Project Stage II

Information Extraction from Text Documents

Team

Anna Chang
Sripradha Karkala
Simmi Pateriya

Text Type

The 300 text documents are real estate listings that were scraped from free-classifieds-usa.com using WebHarvy. They include homes of various sizes and from multiple regions of the US. The WebHarvy data was stored in *.csv format, which we extracted to *.txt using this script.

Entity Extracted

We extracted city names from the files and we labelled them using <city></city> tags. Below are are a few examples:

```
<city>Chester</city>, Ohio, United States

Great Location in <city>Branson</city>!!

18164 Fallatin Loop, <city>Beaverton</city>, OR 97007
```

Features

We extracted eight binary features:

Feature	Example
Is the first letter uppercase?	San Diego
Is the string followed by a state?	Chattanooga, Tennessee
Is the string proceed by “City:”?	City: Boise, Idaho
Is there a comma after the string?	Fairfax, VA
Is there a comma before the string?	21 Chester Lane, Princeton, NJ
Does “in” proceed the string?	3 BD in Trenton!
Does “city” follow the string?	New York City
Is the string in all caps?	AWESOME HOUSE IN MADISON!!

Train/Test Distribution

Train (Set I)
200 documents were used for training with 467 mentions.

Test (Set J)
100 documents were used for testing with 237 mentions.

Classifier M (Cross Validation on Train/Set I)

We compared five classifiers: Decision Tree, Random Forest, SVM, Logistic Regression, and Linear Regression. The complete results can be viewed here.

For Linear Regression, we applied a threshold of 0.5 to determine if a string was classified as a city or non-city. If the prediction was greater than 0.5, we classified the string as a city. Otherwise it was classified as a non-city.

After performing cross-validation, our best performing classifiers were the Decision Tree and Random Forest Classifier:

	Precision	Recall	F1
Decision Tree	0.945	0.754	0.839
Random Forest	0.947	0.730	0.823

We chose to use the Decision Tree for our final classifier because the precision was similar to the Random Forest, but the recall and F1 were higher.

Classifier X (Final Classifier on Test/Set J)

Below are the results of the Decision Tree when applied to Test Set J:

	Precision	Recall	F1
Decision Tree	0.92	0.78	0.84

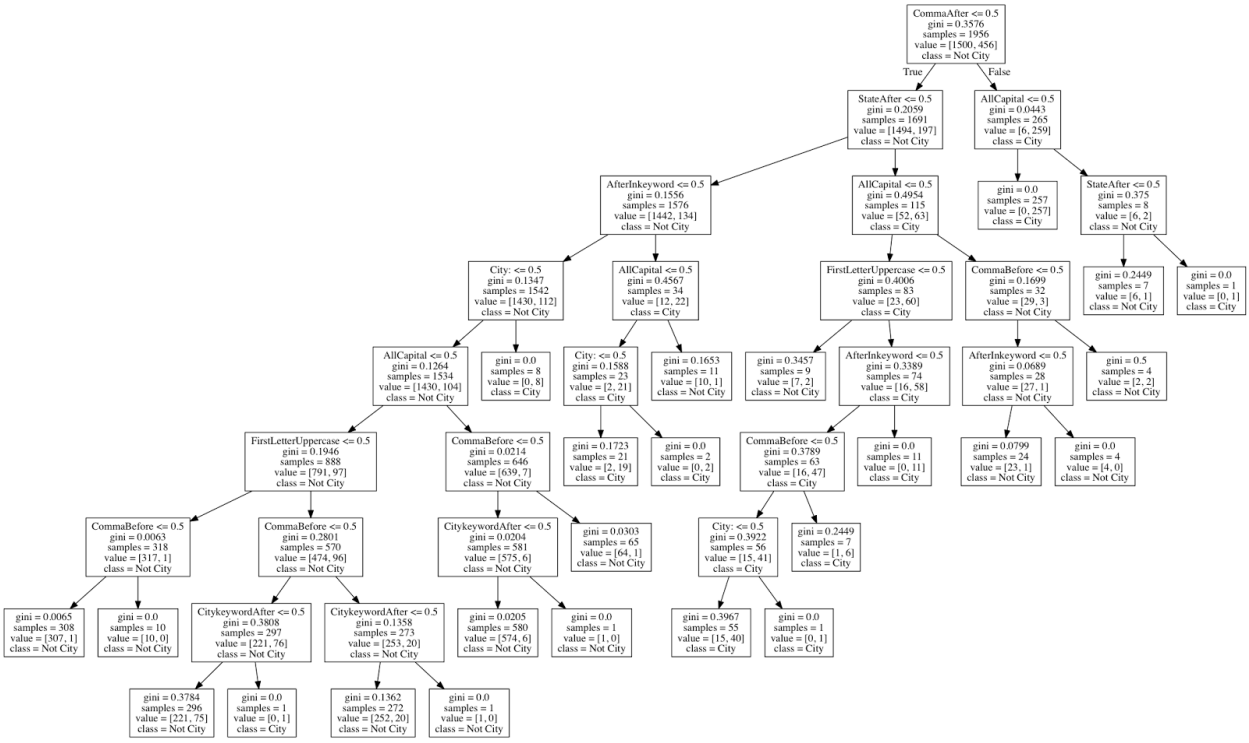
Some Final Thoughts

In Weka, we simulated the results for a pruned decision tree (one with four levels), and we saw that this increase precision to ~94% but reduced recall to 73.5%. Visualizations for our decision trees from scikit-learn and Weka are on the following pages.

Our classifier achieved the target precision and recall so we didn't do any rule-based post-processing.

Decision Tree Visualizations

Scikit-learn



Weka

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

CommaAfter = 1
|   AllCapital = 1
|   |   StateAfter = 1: 1 (2.0)
|   |   StateAfter = 0: 0 (4.0)
|   AllCapital = 0: 1 (136.0)
CommaAfter = 0
|   StateAfter = 1
|   |   AllCapital = 1: 0 (17.0/2.0)
|   |   AllCapital = 0
|   |   |   FirstLetterUppercase = 1: 1 (29.0/6.0)
|   |   |   FirstLetterUppercase = 0: 0 (12.0/1.0)
|   StateAfter = 0
|   |   City: = 1: 1 (4.0)
|   |   City: = 0: 0 (1522.0/58.0)

Number of Leaves  :     8

Size of the tree :      15


Time taken to build model: 0.04 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1655               95.8864 %
Incorrectly Classified Instances        71                4.1136 %
Kappa statistic                          0.8009
Mean absolute error                      0.0741
Root mean squared error                  0.1937
Relative absolute error                 32.5235 %
Root relative squared error             57.4247 %
Total Number of Instances             1726

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.735    0.007    0.938      0.735    0.824      0.809    0.861     0.800     1
                 0.993    0.265    0.961      0.993    0.977      0.809    0.861     0.958     0
Weighted Avg.    0.959    0.232    0.958      0.959    0.957      0.809    0.861     0.938

    Confusion Matrix
```