Tackling the Independence Assumption of the Naive Bayes Classifier: Dealing with Dependent Attributes by Weighting

Team Members: Eshwar Moorthy, John Kim

January 16, 2025

Table of Contents

Part 1 – Abstract

The traditional Naive Bayes classifier assumes independence between features, which is often unrealistic. Weighting each attribute differently depending on its dependencies could solve this problem while maintaining simplicity. Thus, we are proposing a method for determining how to weight each attribute, which would work well with dependent features. We will compare its performance with that of the standard Naive Bayes and Decision Tree models with three different datasets consisting of different extents of feature dependencies.

Part 2 – Introduction

The Naive Bayes Classifier creates a network fully based on Bayes' Theorem of conditional probability, which is the rule used to predict the probability of the different values. Each value in each feature is then assessed to find the value of the predictability and then it's assigned a probability value. One of the main problems with implementing the Naive Bayes model is that there is an assumption of independence within all of the features/attributes in the dataset. If many of the features show clear dependence with each other such as "shipping address" and "billing address."

Although other models can be programmed to classify properly, the independence assumption of Naive Bayes being tackled would make the model a lot more accurate and easier to predict, as Bayes' theorem fully comes from probability rules. The Naive Bayes model is very well known due to its interpretability with datasets consisting fully of independent features. Enhancing the model to attribute/appeal towards dependencies doesn't just improve the application aspect, but it also increases its rating as an accurate predictor with complex models. We wanted to refine this algorithm to tackle dependency in order to extend its usage in scenarios where the features' relations with each other won't harm the model's predictability.

Part 3 – Related Works

Addressing the Naive Bayes' independence assumption is a crucial area that researchers have delved into as a way to analyze the impact on the classification's accuracy. There have been many methods that were implemented to tackle this issue, like Tree-Augmented Naive Bayes (TAN) and Lazy Bayesian Rules (LBR). The TAN method relaxes the assumption of independence by letting each attribute depend on each other, eventually becoming a structure that improves the model's way of showing dependencies while maintaining the computational efficiency when testing the model (Webb et al., 2005). Similar to this, the LBR method creates a dependency structure dynamically for each instance during the steps of building the model, which would take a long time to process (Zheng & Webb, 2000). These methods improve the Naive Bayes model's ability to handle dependencies with traits, especially when there are redundant attributes such as "billing address" and "shipping address," which would make the Naive Bayes' performance worse. Even though the accuracies of these models would be higher, this does lead to sacrificing the algorithm's computational efficiency when training, so further refinements could be made to tackle this issue (Zaidi et al., 2013).

A more recent approach that has been made to refine this Naive Bayes model would be through the implementation of giving different attribute weights, like the Weighted Naive Bayes (WANBIA) algorithm. This method is able to give attributes specific weights that are able to tackle the issue of dependency violations by reducing the impact of redundancy or dependent attributes (Zaidi et al., 2013). Unlike the methods of TAN and LBR where the structure is modified, WANBIA uses a weighted approach in order to learn these weights, eventually leading to optimizing the classification objectives. These are done through 2 different methods: Conditional Log Likelihood and Mean Squared error. (Zaidi et al., 2013). The effectiveness of this comes from its way of handling downweight redundant attributes while being able to retain the highly predictive ones. This was shown when the paper compared the performance with Random Forests and Logistic Regression. (Zaidi et al., 2013). All of these advancements fully match the goal of tackling dependency with Naive Bayes so that the model can predict with complex datasets and keep the same Bayes' Theorem.

Part 4 – Dataset and Features

We used WEKA's *cancer* and *vote* datasets and Kaggle's *balance-scale* datasets so that we can have varying values of average Conditional Mutual Info (CMI), a metric similar to the Pearson correlation coefficient but one that works for categorical data. (A CMI of 0 means that two features are perfectly independent of each other, while a CMI of 1 means that the two are perfectly dependent on each other.) For each dataset, we averaged the CMI value of all permutations of distinct attribute pairs to get the average CMI.

Balance-scale Dataset

- 4 features: *L-Weight, L-Distance, R-Weight, R-Distance*
- The class describes which way the balance scale tips
- The average CMI is 0

Ca*ncer* Dataset

- 10 features: *age, menopause, tumor-size, inv-nodes, node-caps, deg-malig, breast, breast-quad, irradiat*
- The class describes whether the cancer is recurring or not.
- The average CMI is 0.06

Vote Dataset

- 17 features: *handicapped-infants, water-project-cost-sharing, adoption-of-the-budget-resolution, physician-fee-freeze, el-salvador-aid, religious-groups-in-schools, anti-satellite-test-ban, aid-to-nicaraguan-contras, mx-missile, immigration, synfuels-corporation-cutback, education-spending, superfund-right-to-sue, crime, duty-free-exports, export-administration-act-south-africa*
- The class describes whether the voter is a Democrat or Republican
- The average CMI is 0.11

For each dataset, we discretized all attribute values using WEKA's "Discretize" filter and replaced any missing values with the mode of the attribute. Then, we used an 80% train and 20% test split using stratified random sampling for each dataset.

Part 5 – Methods

The default Naive Bayes algorithm goes like this. For each instance, the class label with the greatest P(class | X), as given by the formula below, is picked.

$$P(class = y \mid X) = P(class = y) * \prod_{i=1}^{\#\, of\, features} P(X_i \mid class = y) \quad (1)$$

Our Weighted Naive Bayes algorithm adds to the default Naive Bayes algorithm by having a list of weights, with each weight corresponding to each feature. Here, we are proposing that each value inside the product operation, P(X_i | class=label), is raised by the power of a weight w_i.

$$P(class = y \mid X;\, w) = \gamma_{yx}(w) = P(class = y) * \prod_{i=1}^{\#\, of\, features} P(X_i \mid class = y)^{w_i} \quad (2)$$

The list of weights is calculated by maximizing the Logarithmic Conditional Probability (LCP) via backpropagation. The LCP is calculated as follows:

$$LCP(w) = \sum_{i=1}^{\#\, of\, features} log\left(\frac{\gamma_{yx}(w)}{\sum_{y'=1}^{\#\, of\, labels} \gamma_{y'x}(w)}\right) \quad (3)$$

Once the weights are determined, they are used to classify instances using Equation 2 in a similar way to that of the default Naive Bayes algorithm. For each dataset, we calculated the accuracy using the default Naive Bayes, Weighted Naive Bayes, and Decision Tree classifiers.

Part 6 – Results and Analysis

Our results for the three datasets are as follows:

Dataset: balance-scale.csv
Average Conditional Mutual Information: 0.0
Default Naive Bayes Classifier Accuracy: 90.4%
Weighted Naive Bayes Classifier Accuracy: 90.4%
Decision Tree Classifier Accuracy: 78.4%

Dataset: cancer.csv
Average Conditional Mutual Information: 0.06
Default Naive Bayes Classifier Accuracy: 72.41%
Weighted Naive Bayes Classifier Accuracy: 75.86%
Decision Tree Classifier Accuracy: 65.52%

Dataset: vote.csv
Average Conditional Mutual Information: 0.11
Default Naive Bayes Classifier Accuracy: 89.66%
Weighted Naive Bayes Classifier Accuracy: 94.25%
Decision Tree Classifier Accuracy: 93.1%

These results show that our method for weighting attributes is working. For the *balance-scale* dataset, whose features are completely independent of each other, our Weighted Naive Bayes classifier performed the same as the default Naive Bayes classifier, as it should since the features don't violate the Independence Assumption. For the *cancer* dataset, whose features have some dependency but not too much, our Weighted Naive Bayes classifier performed ~3% better than the default one. Finally, for the *vote* dataset, whose features are even more dependent, our Weighted Naive Bayes classifier performed even better (~4.5% better) compared to the default Naive Bayes Classifier.

Part 7 – Conclusion

Based upon what was above, the Weighted Naive Bayes Classifier implementing LCP emerged out to be the model that had the highest accuracy with respect to its default counterpart, especially when the datasets consisted of features that are highly dependent on each other. The ability for the classifier to assign weights effectively for the attributes clears the issue of dependency that the default Naive Bayes fails to combat. When we compared our performance when Naive Bayes for the cancer dataset and the vote dataset, there was at least a 3% increase in the accuracy as there was dependency within the features themselves. This increase in the performance comes from the classifier's capacity to decrease the impact that is caused by redundancy with the attributes while still applying the predictive power of the other attributes given. When comparing with Decision Tree, we noticed that the accuracies were either very similar to, but slightly less than, our model or they were much less than our model, also showing how competitive our model is in comparison to other highly-used classifiers. With additional resources, we would want to delve more into creating hybrid models that are able to work with datasets of really large sizes and really complex information, which, through the use of more sophisticated weighting techniques and computational capacity, decrease add-ons to this. Overall, our model had predicted higher than the other 2 models that were tested as base cases.

Part 8 – Contributions

Project Idea - Both
Introduction / Related Works - Eshwar
abstract - Eshwar
Datasets - Both
Methods / Results and Analysis - John
Conclusion - Eshwar
Code - John
Presentation - Both

Part 9 – References and Bibliography

Alleviating naive Bayes attribute independence assumption ... (n.d.).

https://jmlr.org/papers/volume14/zaidi13a/zaidi13a.pdf

Shushrut. (2019, April 28). *Balance scale*. Kaggle.

https://www.kaggle.com/datasets/mysticvalley/balance-scale

Webb, G. I., Boughton, J. R., & Wang, Z. (2017, July 21). *Not so naive bayes: Aggregating one-dependence estimators - machine learning*. SpringerLink.

https://link.springer.com/article/10.1007/s10994-005-4258-6