Comprehensive Question Bank: Naïve Bayes Classification

CSCI316 Big Data Mining Techniques and Implementation

Question 1: Theoretical Foundations of Bayesian Classification [8 marks]

Part (a) [3 marks]

Explain the key differences between prior and posterior probabilities in the context of Bayesian classification. Provide a real-world example that illustrates both concepts.

Part (b) [3 marks]

State Bayes' theorem in its general form for classification problems. Explain what each component represents and why we can ignore the denominator $P(X_1,...,X_m)$ in practical classification tasks.

Part (c) [2 marks]

What is the "naïve" assumption in Naïve Bayes classifiers? Discuss one advantage and one limitation of this assumption.

Question 2: Manual Calculation - Discrete Features [10 marks]

Consider the following customer dataset for predicting computer purchases:

Age	Income	Student	Credit Rating	Buys Computer
Youth	High	No	Fair	No
Youth	High	No	Excellent	No
Middle	High	No	Fair	Yes
Senior	Medium	No	Fair	Yes
Senior	Low	Yes	Fair	Yes
Senior	Low	Yes	Excellent	No
Middle	Low	Yes	Excellent	Yes

Part (a) [4 marks]

Calculate the prior probabilities P(Buys Computer = Yes) and P(Buys Computer = No).

Part (b) [4 marks]

Calculate all conditional probabilities for each attribute given each class:

- P(Age | Buys Computer)
- P(Income | Buys Computer)

- P(Student | Buys Computer)
- P(Credit Rating | Buys Computer)

Part (c) [2 marks]

Using your calculated probabilities, classify the following new customer: **Age = Youth, Income = Medium, Student = Yes, Credit Rating = Fair**

Show all calculation steps and determine the final classification.

Question 3: Continuous Features and Gaussian Distribution [9 marks]

Part (a) [4 marks]

Explain two approaches for handling continuous-valued features in Naïve Bayes classifiers. What are the advantages and disadvantages of each approach?

Part (b) [5 marks]

Given the following income data (in thousands) for customers who bought computers: **[35, 42, 48, 52, 58, 65, 72, 85]**

1. Calculate the mean (μ) and variance (σ^2) for this dataset [2 marks]
2. Using the Gaussian probability density function, calculate P(Income = 60 | Buys Computer = Yes) [2 marks]
3. Write the Python code using scipy.stats.norm to compute this probability [1 mark]

Question 4: Implementation Challenges [12 marks]

Part (a) [4 marks]

Numerical Underflow Problem:

1. Explain what causes numerical underflow in Naïve Bayes classifiers [2 marks]
2. Describe the logarithmic transformation solution and why it works [2 marks]

Part (b) [4 marks]

Zero Count Problem: Consider a training dataset where for the class "Spam Email = Yes", you have:

- 0 emails containing the word "meeting"
- 450 emails containing the word "offer"
- 50 emails containing the word "urgent"
- Total: 500 spam emails

1. What problem arises when classifying a new email containing "meeting"? [2 marks]
2. Apply Laplace smoothing and recalculate the probabilities [2 marks]

Part (c) [4 marks]

Write a Python function that implements Laplace smoothing for categorical features:

```python
def laplace_smoothing(feature_counts, total_count, num_categories, alpha=1):
    """
    Apply Laplace smoothing to feature probabilities

    Parameters:
    feature_counts: dict with feature values as keys and counts as values
    total_count: total number of samples in the class
    num_categories: total number of possible categories for this feature
    alpha: smoothing parameter (default=1)

    Returns:
    dict with smoothed probabilities
    """
    # Your implementation here
```

Question 5: Practical Implementation with Scikit-Learn [6 marks]

Part (a) [3 marks]

Write Python code using scikit-learn to:

1. Create and train a CategoricalNB classifier on the customer dataset from Question 2
2. Make predictions for the test instance from Question 2(c)
3. Display the prediction probabilities for both classes

Part (b) [3 marks]

Write Python code using scikit-learn to:

1. Create and train a GaussianNB classifier for continuous features
2. Demonstrate how to handle mixed categorical and continuous features
3. Evaluate the model using cross-validation

Question 6: Real-World Application - Email Spam Detection [11 marks]

You are tasked with building a spam email classifier using Naïve Bayes. The training dataset contains the following information:

Word Frequencies in Spam Emails (1000 emails):

- "free": appears in 600 emails
- "money": appears in 400 emails
- "meeting": appears in 50 emails

Word Frequencies in Ham Emails (2000 emails):

- "free": appears in 200 emails
- "money": appears in 100 emails
- "meeting": appears in 800 emails

Part (a) [3 marks]

Calculate the prior probabilities for Spam and Ham classes.

Part (b) [4 marks]

Calculate the likelihood probabilities for each word given each class. Apply Laplace smoothing with $\alpha = 1$, assuming a vocabulary size of 10,000 words.

Part (c) [2 marks]

Classify the following email: "Free money meeting today"
Show all calculation steps using both regular multiplication and log probabilities.

Part (d) [2 marks]

Discuss the conditional independence assumption in the context of this spam detection problem. Give one example where this assumption might be violated and explain its potential impact.

Question 7: Advanced Topics and Limitations [8 marks]

Part (a) [4 marks]

Performance Analysis:

1. Compare Naïve Bayes with decision trees in terms of:
    - Training time complexity
    - Handling of missing values
    - Feature independence assumptions
    - Interpretability

Part (b) [4 marks]

Handling Dependencies:

1. Explain why the conditional independence assumption can lead to loss of accuracy [2 marks]
2. Briefly describe how Bayesian Belief Networks address this limitation [2 marks]

Question 8: Comprehensive Implementation Task [15 marks]

Part (a) [8 marks]

Implement a complete Naïve Bayes classifier from scratch in Python:

```python
class NaiveBayesClassifier:
    def __init__(self, alpha=1):
        """
        Initialize the Naïve Bayes classifier
        alpha: Laplace smoothing parameter
        """
        pass

    def fit(self, X, y):
        """
        Train the classifier on the given data
        X: feature matrix (2D array)
        y: target labels (1D array)
        """
        pass

    def predict(self, X):
        """
        Make predictions on new data
        X: feature matrix (2D array)
        Returns: predicted class labels
        """
        pass

    def predict_proba(self, X):
        """
        Return prediction probabilities
        X: feature matrix (2D array)
        Returns: probability matrix
        """
        pass
```

Part (b) [4 marks]

Test your implementation on a synthetic dataset and compare results with scikit-learn's MultinomialNB.

Part (c) [3 marks]

Create visualizations showing:

1. Feature probability distributions for each class
2. Classification decision boundaries (for 2D case)

Question 9: Big Data Implementation with PySpark [10 marks]

Part (a) [6 marks]

Write PySpark code to implement Naïve Bayes classification on a large dataset:

1. Set up Spark context and load data from HDFS [2 marks]
2. Preprocess the data (handle missing values, encode categorical variables) [2 marks]
3. Train a Naïve Bayes model using MLlib [2 marks]

Part (b) [4 marks]

1. Evaluate the model using appropriate metrics (accuracy, precision, recall, F1-score) [2 marks]
2. Implement cross-validation for hyperparameter tuning [2 marks]

Question 10: Case Study - Student Performance Prediction [12 marks]

You are analyzing student performance data with the following features:

- Study Hours (continuous): hours spent studying per week
- Attendance (categorical): High, Medium, Low
- Previous Grade (categorical): A, B, C, D, F
- Extra Activities (categorical): Yes, No
- Target: Pass/Fail

Part (a) [4 marks]

Design a preprocessing pipeline that handles both categorical and continuous features for Naïve Bayes classification.

Part (b) [4 marks]

Implement the solution handling:

1. Gaussian distribution for continuous features
2. Multinomial distribution for categorical features
3. Cross-validation for model evaluation

Part (c) [4 marks]

Analyze the results:

1. Which features are most predictive of student success?
2. How does the conditional independence assumption affect this particular problem?
3. Suggest improvements to the model

Marking Rubric Guidelines

Theoretical Questions (40% of marks):

- **Excellent (90-100%)**: Complete understanding with clear explanations and examples
- **Good (70-89%)**: Good understanding with minor gaps in explanation
- **Satisfactory (50-69%)**: Basic understanding with some conceptual errors
- **Poor (<50%)**: Significant misunderstanding or incomplete answers

Manual Calculations (30% of marks):

- **Excellent**: All steps shown clearly, correct methodology and final answers
- **Good**: Minor calculation errors but correct approach
- **Satisfactory**: Correct approach with some calculation mistakes
- **Poor**: Incorrect methodology or major calculation errors

Implementation Tasks (30% of marks):

- **Excellent**: Working code with proper documentation and error handling
- **Good**: Working code with minor issues
- **Satisfactory**: Code works but lacks documentation or has inefficiencies
- **Poor**: Code doesn't work or shows fundamental misunderstanding

---

Additional Practice Questions

Quick Assessment Questions (5 marks each):

1. **Probability Basics**: Given P(A|B) = 0.8, P(B) = 0.3, and P(A) = 0.5, calculate P(B|A).
2. **Feature Independence**: Explain with an example when the naïve assumption might be reasonable and when it might be problematic.

3. **Smoothing Effects**: How does the choice of α in Laplace smoothing affect model performance on training vs. test data?
4. **Comparison Task**: Create a comparison table between Naïve Bayes, Decision Trees, and Logistic Regression across 5 different criteria.
5. **Error Analysis**: Given a confusion matrix for a Naïve Bayes classifier, calculate precision, recall, and F1-score, and interpret the results.

This question bank covers all major aspects of Naïve Bayes classification from theoretical foundations to practical implementation, following the assessment format requirements with a mix of theory, manual calculations, and coding tasks.