Suns, Stars, Stripes and Crosses: Predicting Religion from Flags Using Bayesian Classifiers

Amanda Braun (acb780@uregina.ca) and Riley Herman (herman5r@uregina.ca)
University of Regina

Abstract

While the flag of a nation is a somewhat reliable indicator of the religion most practiced in a country, it is not a perfect indicator. This is the main interest in using a probabilistic algorithm to determine religion from a flag. We wrote an implementation of Naïve Bayesian Classifiers that predicts the religion of a country based on data about its flag. The Naïve Bayesian Classifier is determined using repeated applications of a part of Bayes' Theorem for describing probability based on prior probabilities. While the algorithm does have several assumptions that are made for this data set, the overall accuracy of this implementation was comparable to the (Sci-kit Learn, 2018) implementation.

Keywords: Bayesian classifier, prediction, machine learning, flags, religion

1.0 Introduction

The relationship between national flags and most commonly practiced religions domestically dates back to the origins of flags. The very first national flags were adapted from military use. While the exact time that flags were first used to represent countries, some of the earliest known uses of a national flag were Scotland's white Saltire cross upon a blue background from the early 800s (Bartram, G., 2001) and Japan's red circle upon a white background from the early 600s. (Dyer, H., 1909)

The Saltire cross, also known as the St. Andrew's cross, is named for the patron saint of Scotland. Legend has it that king Óengus II of the Picts in present day Scotland in 832 prayed to St. Andrew on the evening before a battle; the next morning, he and his troops saw the clouds form a Saltire upon a blue sky. While this may not be the exact adoption of the cloud-white Saltire on the sky-blue background for the nation of Scotland, it lays the foundation of the flag squarely in a religious context. (Bartram, G., 2001; Perrin, W. G., 1922)

In Japan's case, the story of the *Hinomaru* (officially known as the *Nisshōki*) probably originates in the 7th century, when the Emperor of Japan referred to his homeland as the land of the rising sun in a correspondence to the Emperor of Sui in modern day China (Dyer, H., 1909). As Japan lies east of China, the sun appears to come from the archipelago from the shores of the mainland. While there is not much record of a flag being designed based on this context, later in Japanese history the *Hinomaru* design is seen referring to this event. The flag was notably used in battle during the 13th century when the Mongols invaded Japan. A Buddhist priest named Nichiren gave a banner bearing this representation of the rising sun to the *shōgun* prior to the battle. (Itoh, M., 2003; Edgington, D. W., 2003)

The effect of religion on country flags is most apparent in states that have a national religion. Officially Christian nations including England, Denmark, Norway, and Greece all sport various types of crosses on their flags. Muslim nations including Algeria, Malaysia, Pakistan, and Turkey have the iconic crescent moon and star symbol. Buddhist nations such as Bhutan, Cambodia, and Sri Lanka have the Buddhist symbols of the Druk, Angkor Wat, and the Bo leaves respectively (Bates, R., 2007; Strangio, S., 2014; Government of Sri Lanka, n.d.). In addition, these countries have had their effect on countries that do not explicitly state a state religion; all of the Scandinavian countries have similar flags to the Norwegian cross; such it is known as a Nordic cross. (Harrington, K., 2003; Macgeorge, A., 1881; Dane P., 2008) Fascinatingly, the oldest flag in this group is the Danish flag from 1370 (Bjerg, H. C., 2006). While Denmark no longer has a state religion, the flag is a remnant from the time when they were a Christian nation. In a similar way, the Singaporean flag has the crescent moon and several stars from the time is was a state in the Muslim nation of Malaysia. (Lee, K. Y., 1998) Yet a crescent moon and stars does not a Muslim nation make; Singapore is approximately 14% Muslim but over 40% Buddhist, the largest religious demographic. (Singapore Statistics, 2015)

While the flag of a nation is a somewhat reliable indicator of the religion most practiced in a country, it is not a perfect indicator. This is the main interest in using a probabilistic algorithm to determine religion from a flag; while there appears to be a correlation due to causation, the points at which the algorithm is incorrect reveal fascinating traits and fascinating stories. In addition, it is revealing which traits correspond most closely to determining the religion of a country. This is the inspiration for this implementation. The classification method that was chosen uses Naïve Bayesian Classifiers. While there exist many different algorithms for classification, this is an achievable implementation based on simple statistics. It is quick, effective, and fundamental to the overall development and history of machine learning (Langley, P., Iba, W., & Thompson, K., 1992). In order to understand the implementation, understanding the algorithm and the statistics which it is based on is crucial.

2.0 Bayesian Probability and Bayes Theorem

Bayesian probability, developed by Thomas Bayes as well as Pierre-Simon Laplace, relies heavily on the notion that probability can be interpreted as a degree of belief in an event. This degree of belief can and does change based on new incoming information, making it clearly apt for use in machine learning techniques. To understand its application, it is important to first understand the fundamentals of Bayesian probability and its foundations in Bayes' Theorem.

2.1 Bayes Theorem

The fundamental root of Bayesian probability is Bayes' Theorem:

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

where $P(A|B)$ and $P(B|A)$ are conditional probabilities; that is, given that B (or A in the latter case) is true, it is the probability that A (or B) is true. $P(A)$ and $P(B)$ are marginal probabilities of A and B respectively. This theorem derives from the law of total probability. The law of total probability is best summed up using this illustration [Figure 2.1]:

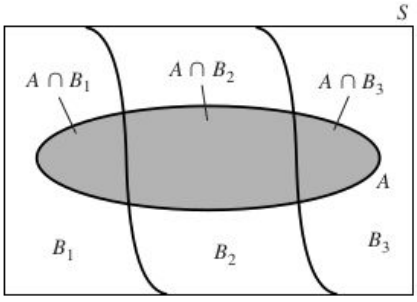

Figure 2.1 (Wackerly, Mendenhall III, & Scheaffer, 2008, p. 71)

In this illustration, the sample space S is split into partitions B such that $S = B_1 \cup B_2 \cup ... \cup B_k$ and $B_i \cup B_j = \emptyset\ (i \neq j)$. In the example shown, $k = 3$. Considering $A \subset S$, then $P(A) = \sum_{i=1}^{k} P(A \cap B_i)$ and by the definition of conditional probability (that is, $P(A|B_i) = \frac{P(A \cap B_i)}{P(B_i)}$): $P(A) = \sum_{i=1}^{k} P(A|B_i)P(B_i)$. Using the same definition of conditional events again and substituting for the law of total probabilities give the general form of Bayes' Theorem:

$$P(B_i|A) = \frac{P(B_i \cap A)}{P(B_i)} \Rightarrow P(B_i|A) = \frac{P(A|B_i)P(A)}{\sum_{i=1}^{k} P(A|B_i)P(B_i)}, \text{ or } P(B_i|A) = \frac{P(A|B_i)P(A)}{P(B_i)}$$

2.2 Bayesian Probability

Bayesian Probability adopts a diachronic interpretation of probability, meaning that the probability of an event can change over time. The probability of the hypothesis changes when new evidence is introduced (Downey, A. B.,2012). Revisiting Bayes' theorem, each term is defined under this interpretation:

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

Where:

- P(A) – Prior probability: the initial probability before new evidence is introduced
- P(A|B) – Posterior probability: the resulting probability after new evidence
- P(B|A) – Likelihood: the probability of new evidence under the hypothesis
- P(B) – Normalizing constant: the probability of new evidence under any hypothesis

As new evidence is introduced, the previous posterior probability becomes the prior probability and a new posterior probability is calculated. The likelihood is typically computed by counting methods. The normalizing constant is the most difficult to compute, as it may be hard to define "under any hypothesis". This can be simplified by selecting a set of hypotheses that are mutually exclusive and collectively exhaustive. (Downey, A. B., 2012)

3.0 Machine Learning and Bayesian Classifiers

Machine learning algorithms enable programs to learn from data without explicitly being programmed. Many types of machine learning systems exist and can be categorized using three broad metrics: the degree of human supervision in training the system, how data is made available to the system, and how the system generalizes. Variations in the degree of human supervision include supervised, unsupervised, semi-supervised, and reinforcement learning. Data can be made available to the system all at once (batch learning), or data can be introduced individually and sequentially (online learning). Machine learning systems can generalize by, using a metric of similarity, comparing new data to old data (instance-based learning), or by analyzing patterns in the data to build a predictive model (model-based learning). (Géron, A., 2019)

Bayesian classifiers commonly use a supervised, batch learning, instance-based approach and our implementation of the Naïve Bayes algorithm follows this. Supervised learning employs labeled training data that includes the correct output. The label, in the case of Bayesian classifiers, is the class that the training data belongs to. The full set of training data is used to train the machine learning system and additions to the data set requires the system to be re-trained. Bayesian classifiers generalize by comparing test data to training data. The metric of similarity used is the probabilities of the class and of the features within the class.

3.1 Naïve Bayes

The Naïve Bayes classifier is a simplified form of Bayes theorem. The classification task entails that given a set of features (f1 … fn) and set of classes (c1 … cn) determine which class a data point belongs to based on its features. Following from Bayes theorem, Naïve Bayes assumes conditional independence of features. In practice, it is likely that the features are not conditionally independent; however, through experimentation, it has been shown that Naïve Bayes classifiers still perform well under this assumption. Where Naïve Bayes differs from Bayes theorem is that the normalizing constant is not used. This reduces the complexity of the computation needed to make classifications. Generally, the normalizing constant is the most difficult to compute and is only required if the true probability of an event is needed. For a classification task, only a relative approximation of probability is needed to measure between classes. The formula for Naïve Bayes is as follows:

$$P(\text{class} \mid f_1, \ldots, f_n) \propto P(\text{class}) \prod_{i=1}^{n} P(f_1 \mid \text{class})$$

P(class | f_1,..., f_n) is calculated for each class and the predicted class is chosen by selecting the maximum:

$$\text{Class} = \arg\max \text{P(class)} \prod_{i=1}^{n} \text{P}(f_1 \mid \text{class})$$

Determining the probability of a class is straightforward – it is the probability of training set rows with that class value divided by the number of total rows. Where specific implementations of Naïve Bayes vary is in how $P(f_n \mid class)$ is calculated. This calculation is context-dependent on the distribution of the data set. For discrete data following a multinomial distribution, an appropriate implementation may use a multivariate Bernoulli, binomial, or a categorical Naïve Bayes. For continuous data, a more appropriate implementation would be the Gaussian Naïve Bayes.

4.0 Implementation and Design

Our Naïve Bayes classifier implementation is written in standard Python and uses the 3.6.7 Python environment. The algorithm can be summarized as follows:

1. Read input data
2. Split data into training and testing sets
    a. 70% training
    b. 30% testing
3. Calculate Probabilities
    a. Prior probability
    b. Likelihood
    c. Posterior probability
4. Select predicted class using probabilities
    a. Maximum posterior probability
5. Calculate accuracy of predictions

A detailed flowchart of the process is shown on the following page [Figure 4.1]. This process is repeated on the same data set for 100 trials.

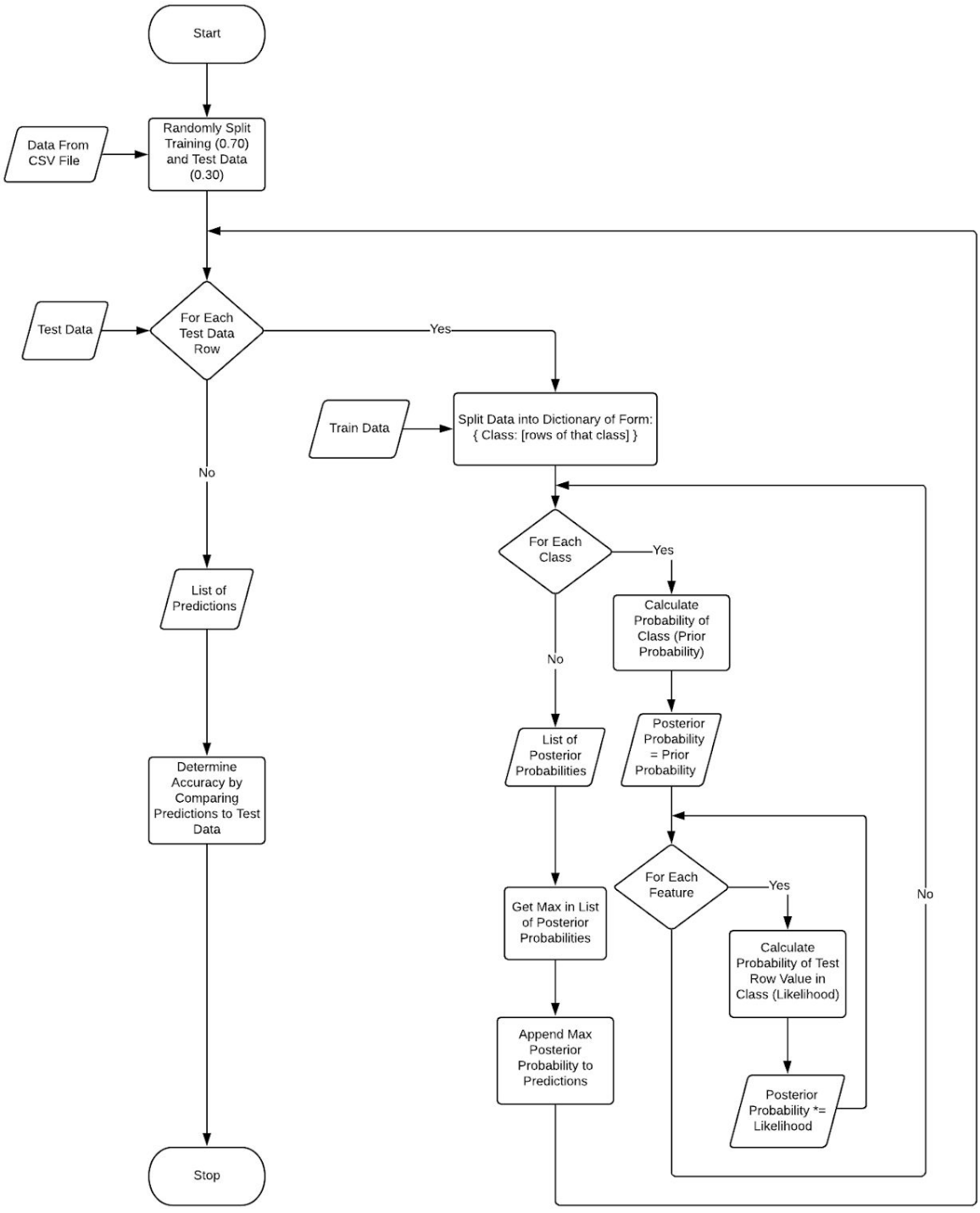

Figure 4.1 Bayes' Algorithm Flowchart

4.0.1 Calculations

The calculations used for Naïve Bayes classification are described in this section.

4.0.1.1 Prior Probability

Calculated by dividing the number of data instances with that class in the training set by the total number of data instances in the training set.

4.0.1.2 Likelihood

Given a test row to be predicted, the likelihood of each feature value in the test row is calculated for each class. This is calculated by dividing the number of data instances with that feature value, within the class, by the total number of data instances of that class.

4.0.1.3 Posterior Probability

Multiple posterior probabilities are calculated for each test row – one for each possible class. The prior probability of a class is multiplied by the likelihood of each test row feature in that class.

4.0.1.4 Classification

Each test row will have a set of posterior probabilities corresponding to the number of possible classes. Classification selects the maximum value in the set of posterior probabilities and that max value is assigned as the predicted class for that test row.

4.0.2 Testing

Alongside our implementation of the Naïve Bayes algorithm, two separate algorithms were implemented to test how they compare with our algorithm in terms of accurate predictions.

4.0.2.1 Random Algorithm

For random testing, a random class is assigned to each row in the test data set.

4.0.2.2 Benchmark Algorithm

To facilitate benchmark testing, we implemented a Naïve Bayes classifier using the Sci-kit Machine Learning Library and Pandas Data Analysis Library (Sci-kit Learn, 2018). The benchmark implementation uses the Anaconda Python environment for interpretation. The

Pandas Library translates the raw csv data into a data frame ready for processing by the Sci-kit Library. The Sci-kit Library handles: splitting the training and testing data; fitting the data into a given a model; and predicting test row classification based on that model. During our testing, the Sci-kit Gaussian Naïve Bayes and Multinomial Naïve Bayes algorithms were tested. The Gaussian Naïve Bayes (average accuracy = 50.4615%) greatly outperformed the Multinomial Naïve Bayes (average accuracy = 16.3846%) and, thus, was selected as the model for benchmarking. The Gaussian Naïve Bayes model assumes that the likelihood of features follows a Gaussian distribution [Figure 4.2]:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Figure 4.2 (Sci-kit Learn, 2018)

4.1 Data and Inputs

The set of data that was used is a modified version of (UCI Machine Learning Repository, 1990). As the original dataset was from 1990 and many changes have occurred in the world since then, the data was updated to reflect this. The current complete data set that was used can be seen in Appendix II. Some resources that were used for updating the data include (Hunt, M. H., 2015; De Waal, T., 2003; Olsson, H., Olsson, C., 2018). A note in the description of the data was also changed concerning the categories of religions, as this was not deemed an accurate method to categorize religions.

The original data placed all of the world's religions into seven categories: Catholic, Christian (not Catholic), Muslim, Buddhist, Hindu, Ethnic, Marxist, and Others. While it is not known the exact process for determining these categories, they nearly sufficed but for a few questionable categories. For example, Marxism was included as a type of religion; however, while the doctrine of Marxism holds a similar power over a country as a state religion, it does not constitute a religious belief. Formerly denoted Marxist nations were put into either the Distinct Cultural Beliefs category or the Other category.

For a more rigorous approach to classifying religions, one can look in several places. One of the oldest ways to classify religions is known as Normative classification: simply, dividing "true" religions from "false" religions. This is an inherently flawed system as defining a "true" or "false" religion without bias is impossible. Another method would be to use geographical methods: religions which are nearest to each other are in similar groups. This has its own problems in defining the geographic bounds of an idea (Encyclopedia Britannica, n.d.). Further along this idea is Ethnographic-linguistic classification. This classification relies on the key relationship between religion and language. The father of this method, Max Müller, suggested an

intimate relationship between the two. Unfortunately, this theory was developed during a time when it was not the connection between language and religion that was being studied: it was the connection between race and religion. Thus many of these theories rely on inherently flawed views of race. (Ward D. H. J., 1909; Müller, M., 1882). The Ethnographic-linguistic approach was preceded by the same Philosophical approach which also overtook it later, spun first by the philosopher Hegel and the theologian Pfleiderer (Hegel, G. W. F., 1895; Encyclopedia Britannica, n.d.). This is a method seeking to categorize religions based on their philosophies. Yet again, many of these methods are incredibly biased and lack understanding of any religions beyond those in the philosopher's backyard. A more scientific approach that led to a classification somewhat similar to the one used in our data set is known as Morphological. It is predominantly focused on classifying religions based on attributes described first by Tylor in (Tylor, E. B., 1920) and refined in other works such as those of Dutch scholar Tiele and fellow Dutch philosopher Kuenen (Kuenen, A., 1882; Encyclopedia Britannica, n.d.). As the topic of classifying religions is hotly debated in the field of philosophy, it is not the place of this data set to solve such an issue. So the data was changed using our best judgement which could be given for the circumstance, and left at that.

Other than the data, the program requires the column in the data set that is to be predicted, and the number of trials to run. The number of trials is needed in order to have a better representation of the accuracy of the predictions, as running the program once may produce an outlier case that can skew the perception of the accuracy of the predictions.

4.2 Outputs

The program uses 70% of the inputted data set to train the classifier (the training set) and 30% of the data set for testing the accuracy of the classifier (the testing set). This is discussed in further detail in the following section. Once the algorithm has completed finding a set of predictions for each tuple in the testing set, it compares this to the actual value. The best, worst, and average accuracies, as well as the time taken, are given as output. For comparison purposes, the algorithm also compares this to randomly assigning predictions, as well as the (Sci-kit Learn, 2018) implementation. The best, worst, and average accuracies of these other predictions are also given.

4.3 Design Decisions

Our complete Python program is available in Appendix I. It is written in and meant to be run using Python 3.6.7, using the following imported libraries: csv (Python Software Foundation, n.d. b), random (Python Software Foundation, n.d. a), and time (Python Software Foundation, n.d. c). CSV, or comma separated value, files are a common and simple method for storing tabular data in a text format. RFC4180 describes the specification for this format (Shafranovich,

Y., 2005). The random library is used for splitting the data into the testing and training sets uniquely each time the program is run. time is used to retrieve the processing time for the algorithm. The intent was to use libraries in the portions of the code that do not contribute to the algorithm itself, but rather to use them to expedite the portions of the code which are peripheral to the algorithm.

At the beginning of the implementation process, the ratio of the data set that went to training was 70%. This was reasoned as a proper balance between an undersized training set, which would lead to poor predictions, and an undersized testing set, which would lead to insufficient testing. A more rigorous approach can be found in (Guyon, I., 1996, p. 7). Using their formula:

$$\frac{f^*}{g^*} = \sqrt{\frac{C \ln(N/\alpha^2)}{h_{max}}}$$

Figure 4.3

where f* and g* are the fractions of data to be used for training and testing respectively, C is a constant 1.5 known as the Chernoff bound, N is the number of families as they are defined in (Highleyman, W. H., 1962), α is the acceptable incorrect ratio, and h represents the largest complexity of the families considered. Using our values were C = 1.5, N = 200, α = 0.05, h = 3, we get a resulting ratio of approximately 70% training data and 30% testing data.

When a probability would be exactly 0, it was decided that the value which should be returned is 0.0000000001. This is because when one probability from the training set is 0 then the rest of the probabilities multiplied by that probability are also 0; as a result this one probability completely overwrites the other probabilities. If the value is replaced by an arbitrarily small number then the contribution of the other probabilities is preserved but the low probability also has its just impact on the probability. This is a well documented approach known as additive smoothing (Manning, C. D., Raghavan, P., & Schutze, M., 2008). Interestingly, the technique was originally developed by none other than Pierre-Simon Laplace in order to predict the likelihood of the sun rising tomorrow. This was the very same Laplace that developed most of the probability theory that is classified as “Bayesian”. (Fahrmeir, K., 2011)

Also factored into the probability is the overall probability of the target attribute. While this is not given in the traditional Naïve Bayes’ algorithm, it increased the accuracy of the predictions by not over-predicting the less-common religions. Had all of the religions specified been equiprobable, this step would not have been necessary.

5.0 Analysis

To analyze the performance of each algorithm, we performed multiple test runs and recorded the results. The data consists of 10 test runs, each with 100 trials for a total of 1,000 trials. The data groupings represent the minimum, average, and maximum accuracies of the Bayes' algorithm (our implementation), the benchmark algorithm, and the random algorithm. The chart below outlines the performance of each algorithm in all 10 test runs [Figure 5.1]:

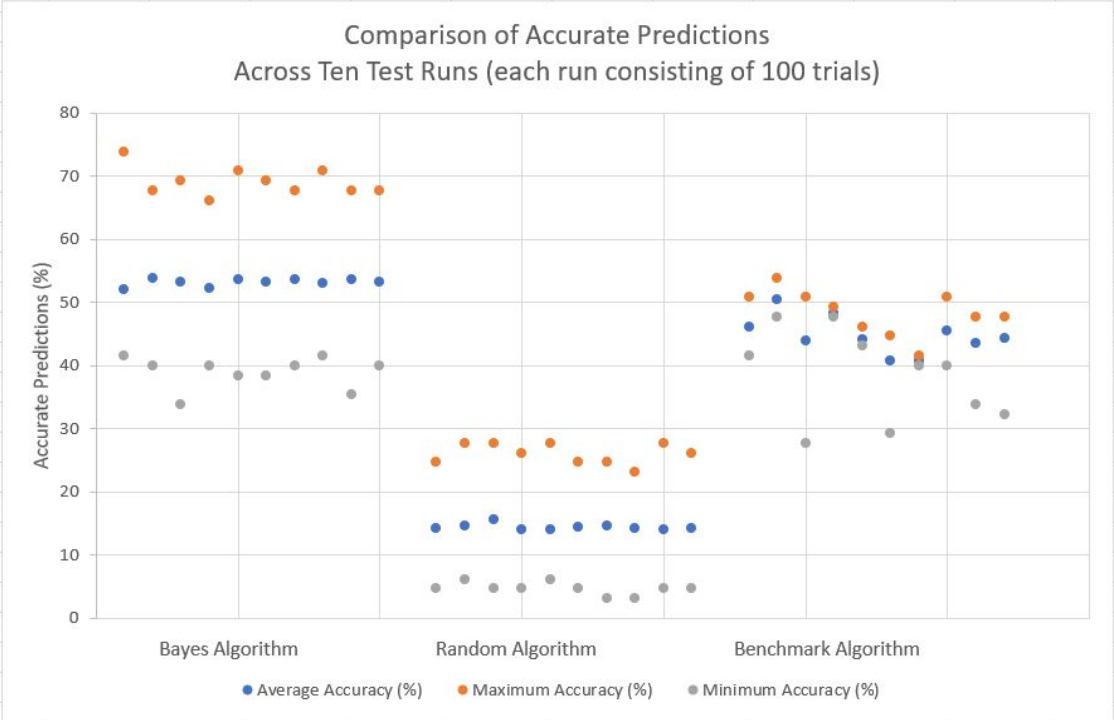

Figure 5.1

See Appendix III for full data table.

Across all test runs, the Bayes' algorithm, random algorithm, and benchmark algorithm have an average accuracy of 53.1431%, 14.3846%, and 44.7631%, respectively. It was expected that our Bayes' Algorithm would outperform the random algorithm, which is an uninformed algorithm operating by pure chance. The Bayes' algorithm outperforming the benchmark algorithm can likely be attributed to the difference in how the likelihood is calculated. For our implementation, the likelihood is determined by calculating the relative frequency of a given feature in each class. In contrast, the benchmark algorithm assumes a Gaussian distribution of features and under this assumption the likelihood is calculated. Our implementation is better suited for the chosen data set and its discrete properties; however, given a different continuous data set the benchmark algorithm may outperform the Bayes' algorithm.

6.0 Conclusion

The implementation of Naïve Bayesian Classifiers discussed in this paper performed comparably to the one implemented in one of the most commonly used libraries in Python. While our implementation was constructed with our specific data set in mind, the method in which we wrote the program was intended to be as flexible as possible; given another data set, there is unlikely to be any changes necessary (provided this data set is in a similar CSV format). However, in this case the benchmark algorithm could outperform our implementation.

Our data focused on the relationship between the dominant or most commonly practiced religion in a country and the national flag of that country. As previously discussed, the relationship between flags and religion has been engrained on the history of many nations: from the countries like Scotland and Japan that created their flags in a religious context to modern states like Libya adapting a more religious reflective flag as the government changes its perspective. Though the nations which do not uphold a state religion are the most fascinating sub set of this data; while most countries with stars and a crescent moon are Muslim majority, Singapore retains its Buddhist beliefs and its historically significant flag. Retained in our implementation of the Naïve Bayesian Classifier is the brief but explosive history of machine learning. The statistical methods first pioneered by Thomas Bayes and Pierre-Simon Laplace in the 1800s befit the relatively new field of machine learning. As the field progresses in all its rapidity, the fundamental roots of machine learning often depended on the contributions of the past and can be better understood through a deep study of these algorithms. In this vein, our implementation is significant not only in its role as a comparable one to the currently held standard in Python, but in its contribution to the understanding of machine learning as it continues its rise in the general field of artificial intelligence.

7.0 Acknowledgements

We would like to acknowledge and thank our sources listed in the reference section; clearly, this report would be nothing without their work. Similarly, the writers of the libraries that were used in the implementation saved us a tremendous amount of time and effort and allowed our focus to be on the algorithm and not all of the legwork peripheral to the algorithm implementation. Finally, we'd like to thank our professor, Dr. Mouhoub, for his support and favourable evaluation.

8.0 References

Bartram, G. (2001). *Proceedings of the XIX International Congress of Vexillology*. York, UK: Fédération internationale des associations vexillologiques.

Bates, R. (2007). *All About Chinese Dragons*. Morrisville, North Carolina: Lulu.com.

Bjerg, H. C. (2006). *Danneborg*. Gjern, Denmark: Hovedland.

Dane, P. (2008). Flags in Context: A Discussion of Design, Genre, and Aesthetics. *Raven: A Journal of Vexillology*. Vol. 15.

De Waal, T. (2003). *Black Garden: Armenia and Azerbaijan Through Peace and War* (illustrated, revised). New York: NYU Press.

Downey, A. B. (2012). *Think Bayes Bayesian statistics made simple; Version 1.0.5*. Needham, MA: Green Tea Press.

Dyer, H. (1909). *Japan in World Politics: A Study in International Dynamics*. Glasgow: Blackie & Son Limited.

Edgington, D. W. (2003). *Japan at the Millennium: Joining Past and Future*. Berkeley, USA: UCB Press.

Encyclopedia Britannica. (n.d.). Classifications of Religion. Encyclopedia Britannica. Retrieved from https://www.britannica.com/topic/classification-of-religions#ref38028.

Fahrmeir, L. & Kneib, T. (2011). Bayesian Smoothing and Regression for Longitudinal, Spatial and Event History Data. *OUP Catalogue*. Oxford: Oxford University Press.

Géron, A. (2019). Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. Retrieved March 24, 2019, from https://learning.oreilly.com/library/view/hands-on-machine-learning/9781492032632/ch01.html

Government of Sri Lanka. (n.d.). *National Symbols of Sri Lanka*. Retrieved from http://www.labour.gov.lk/web/index.php?option=com_content&view=article&id=192%3Anational-symbols-of-sri-lanka&catid=69%3Asri-lanka-facts&lang=en.

Guyon, I. (1996). *A Scaling Law for the Validation-Set Training-Set Size Ratio*. Murray Hill, USA: AT&T Bell Lab.

Harrington, K. (2003). *Colours, crosses or cow-horns? Nordic elements in the design of some North American flags and emblems with an overview of the use of Scandinavian flags in Canada and the U.S.A*. Stockholm: Proceedings of the XX International Congress of Vexillology.

Hegel, G. W. F. (1895). *Lectures on the Philosophy of Religion*. Oakland, USA: UC Press

Highleyman W. H. (1962). *The design and analysis of pattern recognition experiments*. Murray Hill, USA: The Bell Systems Technical Journal.

Hunt, M. H. (2015). *The World Transformed: 1945 to the Present* (2nd ed. revised). Oxford: Oxford University Press.

Itoh, M. (2003). *The Hatoyama Dynasty: Japanese Political Leadership Through the Generations*. London: Palgrave Macmillan.

Kuenen, A. (1882). *National Religions and Universal Religions*. New York: C. Scribner's Sons.

Langley, P., Iba, W., & Thompson, K. (1992). *An Analysis of Bayesian Classifiers*. Moffett Field, USA: NASA Ames Research Centre, AI Research Branch.

Lee, K. Y. (1998). *The Singapore Story: Memoirs of Lee Kuan Yew*. Singapore: Times Editions.

Macgeorge, A. (1881). *Flags: Some Account of Their History and Uses*. Glasgow: Blackie & Son.

Manning, C. D., Raghavan, P., & Schutze, M. (2008). *Introduction to Information Retrieval*. Cambridge: Cambridge University Press.

Müller, M. (1882). *Introduction to the Science of Religion*. London: Longmans Green & Co.

Olsson, H., Olsson, C. (2018). Mauritania. *Africa Yearbook*. Brill. Vol. 14.

Perrin, W. G. (1922). *British Flags; Their Early History and their Development at Sea, with an Account of the Origin of the Flag as a National Device*. Oxford: Oxford University Press.

Pew Research Center. (2014). Religion in Latin America. Washington: Pew Research Center. Retrieved from https://www.pewforum.org/2014/11/13/religion-in-latin-america/.

Python Software Foundation. (n.d. a). 9.6. random - Generate pseudo-random numbers. Retrieved from https://docs.python.org/3.6/library/random.html

Python Software Foundation. (n.d. b). 14.1. csv - CSV File Reading and Writing. Retrieved from https://docs.python.org/3.6/library/csv.html

Python Software Foundation. (n.d. c). 16.3. time - Time access and conversions. Retrieved from https://docs.python.org/3.6/library/time.html

UCI Machine Learning Repository. (1990). Flags Data Set [Data file]. Retrieved from https://archive.ics.uci.edu/ml/datasets/Flags

Tylor, E. B. (1920). *Primitive Culture: Researches into the Development of Mythology, Philosophy, Religion, Language, Art and Custom*. London: Murray.

Sci-kit Learn. (2018). *1.9 Naive Bayes*. Retrieved from https://scikit-learn.org/stable/modules/naive_bayes.html

Shafranovich, Y. (2005). Common Format and MIME Type for Comma-Separated Values (CSV) Files. *The Internet Society*. Retrieved from https://www.ietf.org/rfc/rfc4180.txt

Singapore Statistics. (2015). *Highlights of General Household Survey 2015*. Singapore Statistics.

Retrieved from https://www.singstat.gov.sg/-/media/files/visualising_data/infographics/ghs/highlights-of-ghs2015.pdf

Strangio, S. (2014). *Hun Sen's Cambodia*. New Haven: Yale University Press.

Wackerly, D. D., Mendenhall III, W., & Scheaffer, R. L. (2008). *Mathematical Statistics with Applications* (7th ed.). Belmont, CA: Thomson Brooks/Cole.

Ward, D. H. J. (1909). *The Classification of Religion*. Chicago: The Open Court Publishing Co.

9.0 Appendices

9.1 Appendix I

This is our program, written in and meant to be run using Python 3.6.7. It uses the following libraries: csv (Python Software Foundation, n.d. b), random (Python Software Foundation, n.d. a), and time (Python Software Foundation, n.d. c). Our repository is available at https://github.com/rileyherman/bayesian-classifier.

9.2 Appendix II

This is the data that was used. It is a modified version of the original set from (UCI Machine Learning Repository, 1990).
https://github.com/rileyherman/bayesian-classifier/tree/master/datasets/flags.

9.3 Appendix III

A statistical analysis of the implementation compared to random guessing and the currently most popular Python implementation:

Time (seconds)	Average Accuracy (%)	Maximum Accuracy (%)	Minimum Accuracy (%)
Bayes			
9.6406	51.9538	73.8462	41.4385
9.4062	53.7846	67.6923	40
9.75	53.3077	69.2308	33.8462
9.7031	52.2769	66.1538	40
10.6094	53.5538	70.7692	38.4615
10.7656	53.2154	69.2308	38.4615
10.5781	53.5231	67.6923	40
10.0625	52.9538	70.7692	41.5385
10.5	53.5692	67.6923	35.3846
10.1875	53.2923	67.6923	40
Random			
0.1875	14.1846	24.6154	4.6154
0.1875	14.5692	27.6923	6.1538

0.1719	15.5538	27.6923	4.6154
0.1875	13.9538	26.1538	4.6154
0.2031	14.0923	27.6923	6.1538
0.1875	14.3077	24.6154	4.6154
0.1875	14.6308	24.6154	3.0769
0.1719	14.2154	23.0769	3.0769
0.2031	14.0615	27.6923	4.6154
0.2344	14.2769	26.1538	4.6154
Benchmark			
1.5156	46.0615	50.7692	41.5385
1.5156	50.4615	53.8462	47.6923
1.5156	43.9231	50.7692	27.6923
1.5	48.2154	49.2308	47.6923
1.5781	44.1231	46.1538	43.0769
1.5781	40.8	44.6154	29.2308
1.5625	40.6769	41.5385	40
1.5625	45.4923	50.7692	40
1.7031	43.5077	47.6923	33.8462
1.6406	44.3692	47.6923	32.3077

https://docs.google.com/spreadsheets/d/1thLQlbKX3dXrsGJO8XIG_lfHkQ34l1kSBCp27im8X3c/edit?usp=sharing