DEPARTMENT OF COMPUTER SCIENCES, UNIVERSITY OF WISCONSIN-MADISON

Report for CS 784 Project Phase 3

Shruthi Racha, Ashish Shenoy
November 24, 2015

1 PREPARING THE GOLDEN DATA

The goal of this step is to find the best machine learning algorithm (matcher) for matching the entities obtained after the blocking step.

- Using the candidate set obtained after the blocking process, we arrived at a sub sample data set, *S*, of size 450.
- The sample set, *S*, of 450 was labelled and cleaned up to remove the tuples for which classification was ambiguous to arrive at a golden data set, *G* of size 448.

2 FIRST ITERATION

We divide *G* into *I* (training set) and *J* (evaluation set). The six matchers available in Magellan was trained on *I*. We obtained the following results after performing cross validation for the first time for these methods on *I* :

Matcher	Precision	Recall	F1 Score
Decision Tree	98.33	98.66	97.90
Random Forests	100.00	97.61	98.77
Support Vector Machine	80.00	9.03	16.17
Naive bayes	97.66	98.04	97.81
Logistic Regression	98.57	96.18	97.28
Linear Regression	99.09	92.57	95.62

Table 2.1: Precision, Recall and F1 measures (in percentage) of prediciton on I in first iteration

After obtaining the above results, we decided to select *Random Forest Classifier* as the matcher to be used on the evaluation set. But before evaluating on *J* the following steps were performed to debug and improve the recall of linear regression matcher.

- *I* was split into training set *U* and testing set *V* using a proportion of 0.7.
- Random Forest Matcher was trained on *U* and was fit on *V* and results as shown in Table 2.2 were obtained.
- The Precision and Recall on *U* and *V* was high but there were still one false negative.

Matcher	Set	Precision	Recall	F1 Score	False Pos	False Neg
Random Forests	Fit on U Predict on V	100.00	96.15	98.04	0	1

Table 2.2: Precision, Recall, F1 measures (in percentage), False Positives and False Negatives of prediction on V using Linear Regression

- We decided to tackle the False Negative by using the debugger provided by Magellan for Randomized Forest matcher and implement a trigger.

3 Second Iteration

We used *mg.vis_debug_rf* to identify the false negative in the Randomized Forest Matcher on V. The tuple which was being falsely predicted as negative was as follows :

```
The Taco Shop (608) 250-8226 604 University Ave, Madison, WI
The Taco Bros (608) 422-5075 604 E University Ave, Madison, WI 53715
```

- After examining the feature vector for the above tuple we decided to use *NAME_NAME_mel* and *ADDRESS_ADDRESS_mel* feature to reduce the false negatives.
- The trigger we added is as shown below :

```
neg_trigger = mg.MatchTrigger()
neg_trigger.add_cond_rule('NAME_NAME_mel(ltuple, rtuple) > 0.85',feat_table)
neg_trigger.add_cond_rule('ADDRESS_ADDRESS_mel(ltuple, rtuple) > 0.9',feat_table)
neg_trigger.add_cond_status(False)
neg_trigger.add_action(0))
```

- We obtained the results as shown in Table 3.1 after applying the matcher and trigger on *U* and *V*.
- After running the trigger on *U* and *V*, we decided to run the matcher and trigger on the development set using 10 fold cross validation and obtained the results as shown in Table 3.2.

Matcher	Set	Precision	Recall	F1 Score	False Pos	False Neg
Random Forests	Fit on U Predict on V	100.00	96.15	98.04	0	1
RF + Trigger	Fit on U Predict on V	100.00	88.46	93.88	0	3

Table 3.1: Precision, Recall and F1 measures (in percentage) using Cross Validation on V. False Positives and False Negatives of Prediction on V.

Metric	Number of Folds	Mean Score
Precision	10	100.00
Recall	10	94.00
F1 Score	10	97.00

Table 3.2: Precision, Recall and F1 measures (in percentage) using matcher + trigger and 10 fold Cross Validation on I.

- Since the trigger did not help in decreasing the number of false negatives, we decided to debug the results obtained after running matcher and trigger on *V*.
- As there was no debugger in Magellan for output obtained after the trigger, to identify the false negative tuples, we wrote a python script which outputs the feature vector and the IDs of the false negative and false positive tuples. The python script we used is as shown below :

```
#!/usr/python
import csv
import os
import io
from _sqlite3 import Row

def getFalsePositives(filename):
    count = 0
    print "False Positives"
    with open(filename, 'r') as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            if(len(row)>1):
                length = len(row)
            else:
                continue
            if (row[length-2] == "gold"):
                print row
                continue
            if(int(row[length-2]) == 0 and int(row[length-1]) == 1):
```

```
                print ro
                count += 1
    return count

def getFalseNegatives(filename):
    count = 0
    print "False Negatives"
    with open(filename, 'r') as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            if(len(row)>1):
                length = len(row)
            else:
                continue
            if (row[length-2] == "gold"):
                print row
                continue
            if(int(row[length-2]) == 1 and int(row[length-1]) == 0):
                print row
                count += 1
    return count

if __name__ == '__main__':

    filename = "matches_output.csv"
    #getFalsePositives(filename)
    print "FalsePositives: " + str(getFalsePositives(filename))
    print "FalseNegatives: " + str(getFalseNegatives(filename))
```

- We identified the false negatives tuples as shown below :

```
Blowin' Smoke Barbeque (608) 215-0069 1336 Montondon Avenue, Waunakee, WI
Blowin' Smoke BBQ (608) 215-0069 1336 Montondon Ave, Waunakee, WI

The Taco Shop (608) 250-8226 604 University Ave, Madison, WI
The Taco Bros (608) 422-5075 604 E University Ave, Madison, WI

The Buena Vista   (415) 474-5044 2765 Hyde Street, San Francisco, CA
Buena Vista Cafe  (415) 474-5044 2765 Hyde St, San Francisco, CA
```

- We decided to now run the matcher with reduced feature set.

4 Third Iteration

Since application of Triggers did not give the expected results, we decided to modify the feature set.

- We started removing inappropriate features based on the properties of the function used to arrive at the feature value.
- For Example : We decided to remove all set based similarity scores, such as Jaccard, from the feature vector for PHONE_NUMBER attribute. Similarly we decided to remove character based similarity scores, such as Levenshtein distance, for ADDRESS attribute.
- The final feature set that we used to train the matchers on I is as follows :

```
NAME_NAME_jac_qgm_3_qgm_3
NAME_NAME_cos_dlm_dc0_dlm_dc0
NAME_NAME_jac_dlm_dc0_dlm_dc0
NAME_NAME_lev
PHONENUMBER_PHONENUMBER_mel
PHONENUMBER_PHONENUMBER_lev
PHONENUMBER_PHONENUMBER_nmw
PHONENUMBER_PHONENUMBER_sw
PHONENUMBER_PHONENUMBER_swg
ADDRESS_ADDRESS_jac_qgm_3_qgm_3
ADDRESS_ADDRESS_cos_dlm_dc0_dlm_dc0
ADDRESS_ADDRESS_lev
```

- When the above feature set was used to train on I all the six matcher algorithms the following Precision, Recall and F1 measures were obtained :

Matcher	Precision	Recall	F1 Score
Decision Tree	97.59	96.49	96.59
Random Forests	96.90	96.56	96.59
Support Vector Machine	80.00	7.07	12.89
Naive bayes	95.9	100.00	97.88
Logistic Regression	98.57	90.46	94.10
Linear Regression	95.66	97.59	96.42

Table 4.1: Precision, Recall and F1 measures (in percentage) of prediciton on I with modified Feature Set in third iteration.

- After examining the values in the Table 4.1 we decided to use *Naive Bayes* Matcher henceforth.
- The following results were obtained when the Naive Bayes matcher was used on U and V

Matcher	Set	Precision	Recall	F1 Score	False Pos	False Neg
Naive Bayes	Fit on U Predict on V	100.00	100.00	100.00	0	0

Table 4.2: Precision, Recall and F1 measures (in percentage) using Cross Validation on V. False Positives and False Negatives of Prediction on V.

Metric	Number of Folds	Mean Score
Precision	10	96.87
Recall	10	98.89
F1 Score	10	97.82

Table 4.3: Precision, Recall and F1 measures (in percentage) using matcher + trigger and 10 fold Cross Validation on I.

- Since the values for Precision and Recall was 100% on U and V we decided to use the Naive Bayes matcher on the entire development set I to check its performance on I.
- The recall and precision we obtained is included in the Table 4.3
- Since the results on I was better than the other two approaches in the previous iterations, we decided to stop here and use this matcher on the evaluation set J
- The final results we obtained on the evaluation set are as follows :

Matcher	Set	Precision	Recall	F1 Score	False Pos	False Neg
Naive Bayes	Fit on I Predict on J	97.44	97.44	97.44	1	1

Table 4.4: Precision, Recall and F1 measures (in percentage) False Positives and False Negatives of Prediction on J.

5 TIME AND STAGES

Stage	Tasks	Time Taken(min)
Preparation of Golden Data	1. Resampling to maintain appropriate ratio of matches and non-matches on V.	30
	2. Labelling of Sample.	180
Iteration 1	1. Finding Best Matcher.	10
	2. Debugging on U and V.	20
	3. Predict on J.	5
Iteration 2	1. Examine feature vector values to arrive at rules for trigger.	60
	2. Write rules for Trigger.	30
	3. Debugging the Trigger to improve Recall.	120
	4. Compare results obtained by applying Matcher and Matcher + Trigger on J.	20
Iteration 3	1. Examine feature vector values for False Positives and False Negatives.	20
	2. Eliminating inappropriate features.	60
	3. Running all six Matchers on I with new Feature Set to Select Best Matcher.	10
	4. Debugged on U and V	10
	5. Record results by predicting on J.	10
Iteration 4	1. Analysing False Positive and False Negative tuples	30
	2. Cleaning Golden Data and repeating Iteration 3	30
	3. Record Precision, Recall and F1 score values	5
	4. Store Final Matches	5
	Understanding how to use Magellan for Matching	120
Total Time		775

Table 5.1: Time Estimate for each of the stages