	0.984873
SVM	0.98343	0.952298	0.967604
Linear Regression	0.987859	0.979712	0.983764
Logistic Regression	0.983978	0.985844	0.984909

Naive Bayes	0.985845	0.969569	0.977619

All the learning methods yielded high precision and recall for cross-validation on set I. We select **Logistic Regression** as our best matcher in the cross validation step.

We already received very high precision, recall and F1 score. Therefore, we did not need to go through the debugging stage. The final best matcher that we select is **Logistic Regression.**

Best Matcher: Logistic Regression
Average precision on set I (cross-validated) : 98.40%
Average recall on set I (cross-validated) : 98.58%
Average F1 score on set I (cross-validated) : 98.49%

Testing on Set J

We trained the six matchers on the entirety of set I, and used them to predict labels for the test set J. The following table describes various metrics on the test set.

Classifier	Precision (%)	Recall (%)	F1 (%)
Decision Tree	98.82	98.05	98.43
Random Forest	98.82	98.44	98.63
SVM	98.39	95.31	96.83
Linear Regression	98.43	98.05	98.24
Logistic Regression	97.67	98.44	98.05
Naive Bayes	98.02	96.48	97.24

The best matcher selected through cross-validation on set I was Logistic regression. For this classifier,

Precision on (test set J) : 97.67%
Recall on (test set J) : 98.44%
F1 score (test set J) : 98.05%

Approximate time estimates

Stage	**Time (human hours)**
blocking	4
labelling data	1.5
finding the best matcher	3

Obtaining a higher precision

There seems to be a clear pattern in the false positives predicted by the algorithm. The movie sequel names, that differ from their prequel by just a number (For eg. The Grudge and The Grudge 2), are being falsely predicted as a match. This makes sense, since it would result in a very high value for all the similarity measures. If we had a high number of such mismatches, we can include a feature in our algorithm that specifically checks if the two movies have a prequel-sequel relationship between them.

Serial No.	**Source 1**	**Source 2**
1	Kill Bill: Vol. 2	Kill Bill Vol. 1
2	The Other	The Other Woman
3	The Karate Kid Part II	The Karate Kid
4	Lethal Weapon 2	Lethal Weapon
5	The Grudge 2	The Grudge
6	Harry Potter and the Deathly Hallows: Part 2	Harry Potter and the Deathly Hallows; Part 1

Obtaining a higher recall

We need to look at the false negatives predicted by our algorithm in order to achieve higher recall. The machine learning algorithm that we trained on Set I already had a very high recall on the test set J, but here we present some interesting analysis of the false negatives.

The movies that we see in this list differ in the way they have been represented on the two website. For example, the movie *Precious (2009)* was listed by its name on imdb.com It was based on a novel *Push*, and this information was included in the title itself on allmovie.com. So, the algorithm failed to predict that these are the same movies. If we wanted to remove this false negative, we would need to include some feature in our ML algorithm that checks for subtitles within a movie title.

Another example is *Guns for San Sebastian,* which was listed in another language on allmovie.com . This was the reason why they were not predicted as a match. If we have a lot of cases like this in our data, it may be important to detect them. In that case, we would first need to normalize all movie names to English language by using a translator, and then perform entity matching.

Also, there is sometimes a difference in the way sequels are listed on the two sources. One such example is listed below at no. 3, representing the movie Dark Harvest 2 by two different names.

Serial No.	**Source 1**	**Source 2**
1	Precious	Precious: Based on the Novel 'Push' By Sapphire
2	Guns for San Sebastian	La Bataille de San Sebastian
3	The Maize: The Movie	Dark Harvest 2: The Maize

Comments on Magellan

Overall we found Magellan extremely easy / intuitive to work with. We were surprised at how easily we were able to perform entity matching on an actual data set.

Positives

- powerful functions like *select_matcher*
- debugging the output of the blocker using *debug_blocker* is well... extremely sexy!

Areas for improvement

- would have liked to have an easy way to print false positives and false negatives from *eval_matches.* Even though there are visual debuggers for certain classifiers, this basic debugging functionality would be good to have for all the classifiers