mwaglbnhn

September 13, 2023

[ ]:
```
# Q1. A company conducted a survey of its employees and found that 70% of the␣
↪employees use the company's health insurance plan, while 40% of the␣
↪employees who use the plan are smokers.

# What is the  probability that an employee is a smoker given that he/she uses␣
↪the health insurance plan?
```

Let's denote:

A: The event that an employee uses the health insurance plan. S: The event that an employee is a smoker.

You are given:

P(A): The probability that an employee uses the health insurance plan, which is 70% or 0.70.

P(S|A): The conditional probability that an employee is a smoker given that they use the health insurance plan, which is 40% or 0.40.

You want to find P(S|A), the probability that an employee is a smoker given that they use the health insurance plan.

P(A) * P(S|A) = 0.70 * 0.40 = 0.28

P(S A)= 0.70 / 0.28 = 0.4

[ ]:
```
# Q2. What is the difference between Bernoulli Naive Bayes and Multinomial␣
↪Naive Bayes?
```

Bernoulli Naive Bayes is used for binary data, like whether a word is present or absent in a document (0 or 1), often used in spam detection.

Multinomial Naive Bayes is for count-based data, such as word frequencies in text data, commonly used in tasks like document categorization.

[ ]:
```
# Q3. How does Bernoulli Naive Bayes handle missing values?
```

Bernoulli Naive Bayes handles missing values by treating them as neutral or "don't know" regarding the presence or absence of features. It continues to make predictions based on the available data and the probabilistic model built during training.

[ ]:
```
# Q4. Can Gaussian Naive Bayes be used for multi-class classification?
```

Yes, Gaussian Naive Bayes can be used for multi-class classification, where there are more than two classes or categories to predict.

[ ]:
```
# Q5. Assignment:

# Data preparation:

# Download the "Spambase Data Set" from the UCI Machine Learning Repository␣
 ↪(https://archive.ics.uci.edu/ml/datasets/Spambase).
# This dataset contains email messages, where the goal is to predict whether a␣
 ↪message is spam or not based on several input features.


# Implementation:

# Implement Bernoulli Naive Bayes, Multinomial Naive Bayes, and Gaussian Naive␣
 ↪Bayes classifiers using the scikit-learn library in Python.
# Use 10-fold cross-validation to evaluate the performance of each classifier␣
 ↪on the dataset.
# You should use the default hyperparameters for each classifier.


# Results:

# Report the following performance metrics for each classifier:

# Accuracy
# Precision
# Recall
# F1 score


# Discussion:

# Discuss the results you obtained.
# Which variant of Naive Bayes performed the best?
# Why do you think that is the case? Are there any limitations of Naive Bayes␣
 ↪that you observed?



# Conclusion:

# Summarise your findings and provide some suggestions for future work.
```

```
# Note:  This dataset contains a binary classification problem with multiple␣
 ↪features.
# The dataset is relatively small, but it can be used to demonstrate the␣
 ↪performance of the different variants of Naive Bayes on a real-world problem.
```

[1]:
```
import pandas as pd
import requests


url = "https://archive.ics.uci.edu/ml/machine-learning-databases/spambase/
 ↪spambase.data"


column_names = [
    "word_freq_make", "word_freq_address", "word_freq_all", "word_freq_3d",␣
 ↪"word_freq_our",
    "word_freq_over", "word_freq_remove", "word_freq_internet",␣
 ↪"word_freq_order", "word_freq_mail",
    "word_freq_receive", "word_freq_will", "word_freq_people",␣
 ↪"word_freq_report", "word_freq_addresses",
    "word_freq_free", "word_freq_business", "word_freq_email", "word_freq_you",␣
 ↪"word_freq_credit",
    "word_freq_your", "word_freq_font", "word_freq_000", "word_freq_money",␣
 ↪"word_freq_hp",
    "word_freq_hpl", "word_freq_george", "word_freq_650", "word_freq_lab",␣
 ↪"word_freq_labs",
    "word_freq_telnet", "word_freq_857", "word_freq_data", "word_freq_415",␣
 ↪"word_freq_85",
    "word_freq_technology", "word_freq_1999", "word_freq_parts",␣
 ↪"word_freq_pm", "word_freq_direct",
    "word_freq_cs", "word_freq_meeting", "word_freq_original",␣
 ↪"word_freq_project", "word_freq_re",
    "word_freq_edu", "word_freq_table", "word_freq_conference", "char_freq_;",␣
 ↪"char_freq_(",
    "char_freq_[", "char_freq_!", "char_freq_$", "char_freq_#",␣
 ↪"capital_run_length_average",
    "capital_run_length_longest", "capital_run_length_total", "is_spam"
]


response = requests.get(url)
data_lines = response.text.splitlines()


df = pd.DataFrame([line.split(",") for line in data_lines],␣
 ↪columns=column_names)
```

```
df.to_csv("spambase.csv", index=False)
```

[3]:
```
import pandas as pd
from sklearn.model_selection import cross_val_score
from sklearn.naive_bayes import BernoulliNB, MultinomialNB, GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score


df = pd.read_csv("spambase.csv")


X = df.drop("is_spam", axis=1)
y = df["is_spam"]


bernoulli_classifier = BernoulliNB()
multinomial_classifier = MultinomialNB()
gaussian_classifier = GaussianNB()

# Perform 10-fold cross-validation and calculate performance metrics
def evaluate_classifier(classifier, name):
    accuracy = cross_val_score(classifier, X, y, cv=10, scoring='accuracy').
 ↪mean()
    precision = cross_val_score(classifier, X, y, cv=10,␣
 ↪scoring='precision_weighted').mean()
    recall = cross_val_score(classifier, X, y, cv=10,␣
 ↪scoring='recall_weighted').mean()
    f1 = cross_val_score(classifier, X, y, cv=10, scoring='f1_weighted').mean()

    print(f"Results for {name} Naive Bayes:")
    print(f"Accuracy: {accuracy:.2f}")
    print(f"Precision: {precision:.2f}")
    print(f"Recall: {recall:.2f}")
    print(f"F1 Score: {f1:.2f}")
    print()
```

[4]:
```
evaluate_classifier(bernoulli_classifier, "Bernoulli")
evaluate_classifier(multinomial_classifier, "Multinomial")
evaluate_classifier(gaussian_classifier, "Gaussian")
```

```
Results for Bernoulli Naive Bayes:
Accuracy: 0.88
Precision: 0.89
Recall: 0.88
```

```
F1 Score: 0.88

Results for Multinomial Naive Bayes:
Accuracy: 0.79
Precision: 0.79
Recall: 0.79
F1 Score: 0.79

Results for Gaussian Naive Bayes:
Accuracy: 0.82
Precision: 0.86
Recall: 0.82
F1 Score: 0.82
```

1 Performance Comparison:

Bernoulli Naive Bayes: This classifier performed the best in terms of accuracy, precision, recall, and F1 score, achieving an accuracy of 0.88. It provides the most balanced performance across all metrics.

Multinomial Naive Bayes: While it performed reasonably well with an accuracy of 0.79, it falls slightly behind the Bernoulli variant in terms of accuracy and F1 score. This might be due to the dataset not naturally aligning with the assumptions of a multinomial distribution.

Gaussian Naive Bayes: This classifier achieved an accuracy of 0.82, which is also decent but not as high as Bernoulli Naive Bayes. It exhibits good precision but lags behind in recall compared to the Bernoulli variant.

2 Why Bernoulli Naive Bayes Performed Best:

Binary Features: Bernoulli Naive Bayes performed best because the "Spambase" dataset likely contains many binary features, such as the presence or absence of certain words or characters in emails. Bernoulli Naive Bayes is well-suited for binary data, making it a natural choice for this dataset.

3 Limitations of Naive Bayes:

Independence Assumption: Naive Bayes assumes that features are independent of each other, which may not hold in all real-world datasets.

Sensitivity to Data Quality: Naive Bayes can be sensitive to noisy or irrelevant features, potentially leading to suboptimal performance.

Limited Expressiveness: It may not capture complex relationships or interactions between features, especially when they are crucial for classification.

Assumption Fit: The choice of the Naive Bayes variant should align with the data distribution. Choosing the wrong variant may lead to poorer performance.