KNN Model Lab Exercise: Wine Quality Prediction

Problem Statement

You are a machine learning consultant hired by **VinTech Analytics**, a wine distribution company that wants to implement an automated wine quality assessment system. The company receives wines from various vineyards and needs to predict quality ratings based on physicochemical properties to make informed purchasing and pricing decisions.

Your task is to develop a robust K-Nearest Neighbors (KNN) model that can accurately predict wine quality while demonstrating deep understanding of the algorithm's strengths, limitations, and optimization strategies.

Dataset Information

Source: UCI Wine Quality Dataset

URL: https://archive.ics.uci.edu/ml/datasets/Wine+Quality

- **Red Wine Dataset**: 1,599 samples
- **White Wine Dataset**: 4,898 samples
- **Features**: 11 physicochemical properties + wine type
- **Target**: Quality score (0-10, but typically 3-8)

Features Description:

1. **fixed acidity** - tartaric acid concentration
2. **volatile acidity** - acetic acid concentration
3. **citric acid** - adds freshness and flavor
4. **residual sugar** - remaining sugar after fermentation
5. **chlorides** - salt content
6. **free sulfur dioxide** - prevents microbial growth
7. **total sulfur dioxide** - bound and free SO2
8. **density** - water/alcohol/sugar ratio
9. **pH** - acidity/alkalinity scale
10. **sulphates** - wine additive (potassium sulphate)
11. **alcohol** - percentage by volume

Lab Requirements & Challenges

Phase 1: Data Preprocessing & Engineering

1.1 Data Integration Challenge

- Combine red and white wine datasets
- Create a new categorical feature: wine_type (red=0, white=1)
- **Challenge**: Handle potential data inconsistencies between datasets

1.2 Missing Values Simulation

Since the original dataset has no missing values, introduce realistic missing data:

- Randomly remove 5% of values from citric_acid (MCAR - Missing Completely at Random)
- Remove 3% of sulphates values where quality < 5 (MAR - Missing at Random)

- Implement and compare 3 different imputation strategies

1.3 Feature Engineering

Create at least 3 new engineered features, such as:

- acid_ratio = fixed_acidity / volatile_acidity
- sulfur_ratio = free_sulfur_dioxide / total_sulfur_dioxide
- alcohol_sugar_interaction = alcohol * residual_sugar
- **Justify** your feature choices based on domain knowledge

1.4 Data Scaling

- Compare the impact of different scaling methods: StandardScaler, MinMaxScaler, RobustScaler
- **Challenge**: Analyze which scaler works best for KNN and explain why

Phase 2: Exploratory Data Analysis

2.1 Target Variable Analysis

- Analyze quality distribution - is it balanced?
- Create visualizations showing quality distribution by wine type
- **Critical Question**: How does class imbalance affect KNN performance?

2.2 Feature Relationships

- Create correlation matrix and identify multicollinearity issues
- Perform feature importance analysis using mutual information
- **Challenge**: Which features are most predictive for each wine type separately?

2.3 Distance Analysis

- Visualize feature distributions and identify potential outliers
- **Challenge**: How do outliers affect KNN? Demonstrate with examples.

Phase 3: Model Development & Evaluation

3.1 Train-Test Split Strategy

- Implement stratified sampling to maintain quality distribution
- Use 70-15-15 split (train-validation-test)
- **Justification**: Explain why stratified sampling is crucial for this problem

3.2 Baseline KNN Implementation

- Implement KNN with k=5, Euclidean distance
- Evaluate using: Accuracy, Precision, Recall, F1-score, MAE, RMSE
- **Critical Analysis**: Why is accuracy potentially misleading here?

3.3 Advanced Evaluation

- Implement custom evaluation metric: **Quality-Weighted Accuracy**
    - Penalize predictions that are off by more than 1 quality point
- Create confusion matrix with quality score interpretation
- **Challenge**: Design a business-relevant cost function

Phase 4: Hyperparameter Optimization

4.1 K-Value Optimization

- Test k values from 1 to 50
- Use cross-validation to find optimal k
- **Challenge**: Implement and compare different CV strategies (k-fold, stratified k-fold, time series split if treating as temporal data)

4.2 Distance Metric Comparison

Compare performance using:

- Euclidean distance
- Manhattan distance
- Minkowski distance (p=3)
- **Advanced**: Implement weighted distance based on feature importance

4.3 Advanced KNN Variants

Implement and compare:

- Weighted KNN (distance-based weights)
- **Challenge**: Radius-based neighbors (RadiusNeighborsClassifier)
- **Bonus**: Local Outlier Factor integration

Phase 5: Critical Analysis & Alternatives

5.1 KNN Limitations Analysis

Address these specific limitations:

- **Curse of dimensionality**: Demonstrate with synthetic high-dimensional data
- **Computational complexity**: Measure prediction time vs. training set size
- **Memory requirements**: Calculate and discuss storage needs
- **Imbalanced data handling**: How does KNN handle minority classes?

5.2 Alternative Model Comparison

Implement one alternative model (Random Forest or SVM) and compare:

- Performance metrics
- Training/prediction time
- Interpretability
- **Recommendation**: Which model would you recommend for production and why?

5.3 Feature Selection Impact

- Apply feature selection techniques (SelectKBest, RFE)
- Compare KNN performance with reduced feature sets
- **Analysis**: When does dimensionality reduction help vs. hurt KNN?

Advanced Challenges (Bonus Points)

Challenge A: Custom Distance Metric

Design a domain-specific distance metric that considers:

- Wine chemistry relationships (e.g., pH and acidity correlation)

- Business importance of certain features

Challenge B: Ensemble KNN

Create an ensemble of KNN models with different:

- K values
- Distance metrics
- Feature subsets
- Combine predictions using voting or averaging

Challenge C: Online Learning Simulation

Simulate a scenario where new wine samples arrive continuously:

- Implement incremental updates to the model
- Handle concept drift (wine quality standards changing over time)

Deliverables

1. Technical Implementation

- **Jupyter Notebook** with well-documented code
- All phases implemented with clear section headers
- Code should be reproducible and include random seeds

2. Technical Report (3-5 pages)

Include:

- **Executive Summary** for business stakeholders
- **Methodology** section explaining your approach
- **Results & Analysis** with visualizations
- **Model Comparison** table with all metrics
- **Recommendations** for production deployment
- **Limitations & Future Work**