Project Report COSC 320-001

Lilly Ngo, Brennen McCorison, Layla Phipps
Salisbury University
COSC 320: Advanced Data Structures and Algorithms
Dr. Lu
12/09/2024

Abstract

This project explores the implementation of Naïve Bayes, a probabilistic machine learning model, to predict heart disease using a dataset from the UCI Machine Learning Repository. The dataset combines categorical features, such as sex and chest pain type, with continuous variables like age and cholesterol levels, offering a unique opportunity to implement a multi-faceted approach to generate insights on the repository. Specifically, we utilized CategoricalNB for categorical features and a Kernel Density Estimation (KDE)-based model for continuous data, combining their outputs to improve predictive performance. Our goals were to balance computational and classification accuracy while addressing the practical challenges of working with real-world datasets and implementing solutions in a collaborative environment at an undergraduate level. Results showed that the hybrid model implementation leveraged the strengths of both Naïve Bayes (such as its efficacy with rare categories and its ability to efficiently use large and robust sets of data) and Kernel Density Estimation (with its ability to handle continuous features of datasets), created a very powerful way to pinpoint an efficient application in the use of healthcare.

Introduction

The integration of Artificial Intelligence (AI), more specifically Machine Learning (ML), is increasingly prevalent in a multitude of daily applications. In the healthcare sector, this technology has revolutionized disease prediction, diagnosis, and treatment planning. AI-driven models have the unique ability to analyze vast amounts of patient data, uncover hidden patterns, and provide accurate predictions, significantly aiding medical professionals in making critical decisions. With heart disease being a leading cause of death worldwide, developing accurate and efficient predictive tools is crucial in addressing this pressing health challenge.

The high-pressure nature of healthcare environments often leads to work-related fatigue, which can result in diagnostic errors, reduced efficiency, and patient dissatisfaction. Incorporating ML algorithms into routine medical processes aims to alleviate these pressures on healthcare workers. Automating certain aspects of disease prediction not only reduces the cognitive load on healthcare professionals but also ensures that decisions are data-driven and consistent, prioritizing both the quality of patient care and the wellbeing of healthcare professionals.

This project explores the application of the Naïve Bayes algorithm, a model renowned for its simplicity, interpretability, and computational efficiency. Naïve Bayes is particularly well-suited for healthcare applications that require quick, explainable predictions. Its ability to handle both varying qualities of data makes it effective for analyzing complex and mixed datasets, such as those related to heart disease risk factors. Inspired by previous successes of Naïve Bayes in diagnosing conditions like diabetes, we propose an approach that combines CategoricalNB and a Kernel Density Estimation (KDE)-based method. By evaluating this model's performance, we aim to highlight its potential in reducing diagnostic errors, improving patient outcomes, and advancing AI-driven healthcare.

Methodology

In this study, we seek to implement and evaluate the performance of machine learning models, more specifically, Naïve Bayes, in predicting heart disease using a widely recognized dataset. Our approach involves preprocessing the data, implementing Naïve Bayes models, and analyzing the results to identify key insights and challenges.

This study utilized the Heart Disease data set from the UCI Machine Learning Repository (ID: 45), a widely recognized benchmark for evaluating classification algorithms in healthcare predictive modeling. This dataset contains 303 instances and 76 attributes, but most published works, including ours, utilize a commonly referenced subset of 14 attributes, including *age*, *sex*, chest pain type (*cp*), resting blood pressure (*trestbps*), serum cholesterol (*chol*), fasting blood sugar (*fbs*), resting electrocardiographic results (*restecg*), maximum heart rate achieved (*thalach*), exercise-induced angina (*exang*), ST depression induced by exercise (*oldpeak*), slope of the peak exercise ST segment (*slope*), number of major vessels colored by fluoroscopy (*ca*), thalassemia (*thal*), and the target variable heart disease presence (*num*). The target variable represents the presence (1) or absence (0) of heart disease. The dataset was chosen for its popularity, relevance to real-world healthcare applications, and ability to provide a balanced challenge for machine learning models like Naïve Bayes. Its combination of categorical and continuous variables makes it ideal for testing model performance under diverse conditions. Additionally, the presence of missing values reflects common challenges in medical data, highlighting the importance of addressing such issues when building robust predictive systems.

The performance of the Naïve Bayes classifier was evaluated with multiple using for a comprehensive assessment:

- **Accuracy:** Measures the percentage of correct predictions on the test data. This is the primary metric for evaluating the overall performance of the model.
- **Model Comparisons:** The performance of the Naïve Bayes classifier was compared against variations of the model, including:
    - **Categorical Naïve Bayes (C-NB):** For categorical features.
    - **Kernel Density Estimation Naïve Bayes (KDE-NB):** For continuous features using kernel density estimation.
    - **Combined Model:** A hybrid of C-NB and KDE-NB that combines the log-probabilities of both models to improve prediction accuracy.

Naïve Bayes Classifier Models

We employed three variants of the Naïve Bayes classifier: **Categorical Naïve Bayes (C-NB)**, **Kernel Density Estimation Naïve Bayes (KDE-NB)**, and a **Combined Naïve Bayes (C-NB + KDE-NB)** model. Each model was trained to classify heart disease outcomes based on the 14 selected features present in the dataset. These models were chosen based on the nature of the data, as detailed below.

The **Categorical Naïve Bayes (C-NB)** classifier is well-suited for datasets with categorical features. In this approach, each feature is treated as a categorical random variable, and the classifier assumes that all features are *conditionally independent* given the class label. Categorical features, such as gender or scale of symptoms benefit from this model, as it works under the assumption of conditional independence. The likelihood of each feature is computed based on the frequency of occurrences in the training data, and the log-probability of the target class is then calculated. However, the conditional independence assumption may not always hold, which can lead to performance degradation when features are correlated and, C-NB cannot be applied to continuous features directly, limiting its flexibility when dealing with mixed data types - leading to our use of a hybrid implementation approach.

The **Kernel Density Estimation Naïve Bayes (KDE-NB)** model was employed to handle *continuous features*. KDE is a non-parametric method that estimates the probability density function (PDF) of a feature by smoothing its values using kernel functions, such as a Gaussian kernel. KDE allows for the estimation of the feature distribution without assuming a specific parametric form, making it especially useful for datasets where the distribution of features is unknown or non-normal. For continuous features, such as age or cholesterol levels, KDE provides a flexible and effective way to estimate the underlying distribution, which is particularly useful when the data does not follow a standard distribution like Gaussian. For each continuous feature and each class, a **probability density function** is estimated using KDE. The likelihood of each feature belonging to a particular class is then calculated based on the learned KDE.

The **Combined Naïve Bayes (C-NB + KDE-NB)** model integrates the predictions from both the **C-NB** and **KDE-NB** models. In this hybrid approach, the log-probabilities from both the categorical and continuous models are summed, and the class with the highest combined log-probability is selected as the final prediction. This approach allows for the incorporation of

both categorical and continuous features into a single model. The combined model leverages the strengths of both C-NB (for categorical features) and KDE-NB (for continuous features), making it ideal for datasets that contain mixed data types. By combining both models, we enable each type of feature to contribute to the prediction, potentially improving the model's overall accuracy.

Model Implementation

The Naïve Bayes models were implemented using Python, with libraries like *scikit-learn*, *numpy*, and *pandas* forming the backbone of the process. Categorical features were encoded using *LabelEncoder* to convert non-numeric data into numeric format, while continuous features were scaled where necessary to standardize their ranges. Missing values, which were present in only one feature (ca), were addressed by calculating the median and mode values and importing them into the dataset to ensure completeness.

The dataset was split into features (X) and target labels (y), with the features further divided into categorical and continuous subsets. This ensured that the models could be trained and evaluated on distinct data subsets to minimize overfitting and provide an unbiased performance assessment. The training process was conducted separately for the categorical and continuous models, followed by integration into the combined model. For the Categorical Naïve Bayes (C-NB) model, only the categorical features from the training dataset were used. The CategoricalNB implementation from scikit-learn was employed to calculate likelihoods for each feature based on their frequencies in the training data. This model assumes conditional independence between features, which simplifies the computation of probabilities and enables efficient training. To handle continuous features, the Kernel Density Estimation (KDE-NB) model was implemented, and the training set was processed using *scipy.stats.gaussian_kde*, which estimates the probability density function (PDF) of each feature for each class. For every class, the KDE model fits a separate density function for each continuous feature using the corresponding training data points. The Combined Naïve Bayes (C-NB + KDE-NB) model was designed to integrate both categorical and continuous data for a more comprehensive classification. This hybrid approach combined the log-probabilities from the C-NB and KDE-NB models for each class.

Results

The Naïve Bayes classifier demonstrated strong performance in predicting heart disease, achieving an accuracy of 92% on the test dataset. Precision and recall were also notable at 91% and 89%, respectively, indicating the model's ability to make reliable and balanced predictions. These results highlight the effectiveness of Naïve Bayes in leveraging both categorical and continuous features to produce accurate predictions in this healthcare application. A key strength of Naïve Bayes is efficiency and simplicity, especially with smaller datasets or those with a mix of feature types, as seen in UCI's Heart Disease dataset. By combining categorical and continuous features through a hybrid approach, we capitalized on the strengths of both Categorical Naïve Bayes and Kernel Density Estimation-based Naïve Bayes. This integration allowed the model to adapt well to the dataset's characteristics, making it a robust choice for predictive modeling in this context.

Although the feature independence assumption of Naïve Bayes may not fully align with the interdependent nature of real-world healthcare data, the model still performed exceptionally well. This demonstrates that the Naïve Bayes classifier is highly effective when the focus is on computational efficiency and strong baseline performance. Its ability to handle imbalanced data and missing values further highlights its utility in healthcare predictive modeling, where such challenges are common. In conclusion, Naïve Bayes remains a highly competitive and reliable choice for predicting the presence or absence of heart disease, offering a strong balance of accuracy, simplicity, and interpretability.

Conclusion

This project allowed us to successfully implement a hybrid Naïve Bayes classifier to predict heart disease, achieving a strong accuracy of 92%, with precision and recall of 91% and 89%, respectively. These results reveal the model's ability to balance computational efficiency with classification accuracy, especially in a dataset with both categorical and continuous features. By integrating Categorical Naïve Bayes and Kernel Density Estimation, we addressed realistic challenges in handling real-world data, demonstrating the algorithm's versatility and potential for healthcare applications and more. The project's primary objective—to deepen our understanding of machine learning algorithms—was met through a hands-on exploration of Naïve Bayes and its variations. Beyond achieving strong performance metrics, the experience reinforced key concepts of ML such as probabilistic modeling, feature independence, and hybrid approaches to

mixed data. These findings highlight the versatility and efficacy of simple yet powerful models in addressing complex and important challenges, such as improving diagnostic accuracy in healthcare. Moving forward, exploring the scalability of the hybrid model with larger datasets and comparing its performance to advanced classifiers could offer further insights. This project illustrates the critical role of machine learning in advancing AI-driven healthcare solutions, exemplifying the potential for data-driven, efficient, and comprehendable tools to enhance decision-making in medical contexts.

Future Directions

In this study, we sought to evaluate the performance of Naïve Bayes classifiers on heart disease prediction using a mix of categorical and continuous features. While our original models performed well, additional steps were taken to address the independence assumption inherent in Naïve Bayes, particularly for features that exhibit dependencies in real-world healthcare scenarios. To combat the independence assumption, we implemented a Pearson correlation matrix for continuous features, visualized through a heatmap, to identify and weight significant relationships between variables. For example, correlations such as *age* and *thalach* (0.25) or *ca* and *thalach* (0.17) were used to inform feature importance during classification. Similarly, for categorical variables, Cramér's V was calculated to measure the strength of associations, such as the notable relationship between *cp* (chest pain type) and *exang* (exercise-induced angina), with a score of 0.82. These methods enabled us to incorporate feature relationships into the weighted Naïve Bayes models.

Despite these enhancements, the hybrid model's weighted accuracy was approximately 84%, lower than the unweighted combined model's accuracy of 92%. This result suggests that the weighting approach introduced complexity without significantly improving performance. It highlights that while weighting relationships makes the model more reflective of real-world interdependencies, the simplistic assumptions of Naïve Bayes may not be well-suited for fully capturing these relationships. If the goal is to accurately account for the dependencies between all features, exploring alternative models—such as Random Forests or Gradient Boosted Trees—may be a more effective direction. These models can inherently manage feature dependencies without requiring manual adjustments.

References

Github - Coding Collaborative Process/Repository

Janosi, A., Steinbrunn, W., Pfisterer, M., & Detrano, R. (1989). Heart Disease [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C52P4X.

KDnuggets. (2020, June). The Naive Bayes Algorithm: Everything You Need to Know. https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html

Simplilearn. (n.d.). Naive Bayes Classifier: A Complete Guide for Machine Learning. https://www.simplilearn.com/tutorials/machine-learning-tutorial/naive-bayes-classifier#understanding_naive_bayes_and_machine_learning

Investopedia. (n.d.). Bayes' Theorem Definition. https://www.investopedia.com/terms/b/bayes-theorem.asp

Larrañaga, P., Sierra, B., Gallego, M.J., Michelena, M.J., Picaza, J.M. (1997). Learning Bayesian Networks by Genetic Algorithms: A case study in the prediction of survival in malignant skin melanoma. https://doi.org/10.1007/BFb0029459

Khan, A., Khan, A., Khan, M. M., Farid, K., Alam, M. M., & Su'ud, M. B. M. (2022). Cardiovascular and Diabetes Diseases Classification Using Ensemble Stacking Classifiers with SVM as a Meta Classifier. https://doi.org/10.3390/diagnostics12112595

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

Evaluations - Layla Phipps

K Means Clustering

The K Means clustering presentation, in my opinion, was well done and easy to understand. I liked how they explained step by step how clustering works, why things may be clustered together, and how to cluster effectively. I had no questions on this prior to reading their slides. I also find it interesting how K has to be chosen perfectly to prevent overfitting or underfitting, which I imagine poses one of the biggest challenges in the application of the algorithm, especially with data that may not be *absolutely perfect* for the algorithm, as I imagine most data sets will be.

I think that one way to improve this would have been to show or explain specific examples in which this is often used rather than the blanket statements “Image Compression, Search Result Grouping, etc etc”. *Why* is K means clustering used for these things? Can you tie it back to a real world application to better make sense of the algorithm? Also, there was no comparison to other algorithms which I like to see- is it on the fast or slow side? What other algorithms is it similar to? Etc etc.

Breadth First Search

I don’t think I learned too much from this presentation since I’m already familiar with BFS, but it was interesting to see the intricacies of computing time given different tree sizes, especially with threads from parallel computing. The graphs were very easy to read and showed a lot of information that was easy to digest as well.

I would make quite a few changes to this presentation. The text is not well organized, and is very simplistic. There is not much meat to any of the text in the presentation and the first few slides, in my opinion, don’t explain their application of BFS very well. “Intro to BFS” includes a few bullet points on the foundation of what BFS is and how it may work but are unorganized and don’t transition well from one thought to another. The OpenMP slide is also confusing to me, as “omp critical” and “schedule dynamic” are there and not elaborated upon, and I have no idea what either of those things mean. The “Graph Creation” slide is also lackluster- it does explain how the tree is created but not why those steps are taken or the logic behind it.

Parallel Merge Sort

I very much liked this presentation. I think it’s comprehensive and explains the process and why each step was taken in order to come to the result. It opens with a brief explanation of

merge sort in general, which was helpful to me as I haven't used it or implemented it in a while and was rusty on some of the concept and how it worked. The next slide for applications was interesting to look at as well, as there are many examples and the explanations as to why merge sort is used in these applications are complete and understandable as well.

I also was very glad that this presentation included a slide on what parallel computing is at its very core, as I was unfamiliar with it before these presentations. The explanation of the use of threads and OpenMP was very helpful for me to piece together how parallel computing would work with this algorithm as well. I also think the delving into recursively implementing merge sort versus iteratively implementing it with parallel computing is interesting as it allowed me to pull prior memory as to how recursion is often much slower than iteration and allowed me to apply that knowledge in this context. If I were to improve this presentation I would maybe have split up the talking points between the two presenters more evenly.

Matrix Multiplication

This presentation did a good job explaining key concepts, especially the more complex concepts, of matrix multiplication in both sequential and parallel implementations. It uses usually clear and good quality diagrams and examples to explain how matrix multiplication works, which made it very easy for me to follow. It also highlights the benefits of parallel computing without overexplaining, such as faster computation for large data sizes, which makes it relevent to real-world applications.

In my opinion, some slides have too much text, which overwhelmed me a bit and made it harder to focus on the key points during the presentation. Also, the demonstration results could include more detailed analysis or visual aids, like graphs, to make the comparisons between sequential and parallel implementations easier to interpret.

Prims Algorithm

This presentation provides a straightforward explanation of Prim's Algorithm, with an easy-to-follow step-by-step breakdown which I appreciated as that is the best way I can process and retain information, especially with algorithms. The inclusion of real-world applications such

as network and circuit design helps show the algorithm's importance. The visual example of a graph helps to drive home the point and show context, making it easy to digest to some of us with less experience with the algorithm.

The presentation could benefit from more detailed examples or illustrations to make the algorithm's process more memorable. Additionally, the explanation of why parallel processing is useful is too brief and lacks depth in my opinion, which makes that part of the presentation much weaker.

Parallel computing for large data summation

This document effectively explains the advantages of parallel computing and effectively shows its advantages like its speed, scalability, and utilization of multi-core systems. The step-by-step breakdown of how parallel summation works is clear and easy to understand for all learners in my opinion. Additionally, the inclusion of real-world applications like e-commerce and financial systems shows the practical value of this application.

The performance comparison lacks detailed visual aids or numerical results to support the claims of improved efficiency. Also, the challenges, such as overhead and load balancing, are mentioned but not explored hardly at all, which left me with a limited understanding of how these issues can be tackled in practice.

Sieve

The presentation clearly and concisely explains of the Sieve of Eratosthenes, including step-by-step examples that make it easy to understand. The comparison between sequential and parallel implementations is helpful for demonstrating why you'd choose one over the other with their strengths and weaknesses. Including pseudocode helps me digest code and how it works much better than the code itself, so I really appreciated this approach to tackling the code end of the implementation

The sequential vs. parallel results are mentioned but dont include detailed data or visuals such as graphs to show the performance differences well. Also, some slides are text-heavy, which made them less engaging during the presentation and harder to remember. Breaking down the content visually or through a more concise summary would me more memorable and easier to digest in my opinion.

Machine Learning Linear Regression

This presentation provides a good explanation of linear regression including its assumptions, real-world applications, and mathematical foundation. It, in my opinion, introduces gradient descent very well and explains its significance and challenges perfectly, which helped deepen my understanding. The inclusion of real-world applications makes the content easier to digest and emphasizes its practical importance as well.

Some slides are overly text-heavy, which made the presentation harder to follow for me. Also, while the faults of linear regression are mentioned, the explanation is brief and doesnt have many

examples to show how the limitations impact its performance in real life scenarios, which left me with a half understanding of when/why you'd use this strategy versus others.

Decision Trees

The presentation provides a solid explanation of decision trees (or at least I understood it well), covering how its structured, how they work, and real-world applications. The comparison between programmer-selected and data-driven splits is useful for understanding how decision trees adapt to different datasets, which is something I was confused about at first. Including a specific example, like the poker decision tree, added a lot to the presentation by showing how the algorithm can be applied in real-life scenarios.

Some slides could benefit from clearer visual aids to complement the text-heavy explanations, especially with the calculation of splits. Also, the presentation does not address common limitations of decision trees, such as overfitting or the need for pruning (it was talked about in person, but I always appreciate all pieces being present in the slides for future reference as well), which would provide a more balanced understanding of the algorithm for me.