# **Name: Kashish Khobragade Mail:** [**khobragadekashish@gmail.com**](mailto:khobragadekashish@gmail.com)

# **Hyperspectral Data ML Analysis**

# **Final Report**

## **🔹 Preprocessing Steps and Rationale**

### **1. Data Exploration and Cleaning**

✅ **Objective:** Ensure data quality, identify missing values, and visualize spectral trends.

* **Checked for missing values** to prevent issues in modeling.
* **Statistical summary** provided insights into feature distributions.
* **Plotted spectral reflectance** across wavelengths to observe trends.

### **2. Feature Scaling & Dimensionality Reduction**

✅ **Objective:** Reduce high-dimensional feature space while preserving key spectral information.

#### **2.1 Principal Component Analysis (PCA)**

* **Why?** Hyperspectral data has redundant features; PCA reduces complexity.
* **Outcome:** Retained 95% variance with fewer components, reducing computational cost.

#### **2.2 t-SNE for Visualization**

* **Why?** Captures nonlinear structures that PCA may miss.
* **Outcome:** Revealed local clustering, hinting at nonlinear relationships.

#### **2.3 PCA-based Visualization**

* **2D & 3D PCA plots** confirmed feature separability, guiding model selection.

💡 **Key Takeaways:**

* PCA preserves most spectral information while reducing computation.
* t-SNE confirmed the need for nonlinear models like XGBoost.

## **🔹 Insights from Dimensionality Reduction**

### **1. PCA**

✅ **Findings:**

* The first few components captured most variance, confirming feature redundancy.
* The 3D PCA plot suggested separability, aiding regression tasks.

### **2. t-SNE**

✅ **Findings:**

* Showed **clustering patterns**, proving hyperspectral data has structure.
* Suggested **tree-based models or deep learning** for best performance.

💡 **Model Selection Takeaways:**

* PCA ensured faster model training with minimal information loss.
* t-SNE confirmed that **linear models would underperform**.
* **XGBoost or CNNs were prioritized** to capture spectral relationships.

## **🔹 Model Selection & Performance**

### **✅ Models Tried & Results**

| Model | MAE ↓ | RMSE ↓ | R² ↑ | Notes |
| --- | --- | --- | --- | --- |
| **SVM** | 4177.33 | 17047.24 | -0.04 | Struggled with high-dimensional data. |
| **Random Forest** | 2987.55 | 8637.43 | 0.73 | Good but lacked fine-tuning. |
| **1D CNN** | 3859.34 | 15308.43 | 0.16 | Did not generalize well due to limited data. |
| **Tuned Random Forest** | 3153.07 | 9693.31 | 0.66 | Overfitted slightly after tuning. |
| **XGBoost (Final Model)** | **2558.92** | **8382.82** | **0.75** | Best balance of accuracy &  generalization. |
| **XGBoost with Optuna (Best)** | **1974.12** | **6791.12** | **0.84** | Bayesian optimization enhanced tuning, reducing error. |

### **🔹 Why XGBoost?**

✅ Captured **spectral relationships** better than other models.  
✅ Robust against **high-dimensional input data**.  
✅ **Best trade-off** between complexity and performance.

## **🔹 Key Findings & Areas for Improvement**

### **✅ Experimental Insights**

* **PCA + SPA** significantly reduced features while retaining predictive power.
* **SVM & CNN struggled** due to hyperspectral data’s complexity.
* **Tree-based models (RF & XGBoost) outperformed other approaches.**

### **💡 Suggestions for Improvement**

1. **Feature Engineering:** Explore domain-specific techniques like **continuum removal**.
2. **More Data & Augmentation:** Increase hyperspectral sample size or **use synthetic data**.
3. **Hybrid Models:** Combine **CNN for feature extraction** + **XGBoost for prediction**.
4. **Better Hyperparameter Optimization:** Use **Bayesian Optimization** instead of GridSearch.

## **🔹 Challenges & Trade-offs**

⚠️ **Handling High-Dimensional Data**

* **Trade-off:** PCA reduced complexity, but excessive reduction could lose information.
* **Solution:** Retained **95% variance in PCA + top 20 features via SPA**.

⚠️ **Choosing Between Deep Learning & Tree-Based Models**

* **Trade-off:** CNNs require large datasets, while XGBoost performs well with structured data.
* **Solution:** **XGBoost was selected**, but a CNN-XGBoost hybrid could be explored.

⚠️ **Generalization & Overfitting**

* **Trade-off:** Random Forest overfitted slightly when tuned.
* **Solution:** **XGBoost’s regularization and subsampling** improved generalization.

## **🔹 Conclusion: Why XGBoost?**

✅ **Best trade-off between accuracy, generalization, and interpretability.**  
✅ **Outperformed all other models on RMSE, MAE, and R².**  
✅ **Highly effective for hyperspectral regression problems.**