Quarter 2 Final Report

James Beck & Jia Mody

Dr. Yilmaz's 5th Period Machine Learning Class

January 17th, 2025

Abstract:

Our project enhances the Naïve Bayes classifier by integrating one-dependence estimators with adaptive attribute weighting to mitigate its independence assumption while maintaining efficiency. By integrating a SuperParent Tree-Augmented Naïve Bayes (TAN) model, we introduce limited dependencies between attributes, and through an Artificial Immune System-inspired weighting mechanism, we dynamically adjust attribute importance based on relevance to the target class. We evaluate our model using a real estate pricing dataset, optimizing key parameters such as the super parent selection metric, smoothing parameter, and attribute weight evolution process. Our experiments aim to balance predictive accuracy and computational efficiency, demonstrating the potential of hybrid Naïve Bayes models in handling real-world datasets with interdependent features.

Introduction:

The Naïve Bayes algorithm is a cornerstone of Machine Learning due to its simplicity, efficiency, and effectiveness. Naïve Bayes is particularly valuable in domains such as text classification, spam filtering, and even medical diagnoses, where interpretability and computational efficiency are key.

Despite its utility, the independence assumption of Naïve Bayes is often unrealistic in real-world datasets. Many researchers have proposed modifications to Naïve Bayes framework that relax the independence assumption while maintaining efficiency and interpretability.

Our project aims to mitigate the Naive Bayes's independence assumption by combining one-dependence estimators with self-adaptive attribute weighting. One dependence estimators allow for dependencies between attributes while adaptive attribute weighting "weights" the importance of each attribute in predicting a target class. Integrating these two techniques, our model seeks to balance computational efficiency and predictive power, contributing to the ongoing evolution of Naive Bayes and its applicability to complex real-world problems.

Related Work:

Naive Bayes classifier can be enhanced by relaxing its independence assumption, leading to strategies for improving predictions. One method, Averaged One-Dependence Estimators (AODE), introduces 1-dependence models where each attribute depends on one parent attribute, reducing bias. However, AODE's performance falters on large datasets where attribute dependencies may be more complex, found by Webb, Boughton, & Wang (2005) as well as Lee, O, $ Eck (n.d.). Artificial

Immune Systems (AIS), adapts attribute weights using a process inspired by the human immune system, achieving competitive results across different dataset sizes, although it is computationally expensive (Wu et al., 2015). Attribute weights can also be computed using Conditional Log-Likelihood (CLL) or Mean Squared Error (MSE) (Zaidi, Cerquides, Carman, & Webb, 2013). Locally weighted learning can also improve Naive Bayes. A local model is trained on a weighted subset of data, with the weights based on the distance from the test instance (Frank, Hall, & Pfahringer, 2003). These techniques allow for more accurate predictions by better handling attribute interdependencies.

Dataset & Features:

Our dataset was published on "kaggle.com". The class attribute is house price. It contains 545 instances and 12 features. There is no missing data. Below is an exhaustive list of the attributes:

	Attribute	Description
1	Area	The total area of the house in square feet (#)
2	Bedrooms	Bedrooms in house (#)
3	Bathrooms	Bathrooms in house (#)
4	Stories	Stories in house (#)
5	Mainroad	House is connected to main road (Y/N)
6	Guestroom	House had a guest rooms (Y/N)
7	Basement	House has a basement (Y/N)
8	Hot Water Heating	House has hot water heating (Y/N)
9	Air Conditioning	House has air conditioning systems (Y/N)
10	Parking	Parking spaces available within house (#)
11	Prefarea	House is in preferred location (Y/N)
12	Furnishing Status	Furnishing status of house (Fully Furnished, Semi-Furnished, Unfurnished)

Table 1. Attribute Descriptions

For preprocessing, we had no missing values, and Naïve Bayes does not require normalization, so we just discretized the area and price attributes for convenience.

Methods:

In our algorithm design, we wanted to both (1) account for dependencies between attributes in a way such that our algorithm works on datasets of all sizes and (2) train quickly and efficiently. Therefore, our method is a one-dependence model & attribute weight hybrid that utilizes both methods to achieve high accuracy and a fast attribute weight convergence. In a one dependence model, the probability of the class being *y* given a certain tuple *x* is given by the equation

$$P(y|x_1, x_2, \ldots, x\text{⍰}) \ = P(y) \prod_{i=1}^{n} P(x_\mathrm{i}|x(parent_\mathrm{i}), y)$$

Equation 1. Superparent Naive Bayes

In the above equation, $\mathrm{x_i}$ is now allowed to depend on $\mathrm{x(parent)_i}$ in addition to y.

The one dependence model we will create is *SuperParent TAN*. In *SuperParent TAN*, each attribute depends on the same attribute or *super parent* in addition to the class attribute (in other words, $\mathrm{x(parent)_i}$ is the same for each attribute).

To determine the *super parent*, our algorithm will use one of two metrics: conditional mutual information (CMI) or correlation with the class attribute. An in depth description of these two measures is provided in the "Experiments and Discussion" section.

As previously mentioned, one dependence models do not entirely handle attribute dependencies, particularly in datasets where attributes may be dependent on more than one other attribute. For this reason, we will then weight the attributes in the one dependence model using the self-adaptive attribute weight model suggested in (Wu et. al., 2015).

Below is a step by step description of the algorithm:

1. L weight vectors are initialized. These vectors comprise the first *generation*.
2. The weight vectors are then trained and evaluated on the training data and the best performing weight vector is duplicated. These duplicates replace the lowest performing weight vectors in the *generation and are* then identified.
3. *Mutations* are applied to all weight vectors (except the best performing) to produce weight vectors that are slight variations of the best performing weight vector. This new group of vectors forms the second *generation*. Vectors are mutated via the formula:

$$v_i^{t+1} = w_i^t + F \cdot N(0, 1) \cdot (w_s^t - w_i^t)$$

Equation 2. Mutation Formula

where t represents the weight vectors generation, w_i is the vector to be mutated, w_j is the best performing weight vector, N(0,1) is a normally distributed random variable, and F is the variation factor (equivalent to one minus the vectors performance).

4. This process will be repeated multiple times until the improvement of weight vectors falls below a certain threshold. The best performing weight vector will then be used to create the final weighted Naive Bayes model.

We believe this approach will be more efficient and accurate because the one dependence model plays a significant role in describing attribute dependencies. As a result, attribute weights will have less work to do, describing only the remaining dependencies not accounted for by the one dependence model. In other words, attribute weights will contribute less to increasing the accuracy of the model in this hybrid approach. Therefore, the improvement between generations of weight vectors will fall below a threshold T faster, causing the algorithm to converge more quickly, and fewer dependencies will be approximated by weights, but rather explicitly described by the one dependence model, making the model more accurate.

Experiments & Discussion:

To evaluate our model's performance, we will conduct three experiments: (1) comparing CMI and correlation for super parent selection, (2) fine-tuning the smoothing parameter, and (3) optimizing the number of generations or improvement threshold in AISWNB.

1) CMI vs. Correlation for Super Parent Selection:

We will compare CMI, which measures the dependency between two attributes given the class label and Cramer'sV which measures the association between two variables via the Chi-Square statistic:

$$I(X_i; X_j|Y) = \sum_{x_i, x_j, y} P(x_i, x_j, y) log\frac{P(x_i, x_j|y)}{P(x_i|y)P(x_j|y)} \qquad V(X_i; X_j) = \sqrt{\frac{\chi^2}{n(k-1)}}$$

Equation 3. CMI and Cramer's V Calculation

To determine the super parent using CMI, we will sum the CMI values between an attribute x_i and all other attributes, selecting the attribute with the highest overall CMI. To determine the super parent using Cramer'sV, we will use the attribute that is most highly associated with the class attribute.

2) Fine tuning the degree of smoothing by finding an optimal value of the smoothing parameter *k*:

We will implement additive smoothing to handle unseen attribute-label pairs in Naive Bayes. By adding a constant k to the numerator and k * V (where V is the number of possible attribute values) to the denominator, we ensure no probability equals zero. The smoothing degree is controlled by k: larger k values are suited for small datasets with many unseen pairs, while smaller k values are better for larger datasets. We will test multiple k values to find the optimal one for our dataset.

3) Determining a number of weight generations (m) and or an improvement threshold (T) in the AISWNB algorithm that optimizes accuracy and training time:

AISWNB will either create m generations of weight vectors or generate them until the improvement between generations drops below a threshold T. Higher m and lower T values lead to more generations, improving accuracy but increasing training time. Our goal is to find m and T values that balance accuracy and training time, which will be determined experimentally.

Because our objective is to design an algorithm that improves the predictive success of Naive Bayes in situations when attributes are not necessarily independent, in each of the following experiments, we selected the parameters and algorithms that optimized accuracy. Accuracy is given by:

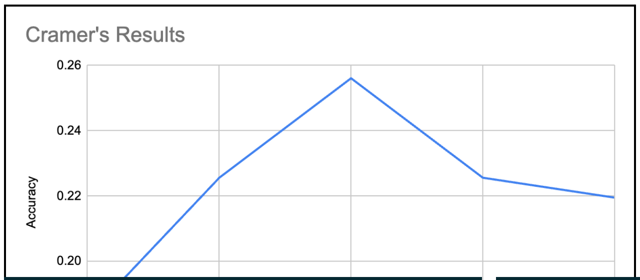
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

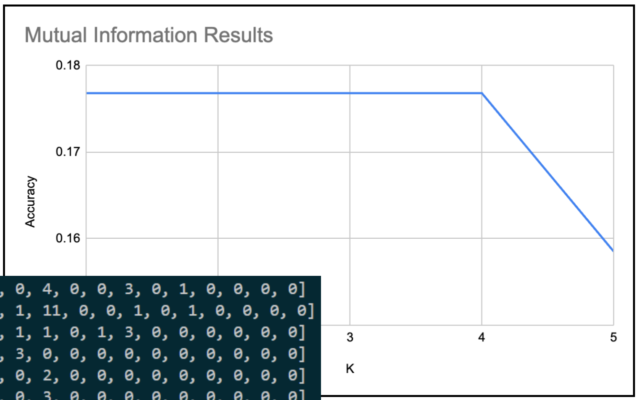
Equation 4. Accuracy Formula

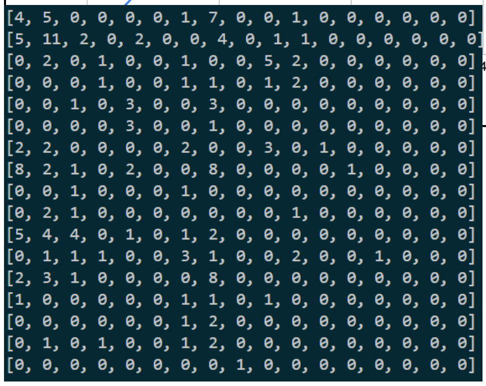
Experiment 1 & 2:

Because the optimal value of the smoothing parameter *k* may be contingent on the superparent selection algorithm used (either CMI or Cramer'sV) we decided to conduct experiment 1 and experiment 2 simultaneously. For each superparent selection method, we tested multiple values of *k* in order to ascertain the optimal smoothing parameter for each method. We then compared the highest accuracies of each algorithm.

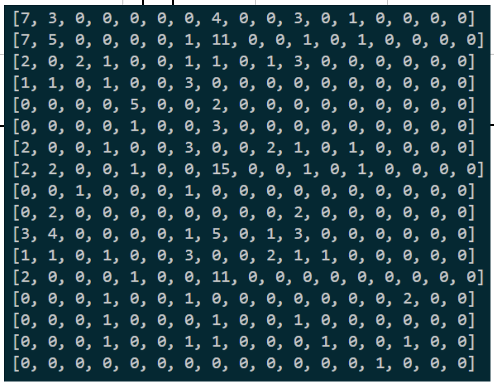
Figure 1. Accuracy of Superparent modal for different k-values given a parent selected by a) Cramer'sV or b) CMI.







```
[4, 5, 0, 0, 0, 0, 1, 7, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[5, 11, 2, 0, 2, 0, 0, 4, 0, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 2, 0, 1, 0, 0, 1, 0, 0, 5, 2, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 2, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 3, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 3, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[2, 2, 0, 0, 0, 0, 2, 0, 0, 3, 0, 1, 0, 0, 0, 0, 0]
[8, 2, 1, 0, 2, 0, 0, 8, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 2, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[5, 4, 4, 0, 1, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 1, 1, 0, 0, 3, 1, 0, 0, 2, 0, 0, 1, 0, 0, 0]
[2, 3, 1, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 1, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
```



```
[7, 3, 0, 0, 0, 0, 0, 4, 0, 0, 3, 0, 1, 0, 0, 0, 0]
[7, 5, 0, 0, 0, 0, 1, 11, 0, 0, 1, 0, 1, 0, 0, 0, 0]
[2, 0, 2, 1, 0, 0, 1, 1, 0, 1, 3, 0, 0, 0, 0, 0, 0]
[1, 1, 0, 1, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 5, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[2, 0, 0, 1, 0, 0, 3, 0, 0, 2, 1, 0, 1, 0, 0, 0, 0]
[2, 2, 0, 0, 1, 0, 0, 15, 0, 0, 1, 0, 1, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0]
[3, 4, 0, 0, 0, 0, 1, 5, 0, 1, 3, 0, 0, 0, 0, 0, 0]
[1, 1, 0, 1, 0, 0, 3, 0, 0, 2, 1, 1, 0, 0, 0, 0, 0]
[2, 0, 0, 0, 1, 0, 0, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
```

Because the accuracies of both

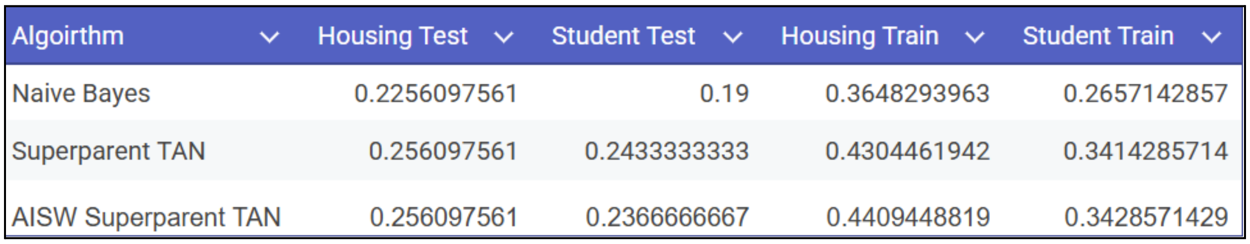
algorithms diminish after k=4, we only tested k-values in [1, 5]. For our dataset, Cramver'sV selected "Bathrooms" as the parent attribute while CMI selected "Area". For Cramer'sV, the algorithm achieved its highest accuracy (~25.6%) when k=3. For CMI, accuracy remained the same for k in [1, 4]. The confusion matrix for each parent selection algorithm when k=3 is given below:

Figure 2. Confusion Matrix of a) CMI (left) and b) Cramver's V (right)

Because we obtained our highest accuracy of ~25.6% using Cramer'sV when k=3, we decided to use Cramer'sV for our parent selection method, and k=3 for our smoothing parameter. Additionally, Cramer'sV is more computationally efficient to calculate than CMI, making it the optimal attribute selection method for our dataset.

Experiment 3:

To determine the optimal number of AISWNB generations, we ran AISWNB on its own and in combination with CMI and Cramer's. Originally, AISWNB did not work as intended, however there was improvement after debugging. The results are shown below:



Algoirthm	Housing Test	Student Test	Housing Train	Student Train
Naive Bayes	0.2256097561	0.19	0.3648293963	0.2657142857
Superparent TAN	0.256097561	0.2433333333	0.4304461942	0.3414285714
AISW Superparent TAN	0.256097561	0.2366666667	0.4409448819	0.3428571429

Table 2. Accuracy of each model on the Housing and Student Performance dataset.

As shown in the figure above, the accuracies increased with the training sets drastically, but did not repeat with the testing sets. This may be due to the fact that we already tested with one dependence, so AISWNB weighted the attributes twice. It might be better to create a matrix of weights instead, where instead of each attribute having a weight, each attribute and parent had a weight.

Conclusion & Future Work:

In our study, we determined that Cramer'sV was the optimal parent selection algorithm due to its computational efficiency and the fact that it selected a better parent than CMI. When using CMI as the parent selection algorithm, the optimal value of the smoothing parameter was k=3. Using AISWNB in addition to the Superparent TAN model did not add any meaningful contribution to our algorithm's performance.

In the future, we hope to improve our model's accuracy on more complex datasets, particularly those with higher level attribute dependencies. One solution is allowing for higher level dependencies (as seen in TAN models). Additionally, we would refine the adaptive weighting mechanism by exploring alternative optimization strategies such as by minimizing MSE or another metric. Finally, we would test the model on a diverse array of datasets to evaluate its flexibility.

Contributions:

Jia- Introduction, dataset and features, past work, abstract, conclusion and future work, preprocessed data, coded attribute selection aspect, experiments and discussion
James: Past work, methods, combined both parts of code for algorithm, coded one dependency aspect, experiments and discussion, conclusion and future work

References:

Webb, G., Boughton, J. & Wang, Z. Not So Naive Bayes: Aggregating One-Dependence Estimators. *Mach Learn* **58**, 5–24 (2005). https://doi.org/10.1007/s10994-005-4258-6

Jia Wu, Shirui Pan, Xingquan Zhu, Zhihua Cai, Peng Zhang, Chengqi Zhang, Self-adaptive attribute weighting for Naive Bayes classification, Expert Systems with Applications, Volume 42, Issue 3, 2015, Pages 1487-1502, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2014.09.019.

Zaidi, N. A., Cerquides, J., Carman, M. J., & Webb, G. (2013, January). Alleviating naive Bayes attribute independence assumption ... https://jmlr.org/papers/volume14/zaidi13a/zaidi13a.pdf

Lee, Y., O, S., & Eck, J. E. (2025). Improving Recidivism Forecasting With a Relaxed Naïve Bayes Classifier. Crime & Delinquency, 71(1), 89-117. https://doi.org/10.1177/00111287231186093

Frank, E., Hall, M., & Pfahringer, B. (2003). Locally weighted Naive Bayes. https://arxiv.org/pdf/1212.2487

Utilized Scikit for Test Train Split