Classifier Evaluation - Comprehensive Question Bank

CSCI316 Big Data Mining Techniques and Implementation

Question 1: Confusion Matrix and Basic Metrics (8 marks)

A machine learning model was trained to predict whether customers will purchase a premium subscription service. The following confusion matrix shows the results on a test dataset of 5000 customers:

Actual\Predicted	Subscribe	No Subscribe	Total
Subscribe	850	150	1000
No Subscribe	200	3800	4000
Total	1050	3950	5000

Part A (3 marks): Calculate the following metrics showing all steps:

- Accuracy
- Error Rate
- Sensitivity (Recall)

Part B (3 marks): Calculate Precision and Specificity, explaining what each metric represents in the context of subscription prediction.

Part C (2 marks): Identify whether this dataset suffers from class imbalance problem and justify your answer with calculations.

Question 2: Advanced Evaluation Metrics and F-measures (10 marks)

A medical diagnostic system is designed to detect a rare disease that affects 2% of the population. The system was tested on 10,000 patients with the following results:

Actual\Predicted	Disease	Healthy	Total
Disease	180	20	200
Healthy	300	9500	9800
Total	480	9520	10000

Part A (4 marks): Calculate Precision, Recall, and F1-score showing all calculation steps.

Part B (3 marks): Calculate F2-score (β=2) and F0.5-score (β=0.5). Explain which F-measure would be more appropriate for this medical diagnostic system and why.

Part C (3 marks): Discuss the implications of false positives and false negatives in this medical context. Which error type is more critical and how does this relate to the choice of evaluation metric?

Question 3: Cross-Validation Implementation (12 marks)

Part A (6 marks): Implement a Python function `stratified_k_fold_split(X, y, k=5)` that performs stratified k-fold cross-validation splitting. The function should:

- Take input features X, labels y, and number of folds k
- Return k tuples of (train_indices, test_indices)
- Ensure each fold maintains approximately the same class distribution as the original dataset
- Handle edge cases where class counts are not perfectly divisible by k

```python
def stratified_k_fold_split(X, y, k=5):
    """
    Your implementation here
    """
    pass
```

Part B (3 marks): Explain the advantages of stratified cross-validation over regular k-fold cross-validation, particularly in cases of class imbalance.

Part C (3 marks): Compare holdout method, k-fold cross-validation, and leave-one-out cross-validation in terms of:

- Computational complexity
- Bias-variance trade-off
- Appropriate use cases

Question 4: Statistical Significance Testing (9 marks)

You have compared two classification models (M1 and M2) using 10-fold cross-validation and obtained the following error rates for each fold:

Fold	M1 Error	M2 Error
1	0.12	0.15
2	0.14	0.13
3	0.11	0.16
4	0.13	0.14
5	0.15	0.12
6	0.10	0.17
7	0.12	0.14
8	0.14	0.15
9	0.13	0.13
10	0.11	0.16

Part A (4 marks): Calculate the mean error rates for both models and compute the t-statistic for comparing M1 and M2. Show all calculation steps including the standard deviation calculation.

Part B (3 marks): Using a significance level of 5% ($\alpha = 0.05$), determine whether there is a statistically significant difference between the two models. Use the t-distribution table value for 9 degrees of freedom at $\alpha/2 = 0.025$ is 2.262.

Part C (2 marks): Interpret your results and recommend which model to choose, if any. Explain the practical implications of your statistical test.

Question 5: ROC Curves and AUC Analysis (11 marks)

A binary classifier produces the following predictions with associated probability scores on a test set:

Instance	Actual Class	Predicted Probability	Rank
1	Positive	0.95	1
2	Negative	0.88	2
3	Positive	0.76	3
4	Positive	0.72	4
5	Negative	0.65	5
6	Negative	0.43	6
7	Positive	0.38	7
8	Negative	0.22	8

Part A (5 marks): Manually construct the ROC curve by calculating TPR and FPR at different threshold points. Show the calculation steps for at least 4 different thresholds.

Part B (3 marks): Calculate the Area Under the Curve (AUC) using the trapezoidal rule. Show your calculations step by step.

Part C (3 marks): Interpret the AUC value and explain what it means for model performance. Compare this model's performance to a random classifier and a perfect classifier.

Question 6: Comprehensive Model Evaluation with PySpark (10 marks)

You are working with a large dataset of flight delay predictions using PySpark MLlib. The dataset contains features like weather conditions, airline, day of week, etc., to predict whether a flight will be delayed (binary classification).

Part A (4 marks): Write PySpark code to load a dataset, split it into training and testing sets (70-30 split), and train a Random Forest classifier. Include proper data preprocessing steps.

```python
from pyspark.sql import SparkSession
from pyspark.ml.classification import RandomForestClassifier
from pyspark.ml.evaluation import BinaryClassificationEvaluator
# Your code here
```

Part B (3 marks): Implement code to calculate and display multiple evaluation metrics including accuracy, precision, recall, F1-score, and AUC using PySpark's evaluation tools.

Part C (3 marks): Discuss how you would handle the following challenges in this big data context:

- Class imbalance (assume only 20% of flights are delayed)
- Feature selection with high-dimensional data
- Cross-validation with large datasets

Question 7: Model Selection Criteria (6 marks)

Part A (3 marks): A company is choosing between three different classification models for their fraud detection system. List and explain five key criteria they should consider beyond just accuracy when selecting the final model.

Part B (3 marks): For each criterion you mentioned, explain how it specifically applies to fraud detection and why it matters in this business context. Consider factors like cost of false positives vs. false negatives, regulatory requirements, and operational constraints.

Question 8: Real-world Application Analysis (8 marks)

You are consulting for an e-commerce company that wants to implement a recommendation system classifier to predict whether a customer will like a product (binary classification). They have provided you with the following business requirements:

- The system must process 1 million predictions per hour
- False positives (recommending disliked products) cost $2 in lost customer satisfaction
- False negatives (missing good recommendations) cost $5 in lost sales
- The current manual system has 75% accuracy
- Model interpretability is important for business stakeholders

Part A (4 marks): Design an evaluation strategy that addresses these business requirements. Specify which metrics you would prioritize and what evaluation methodology you would use.

Part B (4 marks): Explain how you would set the classification threshold to optimize business value rather than accuracy. Include a cost-benefit analysis framework and discuss how you would validate your threshold choice.

Question 9: Implementation Challenge - Custom Metrics (7 marks)

Part A (4 marks): Implement a Python function that calculates a custom business metric called "Profit Score" defined as:

```
Profit Score = (TP × Revenue_per_TP - FP × Cost_per_FP - FN × Cost_per_FN) / Total_Instances
```

```python
def calculate_profit_score(y_true, y_pred, revenue_per_tp=100, cost_per_fp=20, cost_per_fn=50):
    """
    Calculate custom profit score for business-oriented evaluation

    Parameters:
    y_true: actual labels
    y_pred: predicted labels
    revenue_per_tp: revenue generated per true positive
    cost_per_fp: cost incurred per false positive
    cost_per_fn: cost incurred per false negative

    Returns:
    profit_score: calculated profit score
    """
    # Your implementation here
    pass
```

Part B (3 marks): Demonstrate your function with a sample dataset and explain scenarios where this metric would be more valuable than traditional accuracy metrics.

Question 10: Advanced Cross-Validation Techniques (9 marks)

Part A (5 marks): Implement a time-series cross-validation function for temporal data where future data cannot be used to predict past events:

```python
def time_series_cv_split(X, y, n_splits=5, test_size=0.2):
    """
    Implement time-series cross-validation with expanding window

    Parameters:
    X: features (time-ordered)
    y: labels (time-ordered)
    n_splits: number of splits
    test_size: proportion of data for testing in each split

    Returns:
    Generator yielding (train_indices, test_indices) for each split
    """
    # Your implementation here
    pass
```

Part B (2 marks): Explain why standard k-fold cross-validation is inappropriate for time-series data and how your implementation addresses this issue.

Part C (2 marks): Compare expanding window vs. sliding window approaches for time-series cross-validation, discussing the trade-offs between them.

Answer Guidelines and Marking Rubric

Theoretical Questions (40% of marks):

- **Full marks**: Complete explanation with correct terminology and context
- **Partial marks**: Correct concept but incomplete explanation or minor errors
- **No marks**: Incorrect understanding or missing key concepts

Calculation Questions (35% of marks):

- **Full marks**: Correct final answer with all intermediate steps shown
- **Partial marks**: Correct method but arithmetic errors or missing steps
- **No marks**: Incorrect method or no working shown

Implementation Questions (25% of marks):

- **Full marks**: Working code with proper error handling and documentation
- **Partial marks**: Code with minor bugs or missing edge cases
- **No marks**: Non-functional code or incorrect algorithm

Expected Study Time: 15-20 hours

Recommended Practice: Work through 2-3 questions daily, focusing on manual calculations before moving to implementation tasks.