Milestone 2: Feature Analysis Implementation

Due: Should be finished by 2/18/2025

Overview

In this milestone, you will implement the analysis of variable importance and scaling effects, building on the theoretical foundation from Milestone 1. You'll create code to analyze how different features contribute to the overall variance in your data, with special attention to the linear algebraic structure of your transformations.

Objectives

1. Implement variable importance analysis through subspace decomposition
2. Create scale normalization procedures that preserve key matrix properties
3. Develop initial SVD code structure with focus on numerical stability
4. Analyze the effects of different normalization approaches on eigenstructure

Key Concepts to Demonstrate

1. Understanding of how scaling affects fundamental subspaces
2. Connection between variable importance and row/column spaces
3. Role of condition number in numerical stability
4. Relationship between scaling and eigenvalue distribution

Required Deliverables

1. Code Implementation (50%)

- Variable importance analysis implementation:
  - Compute variance contributions through projection matrices
  - Analyze feature correlations via inner products

  - Implement rank-revealing decomposition
  - Connect to fundamental subspaces:
    * Row space: directions of variation
    * Null space: invariant feature combinations
    * Column space: achievable transformations
    * Left null space: zero-variance directions
- Scale normalization implementation:
  - Matrix-based standardization operations
  - Analysis of condition number effects
  - Preservation of subspace structure
  - Study of how scaling affects:
    * Eigenvalue distribution
    * Principal directions
    * Rank properties
    * Numerical stability
- Initial SVD code structure:
  - Direct implementation using numpy
  - Numerical stability considerations
  - Rank determination methods
  - Connection to fundamental theorem:
    * U basis for column space
    * V basis for row space
    * Σ reveals rank structure
    * Null spaces from small singular values
- Test cases and validation:
  - Synthetic data with known properties
  - Edge cases for rank deficiency
  - Stability under scaling
  - Verification of subspace properties

2. Analysis (30%)

- Variance-based feature relevance analysis:
  - Analyze how variance distributes across fundamental subspaces
  - Show relationship between feature correlations and row space

  - Demonstrate how null space reveals redundant features
  - Connect left null space to estimation uncertainty
- Scale normalization effects:
  - Document how scaling transforms fundamental subspaces
  - Analyze condition number before and after normalization
  - Show preservation/changes in orthogonality relationships
  - Quantify impact on eigenvalue distribution
- Cross-correlation impact:
  - Study how correlations affect rank of covariance matrix
  - Analyze relationship between correlation and collinearity
  - Show impact on numerical stability of SVD
  - Connect to practical feature selection decisions

3. Documentation (20%)

- Code documentation
- Analysis results
- Implementation decisions

Technical Requirements

- Use Python with numpy for implementations
- Include test cases
- Provide example usage
- Document all functions

Code Structure

```
# Required imports
import numpy as np
from typing import Tuple, List

def compute_variable_importance(X: np.ndarray) -> np.ndarray:
    """
    Compute importance scores for each variable
```

```
    Args:
        X: Data matrix (n_samples, n_features)
    Returns:
        Importance scores for each feature
    """
    pass

def normalize_features(X: np.ndarray) -> Tuple[np.ndarray, np.ndarray]:
    """
    Scale and center features

    Args:
        X: Data matrix (n_samples, n_features)
    Returns:
        Normalized data, scaling factors
    """
    pass

def initial_svd_analysis(X: np.ndarray) -> Tuple[np.ndarray, np.ndarray, np.ndarray]:
    """
    Perform initial SVD analysis

    Args:
        X: Data matrix (n_samples, n_features)
    Returns:
        U, S, Vh matrices
    """
    pass
```

Evaluation Criteria

- Code correctness and efficiency
- Implementation completeness
- Quality of analysis
- Documentation clarity

Resources

- Lessons 11-12 content
- NumPy documentation
- Course notes on feature scaling

Submission Guidelines

1. Submit Python files with implementations
2. Include test cases
3. Provide analysis document
4. Add usage examples

Tips for Success

- Test with simple cases first
- Validate results
- Document assumptions
- Consider edge cases