2. Naive Bayes Classification

Machine Learning and Real-world Data (MLRD)

Simone Teufel

Last session: we used a sentiment lexicon for sentiment classification

- Movie review sentiment classification was based on a lexicon, ie., lexicographers' intuition.
- Possible problems with using a lexicon:
  - required many hours of human labour to build
  - is limited to the words the humans decided to include (intuition can be limited)
  - is static and cannot react to language change (*terrific*!, *sick*!)

Today we will build a machine learning classifier that makes decisions based on the data that it's been exposed to.

What is Machine Learning?

- a program that learns from data.
- a program that adapts after having been exposed to new data.
- a program that learns implicitly from data.
- a program that learns without explicit programming.

A Machine Learning approach to sentiment classification

- The sentiment lexicon approach relied on a fixed, predefined list of words.
- The list was decided independently from our data before the experiment
- Instead we want to use machine learning to find out which words (out of all words in our data) express sentiment

Classification based on observations

First some terminology:

- features are easily (automatically) observable properties of the data.
- In our case the features of a movie review will be the words they contain.
- classes are the meaningful labels associated with the data, which are not easily automatically observable.
- In our case the classes are the two sentiments: POS and NEG.
- Classification then is function that maps from features to a target class.
- In our case, from the words in a review to a sentiment.

Probabilistic classifiers provide a distribution over classes

- Given a set of input features a probabilistic classifier returns the probability of each class.
- That is, for a set of observed features O and classes $c_1 ... c_m \in C$ gives $P(c_i|O)$ for all $c_i \in C$
- For us O is the set all the words in a review $\{w_1, w_2, ..., w_n\}$ where w_i is the ith word in the review, $C = \{\text{POS}, \text{NEG}\}$
- We get: $P(\text{POS}|w_1, w_2, ..., w_n)$ and $P(\text{NEG}|w_1, w_2, ..., w_n)$
- We can decide on a single class by choosing the one with the highest probability given the features:

$$\hat{c} = \operatorname*{argmax}_{c \in C} P(c|O)$$

Today we will build a Naive Bayes Classifier

- Naive Bayes classifiers are simple probabilistic classifiers based on applying Bayes' theorem.

Bayes Theorem:

$$P(c|O) = \frac{P(c)P(O|c)}{P(O)}$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c|O) = \underset{c \in C}{\operatorname{argmax}} \frac{P(c)P(O|c)}{P(O)} = \underset{c \in C}{\operatorname{argmax}} P(c)P(O|c)$$

- We can remove $P(O)$ because it will be constant during a given classification and not affect the result of argmax

Naive Bayes classifiers assume feature independence

$$c_{NB} = \operatorname*{argmax}_{c \in C} P(c|O) = \operatorname*{argmax}_{c \in C} \frac{P(c)P(O|c)}{P(O)} = \operatorname*{argmax}_{c \in C} P(c)P(O|c)$$

- For us $P(O|c) = P(w_1, w_2, ..., w_n|c)$
- Naive Bayes makes a strong (naive) independence assumption between the observed features.

$$P(O|c) = P(w_1, w_2, ..., w_n|c) \approx P(w_1|c) \times P(w_2|c) \times \cdots \times P(w_n|c)$$

so then:

$$c_{NB} = \operatorname*{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(w_i|c)$$

The probabilities we need are derived during training

$$c_{NB} = \operatorname*{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(w_i|c)$$

- In the training phase, we collect whatever information is needed to calculate $P(w_i|c)$ and $P(c)$.
- In the testing phase, we apply the above formula to derive c_{NB}, the classifier's decision.
- This is supervised ML because you use information about the classes during training.

The distinction between testing and training

- A machine learning algorithm has two phases: training and testing.
- Training: the process of making observations about some known data set
- In supervised machine learning you use the classes that come with the data in the training phrase
- Testing: the process of applying the knowledge obtained in the training stage to some new, unseen data
- We never test on data that we trained a system on

Task 2: Step 0 – Split the dataset from Task 1

- From last time, you have 1800 reviews which you used for evaluation.
- We now perform a data split into 200 for this week's testing and 1600 for training.
- You will compare the performance of the NB classifier you build today with the sentiment lexicon classifier.
- i.e. the NB classifier and the sentiment lexicon classifier will be evaluated on the same 200 reviews.
- Preview: There exist a further 200 reviews (bringing the total to 2000) that you will use for more formal testing and evaluation in a subsequent session.

Task 2: Step 1 – Parameter estimation

- Write code that estimates $P(w_i|c)$ and $P(c)$ using the training data.

Maximum likelihood estimation (MLE) is a method of estimating the parameters of a statistical model given observations

$$\hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)}$$

where $count(w_i, c)$ is number of times w_i occurs with class c and V is vocabulary of all distinct words.

$$\hat{P}(c) = \frac{N_c}{N_{rev}}$$

where N_c is number of reviews with class c and N_{rev} is total number of reviews

$$\hat{P}(w_i|c) \approx P(w_i|c) \text{ and } \hat{P}(c) \approx P(c)$$

Task 2: Step 2 – Classification

In practice we use logs:

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}}\ logP(c) + \sum_{i=1}^{n} logP(w_i|c)$$

Problems you will notice:

- A certain word may not have occurred together with one class
- Understand why this is a problem
- Work out what you could do to deal with it (there is more than one thing you could do)

Task 2: Step 3 – Smoothing

Add-one (Laplace) smoothing is the simplest form of smoothing:

$$\hat{P}(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w \in V}(count(w, c) + 1)} = \frac{count(w_i, c) + 1}{(\sum_{w \in V} count(w, c)) + |V|}$$

where V is vocabulary of all distinct words

See handbook and further reading:
`https://web.stanford.edu/~jurafsky/slp3/4.pdf`

Demonstrator Session today

- Currently, ticks are voluntary, but passing pretester is mandatory.
- Use the blackboard to announce that you are ready to be ticked
- Move on to today's tick as soon as you can.
- There is a starred tick available today (in "Further Notes on Naive Bayes"); strictly voluntary
- Good luck and have fun!
- Join me and the demonstrators in the lab at 2:30.