Data Mining Project

Name: Abhinab Chakraborty

Registration Number: CU23MSD0001A

M.Sc. Data science

Course: Data Mining

CHANAKYA UNIVERSITY

Assignment Guide

Prof. Usha Subramanian

Professor of Practice

School of Engineering

CHANAKYA UNIVERSITY, BANGALORE

Machine Learning

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed. Supervised learning and unsupervised learning are two main types of machine learning.[1]

1. Supervised machine learning

2. Unsupervised machine learning

Supervised Machine Learning Algorithm

Supervised learning is a type of machine learning algorithm that learns from labelled data. Labelled data is data that has been tagged with a correct answer or classification.

Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Supervised learning is when we teach or train the machine using data that is well-labelled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data (set of training examples) and produces a correct outcome from labelled data.[1]

Supervised machine learning algorithm is further divided into Classification machine learning algorithm and Regression machine learning algorithm.

In this data mining project, we are going to focus specially on Gaussian Naïve Bayes and K-Nearest Neighbor classification supervised machine learning algorithm.

Gaussian Naïve Bayes

Introduction

Gaussian Naïve Bayes is a variant of the Naïve Bayes algorithm, which is based on Bayes' theorem with an assumption of independence between predictors. The Gaussian Naïve Bayes classifier specifically assumes that the continuous values associated with each class are distributed according to a Gaussian (normal) distribution.

Bayes Theorem

Bayes' theorem is a fundamental concept in probability theory and statistics that describes how to update the probability of a hypothesis based on new evidence. It's named after Thomas Bayes, an 18th-century statistician and minister who first proposed the basic form of the theorem. Let's dive into a detailed discussion of Bayes' theorem, its components, applications, and significance.

The Theorem: Bayes' theorem is expressed mathematically as:

$P(A|B) = [P(B|A) * P(A)] / P(B)$

Where:

- P(A|B) is the posterior probability: the probability of event A occurring given that B is true.
- P(B|A) is the likelihood: the probability of event B occurring given that A is true.
- P(A) is the prior probability: the initial probability of A before considering B.
- P(B) is the marginal likelihood: the probability of observing event B.

Components of Bayes' Theorem:

1. Prior Probability: This is our initial belief about the probability of an event before considering new evidence. It represents our starting point or baseline assumption.
2. Likelihood: This term quantifies how probable the observed evidence is, assuming our hypothesis is true. It measures the compatibility between the evidence and the hypothesis.
3. Marginal Likelihood: Also known as the evidence, this term represents the probability of observing the evidence regardless of whether the hypothesis is true or false.
4. Posterior Probability: This is the updated probability of our hypothesis after considering the new evidence. It's the main output of Bayes' theorem and represents our revised belief.

Naïve Bayes

Naïve Bayes extends this concept to classification problems, where we want to find the probability of a class given a set of features. It's "naïve" because it assumes that all features are independent of each other, which simplifies the calculations but isn't always true in real-world scenarios.

Gaussian Naïve Bayes

Gaussian Naïve Bayes specifically deals with continuous data, assuming that the values associated with each class follow a Gaussian distribution.

Key Assumptions

1. Feature Independence: The features are assumed to be independent of each other given the class.

2. Gaussian Distribution: For each class, the values of each feature are assumed to be normally distributed.

Mathematical Formulation

For a feature vector $X = (x_1, x_2, ..., x_n)$ and a class variable y, the Gaussian Naïve Bayes classifier calculates:

$P(y|X) \propto P(y) \prod_i P(x_i|y)$

Where $P(x_i|y)$ is the probability density of the Gaussian distribution:

$P(x_i|y) = (1 / \sqrt{(2\pi\sigma^2_y)}) * \exp(-(x_i - \mu_y)^2 / (2\sigma^2_y))$

Here:

- μ_y is the mean of the feature for class y
- σ^2_y is the variance of the feature for class y

Training the Model

Training a Gaussian Naïve Bayes model involves:

1. Calculating the prior probability P(y) for each class.
2. For each feature in each class, calculating the mean μ_y and variance σ^2_y.

Classification

To classify a new instance:

1. Calculate $P(y) \prod_i P(x_i|y)$ for each class.
2. Assign the instance to the class with the highest probability.

Advantages and Disadvantages

Advantages

1. Simple and fast to train and predict
2. Works well with small datasets
3. Handles high-dimensional data efficiently
4. Not sensitive to irrelevant features

Disadvantages

1. Assumes feature independence, which is often not true
2. Limited by the Gaussian assumption for feature distributions
3. Can be outperformed by more sophisticated models on complex datasets

Applications

Gaussian Naïve Bayes is used in various fields, including:

1. Text classification and spam filtering
2. Sentiment analysis
3. Recommendation systems
4. Medical diagnosis
5. Weather prediction

K-Nearest Neighbors (KNN) Classification

Introduction

K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm used for classification and regression. In KNN classification, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.

Core Concept

The fundamental idea behind KNN is that similar things exist in close proximity. In other words, similar data points are likely to be close to each other in a feature space.

Algorithm Steps

1. Choose the number K of neighbors
2. Calculate the distance between the query instance and all training samples
3. Sort the distances and determine the K nearest neighbors
4. Gather the category of the nearest neighbors
5. Use simple majority of the category of nearest neighbors as the prediction value

Distance Metrics

KNN relies on distance metrics to determine the "closeness" of data points. Common distance metrics include:

1. Euclidean Distance: $\sqrt{(\Sigma(x_i - y_i)^2)}$

2. Manhattan Distance: $\Sigma|x_i - y_i|$
3. Minkowski Distance: $(\Sigma|x_i - y_i|^p)^{1/p}$
4. Hamming Distance (for categorical variables): $\Sigma(x_i \neq y_i)$

Choosing K

The choice of K is crucial:

- Small K: More sensitive to noise
- Large K: More computationally expensive and can blur the boundaries between classes

Techniques for choosing K:

1. Cross-validation
2. Grid search
3. Domain knowledge

Advantages

1. Simple to understand and implement
2. No assumptions about data distribution
3. Effective for multi-class problems
4. Can be used for both classification and regression

Disadvantages

1. Computationally expensive for large datasets
2. Sensitive to irrelevant features and the scale of the data
3. Requires feature scaling
4. Does not work well with high-dimensional data (curse of dimensionality)

Variants and Improvements

1. Weighted KNN: Gives more weight to closer neighbors
2. Condensed KNN: Reduces the dataset size
3. Local Weighted Average: Uses kernel regression for smoother boundaries

Mathematical Formulation

For a given test point x:

1. Calculate distances to all training points: d (x, x_i) for all x_i in the training set
2. Sort distances and select K nearest neighbors
3. For classification, use majority voting: $\hat{y}$ = argmax_y Σ I(y = y_i), where I is the indicator function

Difference Between Gaussian Naïve Bayes and K-Nearest Neighbors

Aspect	Gaussian Naive Bayes	K-Nearest Neighbors
Algorithm Type	Probabilistic classifier based on Bayes' theorem	Non-parametric, instance-based learning algorithm
Assumptions	Independence among features, Gaussian distribution	No specific assumptions about the data distribution
Training Process	Computes mean and variance for each class	No explicit training phase; stores training data
Prediction Process	Calculates posterior probabilities and chooses the highest	Finds k-nearest neighbors and classifies based on majority vote
Performance with Noisy Data	Generally robust to noise	Sensitive to noise, can lead to misclassification
Scalability	Scales well to large datasets	Doesn't scale well to large datasets
Interpretability	Provides interpretable probabilities and model parameters	Less interpretable, relies on distance calculations
Use Cases	Text classification, spam detection, etc.	Pattern recognition, image classification, etc.

Methodology

- **Importing Data:** We are given two datasets (red wine and white wine) having similar features i.e. density, pH, alcohol, volatile, acidity, residual sugar, fixed acidity, citric acid, chlorides & quality. The red wine data consists of 1599 rows & white wine data consists of 4898 rows with 12 similar features. The red wine data has 6 qualities as output whereas white wine data has 7 qualities as output.
- **Data preprocessing:** Once the raw data is collected, there are a few crucial steps that are performed before the data is used for modelling. Steps like data cleansing, normality check, outlier detection and removal are crucial in order to make the data more precise and increase the efficiency of the data analysis in turn making the model much more accurate. The pre-processed data is then ready to be used for the next phase of the study, which involved the application of data mining algorithms to build models that could accurately predict output.

1. Cleaning data: This is the initial step in data-preprocessing. Firstly, we have separated the features into different columns based on semi-colon. In the given dataset we checked for null values. The data type of different columns was checked. There were no missing values. As a result, an appropriate dataset with following attributes columns was obtained.

2. Outlier removal: We began by addressing the issue of outliers in our dataset, which were significantly impacting our test and train results. Outliers can skew statistical analysis & model performance. So, their detection and removal was a wise initial step. We have used interquartile range through box plot to eliminate those extreme values. By eliminating those extreme vales, we have observed that rows with quality 3 and 8 of red wine dataset and rows with quality 3, 8 & 9 of white wine dataset has been eliminated completely. By removing outliers, we have created a more representative dataset that better reflects the true underlying patterns in our data.

3. Normality Check: Following the outlier removal, we also conducted a normality check on our features for both datasets. This step is often overlooked but is crucial for many statistical methods & data mining algorithms. We have used density plot to check the normality of features & found that only 3 out of 11 features i.e. density, pH and alcohol were normally distributed, other features like volatile, acidity, residual sugar, fixed acidity, citric acid & chlorides appear to be more skewed or have multiple peaks, indicating that they do not follow a normal distribution as closely.

Normally distributed features often allow for the use of parametric statistical methods & certain data mining algorithms that assume normality. However, the fact

that most of our features do not follow normal distribution suggests that non-parametric methods might be more appropriate for our dataset.

- **Building Model:** With this pre-processed data, we proceeded to apply two different classification algorithms: Gaussian Naive Bayes and K-Nearest Neighbors (KNN) considering three normally distributed features (Density, pH & alcohol) for both the dataset. The Gaussian Naive Bayes algorithm is a probabilistic classifier that assumes the features follow a normal (Gaussian) distribution. It's often used for its simplicity and efficiency, particularly in text classification tasks. However, its assumption of feature independence and normality can be limiting in many real-world scenarios. On the other hand, KNN is a non-parametric method that doesn't make strong assumptions about the underlying data distribution. It classifies new data points based on the majority class of their k nearest neighbors in the feature space.

Result:

Red wine dataset

After splitting the data into training data and test data, the training data contains 1119 rows and 12 columns, whereas test data contains 480 rows and 12 columns. After removing outliers, the class labels 3 & 8 got eliminated from training data as well as test data.

For Gaussian Naïve Bayes, after calculating prior probability for train data quality 5 has the highest probability i.e. 0.44 followed by quality 6, 7 and 4. But for test data quality 6 has the highest probability i.e. 0.45 followed by quality 5, 7 and 4.

After calculating posterior probability for each quality in training data and test data and normalizing it we have observed that quality 5 has the highest posterior probability among all.

For KNN, we have considered normally distributes features same as gaussian Naïve Bayes and predicted the quality for the new data point. The KNN model has predicted the class of new data point as quality 5.

White wine dataset

After splitting the data into training data and test data, the training data contains 3428 rows and 12 columns, whereas test data contains 1470 rows and 12 columns. After removing outliers, the class labels 3, 8 & 9 got eliminated from training data as well as test data.

For Gaussian Naïve Bayes, after calculating prior probability for train data quality 6 has the highest probability i.e. 0.42 followed by quality 5, 7 and 4. Same as train data, the test data also has quality 6 as the highest probability i.e. 0.49 followed by quality 5, 7 and 4.

After calculating posterior probability for each quality in training data and test data and normalizing it we have observed that quality 5 has the highest posterior probability among all.

For KNN, we have considered normally distributes features same as gaussian Naïve Bayes and predicted the quality for the new data point. The KNN model has predicted the class of new data point as quality 6.

The experts have classified the quality of both types of wines and our results showed a clear performance difference between these two algorithms. KNN predicted the output accurately, while Gaussian Naive Bayes produced predictions with some errors. This outcome aligns well with what we might expect given the characteristics of your data. Since only a small portion of our features exhibited normal distribution, the fundamental assumption of Gaussian Naive Bayes was violated for most of our dataset. This violation likely contributed to its poorer performance. Conversely, KNN's ability to capture local patterns in the data without making distribution assumptions allowed it to perform well despite the non-normal nature of most features.

Further Improvement

These findings lead to several important conclusions and suggest some next steps for our analysis.

Firstly, the observation that both Gaussian Naive Bayes and K-Nearest Neighbors (KNN) algorithms predict the same output for both training and test data leads to an intriguing conclusion. This consistency across two fundamentally different algorithms suggests that the underlying patterns in your dataset are likely strong and well-defined. It may indicate that the classification task is relatively straightforward, with clear decision boundaries in the feature space. The identical performance could imply that your features are highly relevant to the classification task, allowing both algorithms to utilize them effectively. However, this uniformity also raises some points for consideration. There's a slight possibility of overfitting, though this is less likely given that it's occurring with two different algorithms. It's also worth investigating the potential for data leakage or checking if the chosen value of K in KNN is limiting its ability to capture complex, nonlinear relationships that might exist in the data. While this consistency could be seen as a sign of robustness in your

models, it's unusual enough to warrant further investigation. Consider trying other algorithms, examining your feature space, checking class distributions, and verifying your train-test split. If after thorough investigation you confirm there are no issues with your data or methodology, this consistency could indeed be a positive indication of clear, easily discernible patterns in your dataset. Nonetheless, it's advisable to consult with domain experts to ensure this aligns with the expected complexity of your classification task.

Furthermore, while our current approach of removing outliers has improved results, it's worth considering whether some of these outliers might contain valuable information. Exploring robust algorithms that can handle outliers without removing them entirely could provide additional insights. Techniques using algorithms less sensitive to outliers (e.g., tree-based methods) might be worth investigating.

Conclusion

In conclusion, our methodical approach to data preprocessing and model comparison has yielded valuable insights into the nature of our data and the suitability of different classification algorithms. By continuing to refine our methods, exploring additional algorithms, and considering advanced techniques for handling non-normal data, you're well-positioned to develop a robust and accurate classification model tailored to your specific dataset.

Reference:

1) Greeks for Greeks (https://www.geeksforgeeks.org/supervised-unsupervised-learning/)

2) Fundamental Of Mathematical Statistics

3) Data Mining: Concepts and Techniques, Jiawei Han, Jian Pei and Hanghang Tong, 4th Edition, Elsevier, 2022.

Annexure:

1) Difference Between Gaussian Naïve Bayes and K-Nearest Neighbors