6. Automative review analyzer

Bookmark this page

Now that you have verified the correctness of your implementations, you are ready to tackle the main task of this project: building a classifier that labels reviews as positive or negative using text-based features and the linear classifiers that you implemented in the previous section!

The Data

The data consists of several reviews, each of which has been labeled with -1 or $+1$, corresponding to a negative or positive review, respectively. The original data has been split into four files:

- `reviews_train.tsv` (4000 examples)
- `reviews_validation.tsv` (500 examples)
- `reviews_test.tsv` (500 examples)

To get a feel for how the data looks, we suggest first opening the files with a text editor, spreadsheet program, or other scientific software package (like pandas).

Translating reviews to feature vectors

We will convert review texts into feature vectors using a **bag of words** approach. We start by compiling all the words that appear in a training set of reviews into a **dictionary** , thereby producing a list of d unique words.

We can then transform each of the reviews into a feature vector of length d by setting the i^{th} coordinate of the feature vector to 1 if the i^{th} word in the dictionary appears in the review, or 0 otherwise. For instance, consider two simple documents “Mary loves apples" and “Red apples". In this case, the dictionary is the set $\{\text{Mary}; \text{loves}; \text{apples}; \text{red}\}$, and the documents are represented as $(1; 1; 1; 0)$ and $(0; 0; 1; 1)$.

A bag of words model can be easily expanded to include phrases of length m. A **unigram** model is the case for which $m = 1$. In the example, the unigram dictionary would be $(\text{Mary}; \text{loves}; \text{apples}; \text{red})$. In the **bigram** case, $m = 2$, the dictionary is $(\text{Mary loves}; \text{loves apples}; \text{Red apples})$, and representations for each sample are $(1; 1; 0)$, $(0; 0; 1)$. In this section, you will only use the unigram word features. These functions are already implemented for you in the `bag of words` function.

In `utils.py`, we have supplied you with the `load data` function, which can be used to read the `.tsv` files and returns the labels and texts. We have also supplied you with the `bag_of_words` function in `project1.py`, which takes the raw data and returns dictionary of unigram words. The resulting dictionary is an input to `extract_bow_feature_vectors` which computes a feature matrix of ones and zeros that can be used as the input for the classification algorithms. Using the feature matrix and your implementation of learning algorithms from before, you will be able to compute θ and θ_0 .

Discussion

Hide Discussion

Topic: Unit 1 Linear Classifiers and Generalizations (2 weeks):Project 1: Automatic Review Analyzer / 6. Automative review analyzer

Add a Post

Show all posts by recent activity

- Opening the .tsv's in Pandas — 1
  To open the reviews files in pandas, the following line would do the trick - train = pd.read_csv('.../reviews_test.tsv',sep='\t',encoding='ISO-...
- When is the recitation due? — 3
  In the course overview I see "Unit 1 Linear Classifiers and Generalizations (2 weeks) - Recitation 1: Tuning the Regularization Hyperpara...
- linear separability of higher dimensional feature vectors. — 1
  when forming feature vectors like using a bag of words approach as above. how can we be sure they are linearly separable or what if th...
- Encoding and naming issue

Encoding and naming issue
The validation file name is "reviews_val.tsv", not "reviews_validation.tsv". Also, I think the encoding of the file is windows-1252.
1

? NameError: name 'csv' is not defined

< Previous | Next >

/