BIOL 5312 final project report part II

Nick McCloskey

Statistics and machine learning (ML) overlap methodologically, as both fields aim to extract information from patterns in data. The data for a supervised ML task are structured as feature vectors matched with a target label, and the main goal of learning about a population is prediction, as opposed to description or inference (unsupervised ML uses unlabeled data and aligns more in objective with descriptive statistics). In general, supervised ML models are generated as algorithms iterate through training data, compute the error between a prediction and the correct label with a loss function, and update model parameters based on that error. The accuracy of the model is then tested with either a withheld (unseen by the model) portion of the original data, or with another dataset by comparing its predictions to the correct labels for the feature vectors. When vetted as an accurate predictor, the model can be applied to real-world problems where feature vectors are available before or more feasibly than target labels. There are many types and subtypes of supervised ML algorithms, and labels can be categorical (classification tasks) or continuous (regression tasks). The method selected for this report was the naïve bayes (NB) classifier because it is relevant to topics covered in the course and has many potential applications in research and industry.

Because of its direct basis on Bayes' probability theorem (right), the NB algorithm in particular blurs the line between statistics and ML. This formula describes the probability of event A occurring given the occurrence of event B. Independence between events is assumed, and because one feature is assumed to have no effect on another, and also because the algorithm simplifies calculations for tractability, it is named "naïve" (1,2). The features contained in the vector X can be written as

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$X = (x_1, x_2, x_3, ..., x_n)$$

And Bayes' formula can be written in terms of X:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

Expanding with the chain rule,

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

The denominator does not change across examples, the above can be expressed as a proportionality.

$$P(y|x_1, ..., x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

The training for this classifier is very fast (and a bit different from the general description) because it requires only the calculation of the probability of each class and the probability of each class given each feature. The predictor functions by finding the probabilities of the features given each label, normalizing the values so the sum equals 1 (i.e. converting from proportional probabilities to

probabilities), and selecting label with the maximum probability as the prediction. Essentially, an argmax (across all y) operation is conducted on the above formula (3).

There are a few types of naïve bayes classifiers: multinomial NB, where features are frequencies, e.g. count of words in a text corpus; Bernoulli NB, whose input features are Boolean variables; and Gaussian, where features take on continuous values assumed to be drawn from gaussian distributions (1). The current project builds a Gaussian NB binary classifier to predict the potability of water from continuous features, and a Bernoulli NB multi-class classifier, to predict the diagnosis of a disease based on the presence of various conditions. Applications of NB classification to biomedical research include predicting the onset of Alzheimer's disease from genome data (4), the benignity/malignancy of breast cancer tumors from encrypted medical data (5), and the probabilities associated with diseases and symptoms from electronic medical records (6).

The R code is below, followed by a discussion.

```
Set up session
library(naivebayes)
library(tidyverse)
library(ggplot2)
library(psych)

Water potability, from (7)
wp<-read.csv("water_potability.csv")
View(wp)
wp<-drop_na(wp)
xtabs(~Potability,data=wp)
# Potability
# 0    1
# 1998 1278

str(wp)
'data.frame':     2011 obs. of  10 variables:
 $ ph          : num  8.32 9.09 5.58 10.22 8.64 ...
 $ Hardness    : num  214 181 188 248 203 ...
 $ Solids      : num  22018 17979 28749 28750 13672 ...
 $ Chloramines : num  8.06 6.55 7.54 7.51 4.56 ...
 $ Sulfate     : num  357 310 327 394 303 ...
 $ Conductivity  : num  363 398 280 284 475 ...
 $ Organic_carbon : num  18.4 11.6 8.4 13.8 12.4 ...
 $ Trihalomethanes: num  100.3 32 54.9 84.6 62.8 ...
 $ Turbidity   : num  4.63 4.08 2.56 2.67 4.4 ...
 $ Potability  : int  0 0 0 0 0 0 0 0 0 0 ...

Convert potability to factor
wp$Potability<-as.factor(wp$Potability)
```

Training and test data

```
set.seed(88)
ind<-sample(2,nrow(wp),replace=T,prob=c(0.75,0.25))
X_train<-wp[ind==1,1:9]
y_train<-as.factor(wp[ind==1,10])
X_test<-wp[ind==2,1:9]
y_test<-wp[ind==2,10]
```

Predictions and accuracy scores on training and test data

```
model<-gaussian_naive_bayes(X_train,y_train,usekernel=T)
ptrain<-predict(model,newdata=data.matrix(X_train),type="class")
ctrain<-sum(ptrain==y_train)
(acctrain<-ctrain/length(y_train))
```

[1] 0.6211221

```
ptest<-predict(model,newdata=data.matrix(X_test),type="class")
ctest<-sum(ptest==y_test)
(acctest<-ctest/length(y_test))
```

[1] 0.6512097

Disease prediction (8)

```
td<-read.csv("Training.csv")
td$prognosis<-as.factor(td$prognosis)
set.seed(88)
ind<-sample(2,nrow(td),replace=T,prob=c(0.75,0.25))
X_train<-td[ind==1,1:132]
y_train<-as.factor(td[ind==1,133])
X_test<-td[ind==2,1:132]
y_test<-td[ind==2,133]

model<-naive_bayes(X_train,y_train,usekernel=T)
```

Training data and accuracy

```
ptrain<-predict(model,X_train,type="class")
ctrain<-sum(ptrain==y_train)
(acctrain<-ctrain/length(y_train))
```

[1] 0.9991776

Test data and accuracy

```
ptest<-predict(model,X_test,type="class")
ctest<-sum(ptest==y_test)
(acctest<-ctest/length(y_test))
```

[1] 0.9976415

The gaussian naïve bayes classifier was trained on continuous features describing water, including pH, hardness, and conductivity to predict the binary class of potability, either potable (label 1) or not (label 0). Examples were randomly allocated in portions of 75% to training data and 25% to testing data. The accuracy of the model on the training data on which it was fit was only about 62%, and unexpectedly it performed slightly better at 65% for the test data. This is unexpected because generally models perform better on the data used to fit them, especially when overfitting (losing generalizability by working too hard to fit the training data) is a risk, but the test accuracy can be higher than training accuracy, when the underlying distributions across partitions are not equivalent (9). Neither of these accuracy scores are very good, however, in light of the fact that the accuracy of randomly guessing a binary outcome is 50%. Other models such as neural networks may be better suited to making predictions from this dataset.

The Bernoulli naïve bayes classifier was trained on binary features such as presence of acute liver failure, malaise, and back pain to predict one of 42 prognoses including hepatitis, arthritis, and acne. The training data file contained plenty of examples for both training (75%) and testing (25%). (The testing file included in this download was unlabeled, so it is not useful.) The model performed well on both training and testing data, with accuracies of 99.9% and 99.8% respectively. This task was much better suited to NB classification. There were also over twice as many examples available in total, and generally ML algorithms do better with more training data, in part because the greater variation in the data allows them to avoid overfitting.

In conclusion, naïve bayes classifiers can be implemented in R to make predictions about unlabeled data, but the classifiers are not ideal for all datasets, so in those cases other algorithms could be considered.

References

1. https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
2. https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/
3. https://www.geeksforgeeks.org/naive-bayes-classifiers/
4. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3128400/
5. https://pubmed.ncbi.nlm.nih.gov/30641309/
6. https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-2924-0#:~:text=The%20proposed%20na%C3%AFve%20Bayes%20classifier,a%20disease%20into%20the%20ontology.
7. https://www.kaggle.com/datasets/whenamancodes/water-pollution
8. https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning?select=Training.csv
9. https://www.kaggle.com/questions-and-answers/186681#:~:text=The%20test%20accuracy%20can%20be,the%20ones%20in%20train%20data.

Sources

10. https://www.r-bloggers.com/2021/04/naive-bayes-classification-in-r/
11. https://www.edureka.co/blog/naive-bayes-in-r/
12. https://statistics.laerd.com/statistical-guides/sphericity-statistical-guide.php
13. https://rpkgs.datanovia.com/rstatix/reference/anova_test.html
14. https://statistics.laerd.com/statistical-guides/sphericity-statistical-guide-2.php