Department of Computing

Imperial College of Science, Technology and Medicine

NLP and Classificaiton

Intro to how one would develop classification algorithms for natural language processing

Author: Anton Zhitomirsky

Contents

1 NLP Classification tasks

Definition 1.1 (Classification).

$$\hat{y} = \arg\max_y P(y|x) \tag{1}$$

Predicting which 'class' an observation belongs to

A Model produces a score (logit), a sigmoid makes this between 0 and 1, and 0.5 is our decision boundary. In multi-class classification we then use softmax.

1.1 Natural Language Inference

a model is presented with a pair of sentences and must classify the relationship between their meanings. For example in the MultiNLI corpus, pairs of sentences are given one of 3 labels: entails, contradicts and neutral. These labels describe a relationship between the meaning of the first sentence (the premise) and the meaning of the second sentence (the hypothesis) [1]. Here are representative examples of each class from the corpus:

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping
- **Entailment**: If the hypothesis is implied by the premise
- **Contradiction**: If the hypothesis contradicts the premise (in this example, the hypothesis is contradicting)
- **Neutral**: otherwise (neither is necessarily true)

2 Naive Bayes

(Generative Algorithm)

Definition 2.1 (Bayes Rule).

$$\underbrace{P(y|x)}_{\text{Posterior}} = \frac{\overbrace{P(x|y)}^{\text{Likelihood}}\ \overbrace{P(y)}^{\text{Prior}}}{\underbrace{P(x)}_{\text{Evidence}}}$$

Since $P(x)$ won't change for different classes

> ☞ why? - because I believe it is the training data. The training data in a model won't change unless it is retrained, in which case the probabilities will change but the classification outcome shouldn't change because for each x we are always dividing by the same $P(x)$.

$$\hat{y} = \arg\max_y P(y|x) = \arg\max_y P(x|y)P(y)$$

We can further make an assumption that x is a set of features $x_1, \ldots, x_I$ that are independent:

Definition 2.2 (Naiive Bayes Classifier).

$$\hat{y} = \arg\max_y \overbrace{P(x_1, \ldots, x_I|y)}^{P(x_1|y)\cdot\ldots\cdot P(x_I|y)} P(y) = \arg\max_y P(y) \prod_{i=1}^{I} P(x_i|y)$$

2.1 Bag of Words Input Representations

Raw input is transformed into a numerical representation - i.e. each input x is represented by a feature vector by using a Bag of Words approach (count of each word in the input) “This was another good movie for holiday watchers. There was a nice little twist at the end”

Collect statistics from our training data (find what $P(y|x)$ is) after performing limited data-preprocessing.

Training corpus	good	movie	bad	class
another **good movie** for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	1	1	0	+
it seems like just about everybody has made a christmas carol movie . others are just **bad** and the time period seems to be perfect .	0	0	1	+
if you 're looking for the same feel **good** one but in a new setting , this one 's for you .	1	0	0	+
this is a first for me , i didn 't like this **movie** . it was really **bad** .	0	1	1	-
it was **good** but the christmas carol by dickens was emotionally moving .	1	0	0	- 26

Figure 1: Example of training corpus. Here, we are only concerned with the words ‘good’, ‘movie’, and ‘bad’ for classes ‘+’ and ‘-’

$$P(y) \rightarrow P(+) = \frac{3}{5}, \quad P(-) = \frac{2}{5}$$

$$P(good|+) = \frac{2}{4} \qquad \text{i.e. } \frac{\text{frequency of word for class}}{\text{total count of words for this class}}$$

2.2 One smoothed Naive Bayes Classifier

this introduces the problem that one of our probabilities could be zero; therefore, adjust naive bayes classifier with ‘Add-one smoothing’

Definition 2.3 (Add-one smoothing)**.**

$$P(x_i|y) = \frac{count(x_i, y) + 1}{\sum_{x \in V}(count(x, y) + 1)} = \frac{count(x_i, y) + 1}{(\sum_{x \in V} count(x, y)) + |V|}$$

$$P(good|+) = \frac{2+1}{4+3} = \frac{3}{7} \qquad P(good|-) = \frac{1+1}{3+3} = \frac{2}{6}$$
$$P(movie|+) = \frac{1+1}{4+3} = \frac{2}{7} \qquad P(movie|-) = \frac{1+1}{3+3} = \frac{2}{6}$$
$$P(bad|+) = \frac{1+1}{4+3} = \frac{2}{7} \qquad P(bad|-) = \frac{1+1}{3+3} = \frac{2}{6}$$

Therefore for a test example: “Not as **good** as the old **movie**, rather **bad**” we use Equation 2.2 to get $P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3} = 0.021$ and $P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3} = 0.014$. So final outcome of the model is $+$. However, this sentence is clearly negative, yet we classify as positive.

2.3 Binary Naive Bayes

Within binary naive bayes, we only consider if a feature is present, rather than considering every time it occurs. Note, this means re-calculating the conditional probabilities from the training data.

E.g. "Not as **good** as the old **movie**, rather **bad movie**" we have $P(+)P(x|+) = \frac{3}{5} \times \frac{3\times2\times2}{7^3} = 0.021$ and $P(-)P(x|-) = \frac{2}{5} \times \frac{2\times2\times2}{6^3} = 0.014$. If we were still operating under the same formula in Equation 2.2 then we would have iterated over the label 'movie' twice, and thus multiplied by $P(good|+)$ twice (in the positive case).

2.4 Controlling for negation

We append 'Not_' after any logical negation (e..g n't, not, no, never) until the next punctuation mark.

"I didn't like the movie, but it was better than Top Gun" becomes "I didn't NOT_like NOT_the NOT_movie, but it was better than Top Gun"

2.5 Problems

- Conditional independence assumption
- Features considered equally important
- Context of words not taken into account
- New words (not seen at training) cannot be used

2.6 Summary

Very quick to train, Some evidence it works well on very small datasets.

3 Logistic Regression

(Discriminative Algorithm) because we directly learn $P(Y|X)$ since we don't care about $P(Y)$ or $P(X)$. They learn the input feauters most useful to discriminate between the different classes, without considering the likelihood of the input itself.

$$
\begin{aligned}
y(x) = g(z) &= \frac{1}{1+e^{-z}} \\
z &= w \cdot x + b \\
P(y=1) &= \frac{1}{1+e^{-(w\cdot x+b)}}
\end{aligned}
$$

where w is how important an input feature is to the classification decision and we have a threshold at 0.5 for decision making.

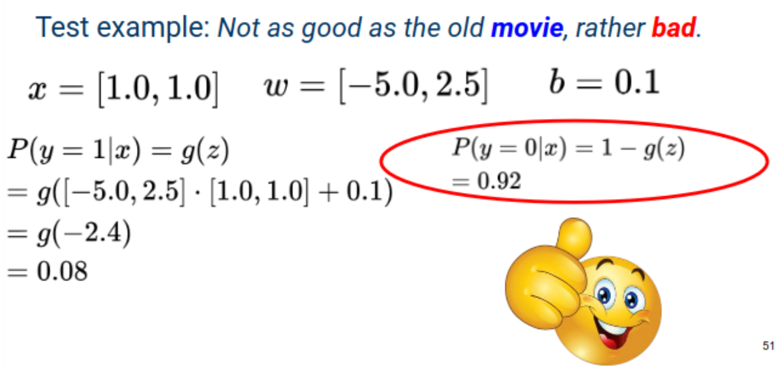


Figure 2: Here the vector x is made up of the collection of words, here, $x_1 = count(movie) \wedge x_2 = count(bad)$

If we don't have the w, then we learn parameters to make the model predictions close to our labels by using a loss function (to measure the distance between the true and predicted labels) and optimization algorithm (to minimize the function usually through gradient descent.)

Definition 3.1 (Logistic Regression).

$$H(P, Q) = -\sum_i P(y_i) \log Q(y_i)$$

3.1 Multiple Classes

In the case where we have words

$$\begin{aligned} x_1(bad) &= 1 \\ x_2(good) &= 1 \\ x_3(and) &= 2 \end{aligned}$$

With sentiment analysis over 3 classes (+, -, and neutral), weights and bias are learnt per class. Then, they use a softmax function over the result of finding z_i.

$$y = g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

3.2 Summary

Logistic Regression considers the importance of feautres, so is better (than Naive Bayes) at dealing with correlated feautres, also better (than Naive Bayes) with larger datasets.

4 Neural Networks (NNs)

Definition 4.1 (Linear Layer).

$$z = w \cdot x + b = \sum_{i=0}^{I} w_i x_i + b$$

Definition 4.2 (Non-linear activation function).

$$y = g(z)$$

Definition 4.3 (Fully-connected layers).

$$FFN(x) = (g^2(g^1(xW^1 + b^1))W^2 + b^2)W^3 + b^3$$

The neural networks allow for automatically learning dense feature representations (or alternatively pre-trained dense representations), not one-hot encodings.

4.1 Document Representation

How to get a document representation of sentence of fixed dimensionality?

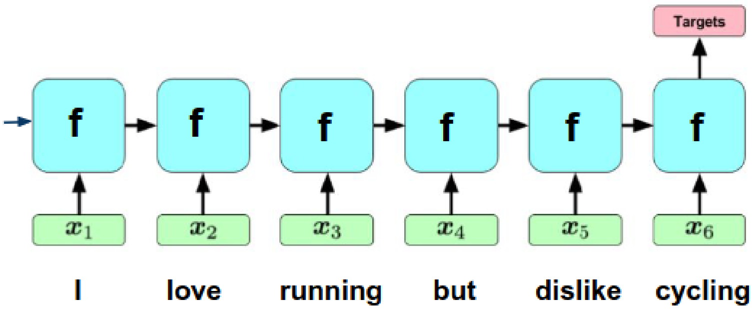
Average of sentence - bad idea:

Model architecutre fixed to sentence length size, model wieghts learnt for specific word positions

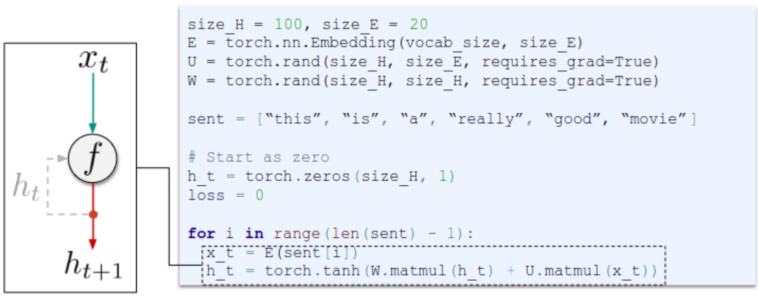
4.2 Neural Networks

Automatically learned features; flexibility to fit highly complex relationships in data, but they require more data to learn more complex patterns.

5 Recurrent neural networks (RNNs)



Natural language data is made up of sequences, so its natural to represent in a RNN; the value of a unit depends on own previous outputs - the last hidden state is the input to the output layer. The words are input into the model after an embedding layer which vecotrizes the ipnut.



5.1 Vanishing gradient problem

The model is less able to learn from earlier inputs (Tanh derivatives are between 0 and 1) (Sigmoid derivatives are between 0 and 0.25) - Gradient for earlier layers involves repeated multiplication of the same matrix W - depending on the dominant eignevalue this can cause gradients to either ‘vanish’ or ‘explode’

> ☞ RNNs perform better when you need to understand longer range dependencies

6 CNNs

CNNs are composed of a series of convolution layers, pooling layers and fully connected layers. Convolutional layers Detect important patterns in the inputs. Pooling layers Reduce dimensionality of features and transform them into a fixed-size. Fully connected layers Train weights of learned representation for a specific task.

We can stack multiple filters on-top of eachother also.

> ☞ CNNs can perform well if the task involves key phrase recognition.

7 Accuracy and F1

Definition 7.1 (accuracy).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Definition 7.2 (f1-measure).

$$f1 = 2 \times \frac{precision \times recall}{precision + recall} = \frac{TP}{TP + 0.5(FP + FN)}$$

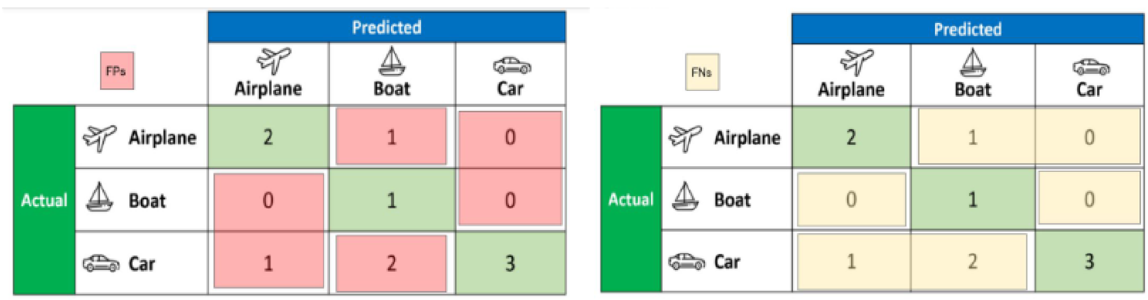
7.1 Macro averaging

averaging of each class F1 scores: increases the emphasis on less frequent classes

7.2 Micro averaging

TPs, TNs, FNs. FPs are summed across each class.

7.2.1 Microaveraged F1



FPs		Predicted: Airplane	Predicted: Boat	Predicted: Car
Actual	Airplane	2	1	0
Actual	Boat	0	1	0
Actual	Car	1	2	3

FNs		Predicted: Airplane	Predicted: Boat	Predicted: Car
Actual	Airplane	2	1	0
Actual	Boat	0	1	0
Actual	Car	1	2	3

$$\frac{\sum_i^C TP_i}{\sum_i^C TP_i + 0.5(\sum_i^C FP_i + \sum_i^C FN_i)} = \frac{\sum_i^C TP_i}{|Dataset|} = Accuracy$$

References

[1] Dan Jurafsky and James H. Martin. *Speech and Language Processing*. 2023. URL: https://web.stanford.edu/~jurafsky/slp3/.