Journal of Machine Learning Research 1 (2019) 1-10 Submitted 9/19; Published 9/19

CSCI 447: Project 1

George Engel GeoEngel.z@gmail.com
Department of Engineering
Montana State University
Bozemane, MT 59715, USA

Troy Oster toster1011@gmail.com
Department of Engineering
Montana State University
Bozeman, MT 59715, USA

Dana Parker danaharmonparker@gmail.com
Department of Engineering
Montana State University
Bozeman, MT 59715, USA

Henry Soule hsoule427@gmail.com
Department of Engineering
Montana State University
Bozeman, MT 59715, USA

Editor: Engel et al.

Abstract

A brief, one paragraph abstract summarizing the results of the experiments

Keywords: Keywords, Go, Here

1. Introduction

Problem statement, including hypothesis

2. Problem Statement

3. The Algorithm

Per the instructions of the assignment, we implemented the naive Bayes algorithm on the five provided datasets. Given some example $x \in X$, where X is our dataset, naive Bayes predicts the correct class c of x by computing the probabilities of each possible classification for x. For class c, the probability is denoted as $P(c|a_1, a_2, ..., a_d)$ where a_k denotes one of d attribute values in x. To compute this probability for each class c, the probabilities of each attribute value are computed. For each attribute value a_k, we compute $P(c) * \prod_{i=0}^{d} P(a_i|c)$, where $P(c)$ is the probability of an attribute being classified as class c. The predict the correct class for x we compute $\underset{c \in C}{\operatorname{argmax}}\, P(c) * \prod_{i=0}^{d} P(a_i|c)$.

4. Our Approach

To properly implement naive Bayes on the 5 datasets, we first needed to properly separate and classify each data set. Each data set needed to be separated by class, and then the count and probability of each attribute value for each class needed to be computed. Our approach for this was to create, for each dataset, an associative array, where each possible classifier was a key. For each key, the corresponding values were each their own associative arrays, the keys of which were all the extant attribute values among that class. The values of each key was a set storing the count and probability of each attribute value.

To assess the performance of our class prediction algorithm, we implemented two loss functions: Precision/Recall, and Cross Entropy. The Precision/Recall loss function computes two values, known as precision and recall, to measure the performance of our class prediction algorithm. This loss function utilizes the amounts of true positive, false positive, and false negative classifications for each class. Precision and recall are are computed as follows:

$$Precision = \frac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i}{TP_i + FP_i}$$

$$Recall = \frac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i}{TP_i + FN_i}$$

TP_i denotes the number of true positive classifications for class i, FP_i denotes the number of false positive classifications for class i, and FN_i denotes the number of false negative classifications for class i. C is the set of all possible classes. Precision can be interpreted as a measure of how accurate true positive classifications, as it computes the fraction all positive classifications for a class that were truly positive. Recall measures accuracy among the values for each class that should have been positive, as it computes the fraction of all values that truly belonged to a class that were classified as positive.

5. Results

Acknowledgments

Acknowledgements go here.

Appendix A.

In this appendix we prove the following theorem from Section 6.2:
Theorem *Let u, v, w be discrete variables such that v, w do not co-occur with u (i.e., $u \neq 0 \;\Rightarrow\; v = w = 0$ in a given dataset $\mathcal{D}$). Let N_{v0}, N_{w0} be the number of data points for which $v = 0, w = 0$ respectively, and let I_{uv}, I_{uw} be the respective empirical mutual information values based on the sample $\mathcal{D}$. Then*

$$N_{v0} \;>\; N_{w0} \;\;\Rightarrow\;\; I_{uv} \;\leq\; I_{uw}$$

with equality only if u is identically 0. ■
Proof. We use the notation:

$$P_v(i) \;=\; \frac{N_v^i}{N}, \quad i \neq 0; \quad P_{v0} \;\equiv\; P_v(0) \;=\; 1 - \sum_{i \neq 0} P_v(i).$$

These values represent the (empirical) probabilities of v taking value $i \neq 0$ and 0 respectively. Entropies will be denoted by H. We aim to show that $\frac{\partial I_{uv}}{\partial P_{v0}} < 0$....

Remainder omitted in this sample. See http://www.jmlr.org/papers/ for full paper.