Mark Kim

Professor Anagha Kulkarni

CSC 620

December 9, 2022

I Regular Expressions

Regular expression (RegEx) is a domain specific language that allows us to search for lexical patterns in a corpus. By applying such expressions, we can normalize text by removing stop words, punctuation, etc. This tool, however, can be indiscriminate in its application. Nevertheless, it can be a powerful tool for finding (and/or replacing) text according to static rules set by the user.

II Edit Distance

Edit distance is a method of quantifying similarities or dissimilarities between text. If two texts have a low edit distance, they are highly similar, and high edit distance means the texts have low similarity. *Minimum* edit distance simply quantifies the minimum number of editing operations it takes to convert one string to the next. These edit operations consist of *insertion*, *deletion*, and *substitution*. The Levenshtein formulation of operations applies a cost for each operation as follows:

- Insertion: 1
- Deletion: 1
- Substitution: 2

The operations alone does not allow us to find the minimum distance. One must also take into account *alignment* to find the minimum distance.

III N-gram based Language Modeling

N-gram based language modeling is a method of modeling that uses the counts of words in a corpus to determine the probability that a particular word will occur. This type of modeling is based off of the Chain Rule of Probability. This means that the probability of a particular sequence of words is simply the product of the probabilities of each word that occurs in the corpus given the words preceding it:

$$P(w_1, w_2, \cdots, w_n) = \prod_i P(w_i \mid w_1 w_2 \cdots w_{i-1}).$$

It is sufficient, however, to simplify these calculations as follows:

- Unigram Model: $P(w_1, w_2, \cdots, w_n) \approx \prod_i P(w_i)$
- Bigram Model: $P(w_1, w_2, \cdots, w_n) \approx \prod_i P(w_i \mid w_{i-1})$

IV Text Classification using Naïve Bayes

The Naïve Bayes method of text classification relies on the Bayes Rule which states

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)},$$

where c is the class and d is the document (the denominator is ignored for our use). Then our predicted class will be

$$\begin{aligned} C_{MAP} &= \operatorname*{argmax}_{c \in C} P(c \mid d) \\ &= \operatorname*{argmax}_{c \in C} P(d \mid c)P(c) \\ &= \operatorname*{argmax}_{c \in C} P(x_1, x_2, \cdots, x_n \mid c)P(c) \end{aligned}$$

Despite the fact that probabilities of features (which can be words, characters, bigrams, etc.) are not independent given a document class c, we can approximate the probabilities of those features (given a class c) as follows:

$$C_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{x \in X} P(x \mid c).$$

We need to find the Maximum Likelihood Estimates by using the relative frequencies in the data which can be calculated by

$$\hat{P}(x_i \mid c_j) = \frac{\operatorname{count}(x_i, c_j)}{\sum_{k=1}^{n} \operatorname{count}(x_k, c_j)}$$

Once trained, we use our argmax function to predict/assign a class label to the document we are trying to classify.

V Text Classification using Logistic Regression

Similar to Naïve Bayes, Logistic Regression uses a feature set for text classification and is also uses a supervised learning model. Logistic regression uses the statistical modelling technique called regression analysis. This method of analysis consists of two parts:

1. The Objective/Loss Function
2. Optimization of the Objective Function by using Stochastic Gradient Descent

The Objective Function tells us the loss (number of classification errors) when using a particular weight vector $\vec{w}$ and bias b when using the linear regression equation

$$\begin{aligned} z &= \vec{w} \cdot \vec{x} + b \\ &= \left(\sum_{i=1}^{n} w_i x_i \right) + b \end{aligned}$$

For a single class evaluation (class a/not class a), our MLE is the sigmoid function

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}.$$

For multiclass evaluation, we use the softmax function which essentially generalizes the sigmoid function with

$$[\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_k] = \mathrm{softmax}\left(\vec{Z}\right) = [\mathrm{softmax}\left(Z_1\right), \mathrm{softmax}\left(Z_2\right), \cdots, \mathrm{softmax}\left(Z_k\right),]$$

where

$$\hat{y}_i = P(y = i \mid x) = \mathrm{softmax}(Z_i) = \frac{e^{\vec{w}_i \cdot \vec{x} + b_i}}{\sum_{j=1}^{k} e^{\vec{w}_j \cdot \vec{x} + b_j}}, \quad 1 \leq i \leq k$$

which gives us a probability distribution of classes.

During the training step, we compare the ground truth labels with the MLE results with a Cross-Entropy Loss Function. This loss function is then optimized (minimized) via Gradient Descent (e.g. calculating the partial derivative, and using that to ”descend” towards a lower minimum and repeating).

VI Text Classification Evaluation

Evaluating text classification can be done using an *extrinsic*, or *intrinsic* approach. For the intrinsic case, we do a qualitative evaluation, whereas for the extrinsic case, we use the models to see which produces better results. Using intrinsic evaluation, labelled data is split between a training and test set. The model is trained on the training set, then the model's predictions for the test set are compared against the ground truth labels to see how well the predictions match. *Precision* and *recall* are the metrics used to evaluate the effectiveness of the classification model. Precision measures the percent of a true positive prediction that were actually that ground truth class:

$$P = \frac{tp}{(tp + fp)}, \quad \text{where } tp = \text{true positive, and } fp = \text{false positive.}$$

Whereas, recall measures the percent of a ground truth class that are predicted as that particular class:

$$R = \frac{tp}{(tp + fn)}, \quad \text{where } tp = \text{true positive, and } fp = \text{false negative.}$$

A combined metric called F-measure will assess the inherent tradeoff between precision and recall with

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}, \quad 0 \leq \beta \leq \infty.$$

VII Vector Semantics

VIII Feedforward Neural Networks

IX Sequence Processing using Recurrent Neural Networks

X Sequence Processing using Transformers