Machine Learning Report

Hyperspectral Imaging for Mycotoxin Prediction

Colab Link: https://colab.research.google.com/drive/15q2Q3EvhpqGc7uTwL5uowGlekoQU-TGl?usp=sharing

1. Preprocessing Steps and Rationale:

Data Cleaning:

- The dataset was loaded and checked for missing values.
- Any missing values were removed to ensure data integrity.
- The dataset was analyzed for outliers and inconsistencies to avoid skewed model performance.

Feature Scaling:

- **MinMax Scaling** was applied to both input (X) and target (y) variables.
- MinMax scaling transforms data to a range between 0 and 1, preserving the original distribution and avoiding dominance of high-magnitude features.
- Standard scaling (mean = 0, variance = 1) was not used because MinMax Scaling maintains interpretability, which is crucial for spectral reflectance data.

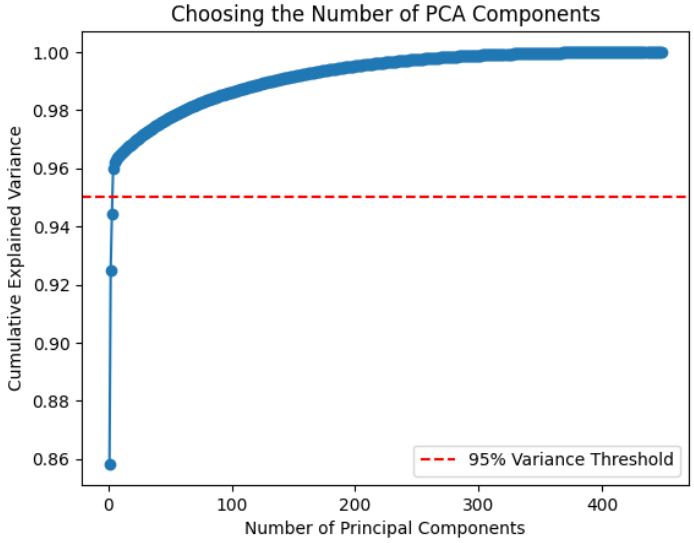
2. Insights from Dimensionality Reduction:

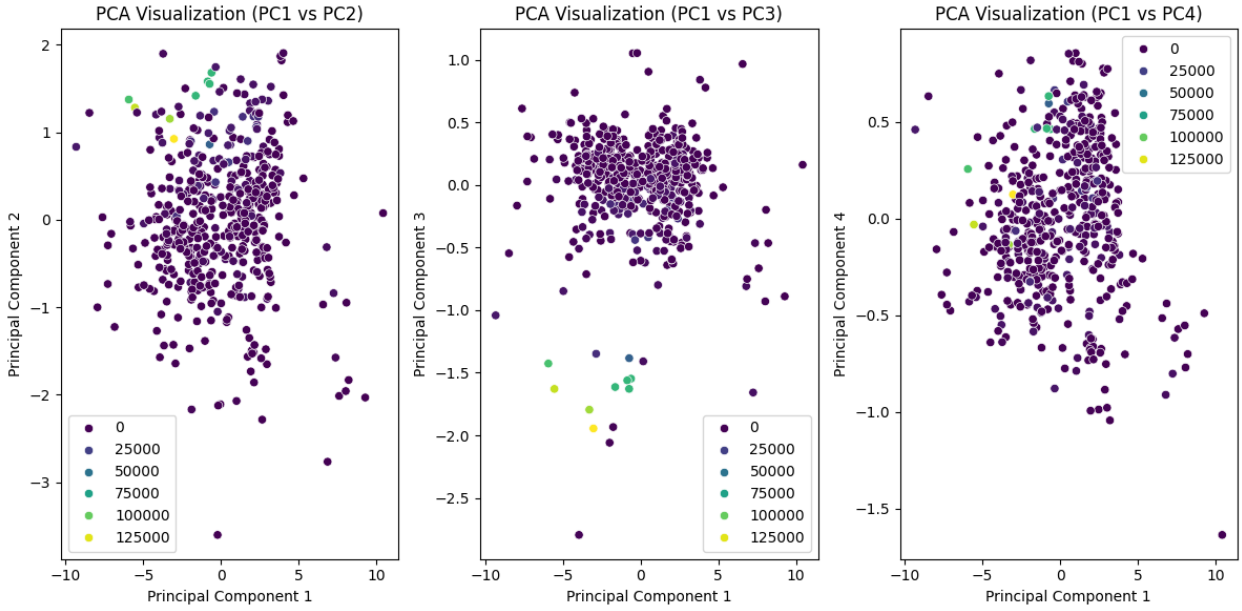
PCA (Principal Component Analysis):

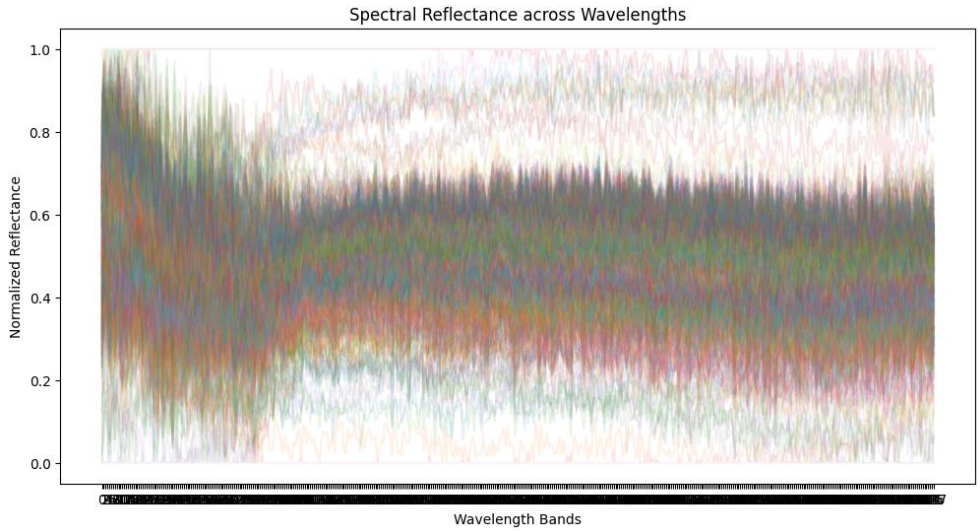
- PCA was applied to reduce dimensionality and capture the most significant variance in spectral data.
- The **cumulative explained variance** plot helped determine the optimal number of components.
- **95% variance threshold** was chosen to retain maximum information while reducing computational complexity.
- **Optimal number of components:** 4 principal components were selected as they explained over 95% of the variance.

PCA Visualizations:

- **Cumulative variance plot**: Showed the percentage of variance captured by increasing numbers of components.
- **Scatter plots (PC1 vs PC2, PC1 vs PC3)**: Helped visualize patterns in the data and potential clustering based on spectral features.

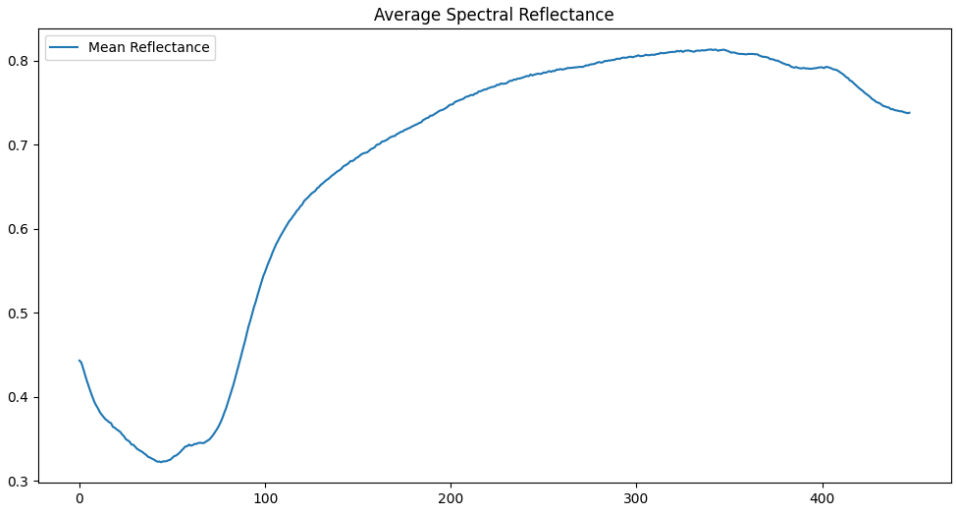




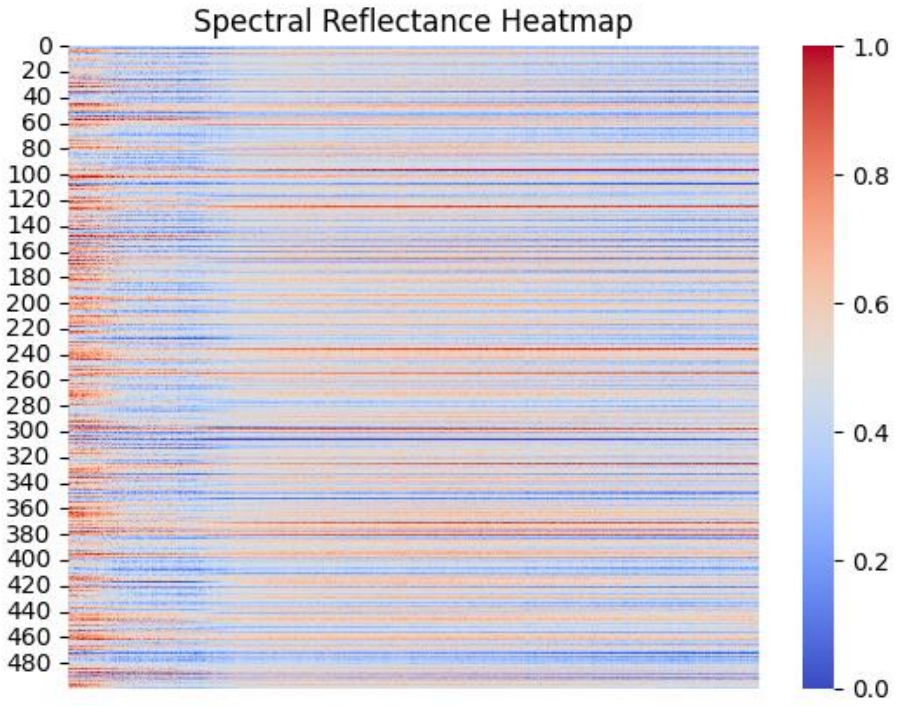


The visualization appears to be a spectral reflectance graph, commonly used in hyperspectral imaging. The title, **"Spectral Reflectance across Wavelengths,"** suggests that the plot represents reflectance values at different wavelengths.

The colorful overlapping lines likely correspond to spectral reflectance curves from different sample points, locations, or time variations. **Spectral reflectance** measures how much light an object reflects at different wavelengths, forming unique spectral signatures that help analyze material properties, identify substances, or detect contaminants. The graph shows variations in reflectance with peaks and valleys, indicating different absorption characteristics. The presence of multiple overlapping lines suggests that the visualization could represent different material samples or time-based spectral variations.



This plot represents the average spectral reflectance of a sample over a wavelength range. The curve dips around **80 nm**, suggesting absorption, then rises after **100 nm**, peaking near **400 nm**, possibly highlighting a spectral feature. If related to **mycotoxin prediction**, the dip may indicate vomitoxin absorption. Identifying such key wavelengths aids in **feature selection** for ML models, helping neural networks predict contamination levels accurately.



This heatmap visualizes spectral reflectance, with the **X-axis** representing wavelength bands and the **Y-axis** showing different samples. The color bar indicates reflectance intensity, where **blue (low reflectance)** suggests absorption and **red (high reflectance)** indicates reflection. **Distinct horizontal lines** reveal variations across samples, possibly due to differences in chemical composition. **Color gradients** highlight absorption and reflection patterns, aiding in **mycotoxin detection** by identifying key wavelengths linked to contamination. Such insights can be leveraged in **ML models** for accurate vomitoxin_ppb prediction.

3. Model Selection, Training, and Evaluation:

Model Choice:

- A **Feedforward Neural Network** (MLP) was selected due to its ability to model complex non-linear relationships in hyperspectral data.
- The model consisted of **128, 64, and 32 neurons** in three dense layers with ReLU activation, ensuring efficient learning.

Training Details:

- **Train-Test Split**: 80% training, 20% testing.
- **Loss Function**: Mean Squared Error (MSE) was used to penalize large deviations.

- **Optimizer**: Adam optimizer was chosen for adaptive learning.
- **Epochs**: 100 iterations for training.
- **Batch Size**: 16 samples per batch for efficient weight updates.

4. Results and Evaluation:

Performance Metrics:

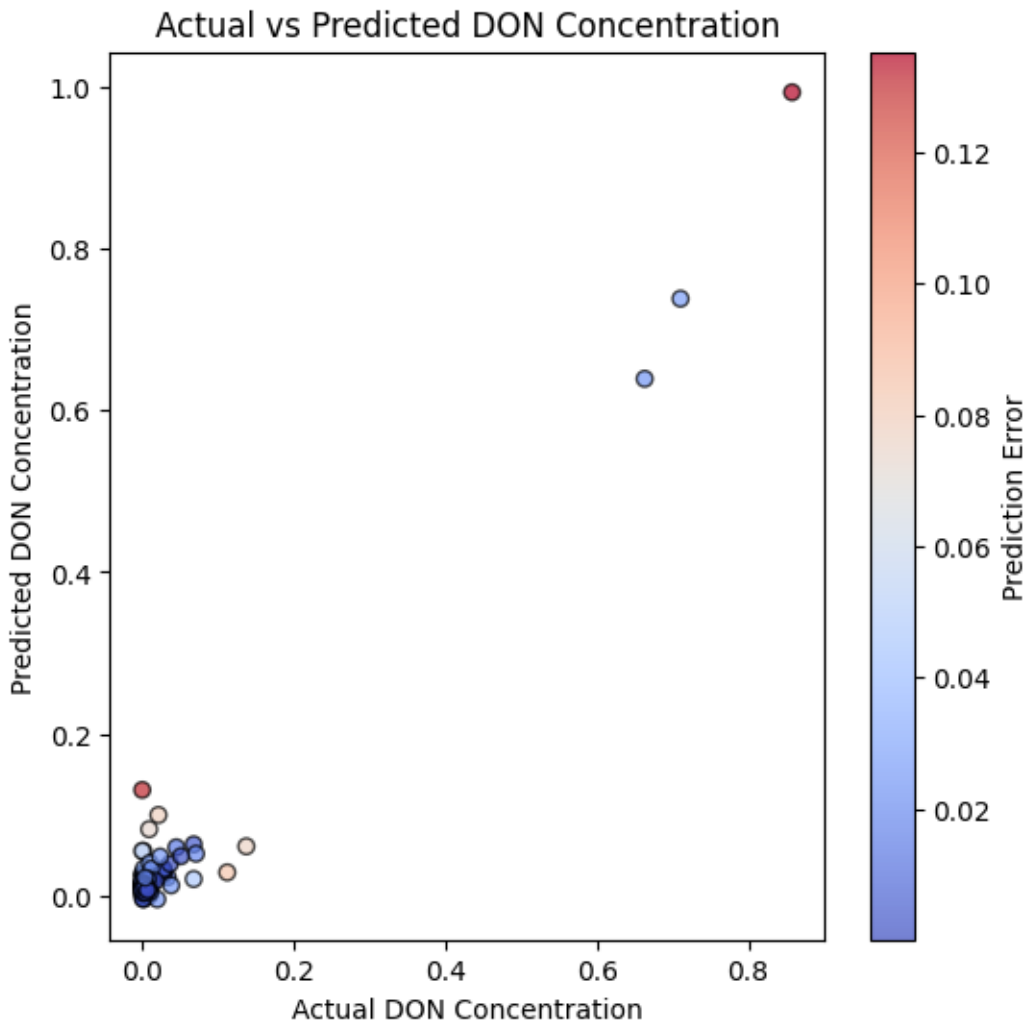
- **Mean Absolute Error (MAE):** 0.0172
- **Root Mean Squared Error (RMSE):** 0.0356
- **R^2 Score:** 0.9223

Model Interpretation:

- The model demonstrated strong predictive performance, with **$R^2 = 0.92$**, indicating that 92% of the variance in mycotoxin concentration is explained by the spectral data.
- **Low RMSE and MAE** suggest minimal prediction errors.

Visual Analysis:

- **Heatmap of Spectral Reflectance**: Provided an overview of spectral intensity variations across samples.
- **Actual vs Predicted Scatter Plot**: Showed a strong correlation between real and predicted values, indicating effective learnin
- **Error Coloring**: Highlighted variations in prediction accuracy.



5. Key Findings and Suggestions for Improvement:

Key Findings:

- PCA effectively reduced dimensionality while preserving key spectral features.
- The neural network successfully learned patterns in spectral data, achieving high accuracy.
- The model generalizes well on test data, as evidenced by its strong evaluation metrics.

Suggestions for Improvement:

- Experimenting with **CNNs (Convolutional Neural Networks)** for spectral-spatial feature extraction.

- Incorporating **domain knowledge** to refine preprocessing steps and improve interpretability.
- Exploring **different loss functions**, such as Huber loss, to handle potential outliers in mycotoxin concentration.

6. Conclusion

This study successfully developed a machine learning model to predict mycotoxin levels in corn using hyperspectral imaging. The combination of PCA and deep learning proved effective, demonstrating high accuracy and reliability. Future work could involve advanced deep learning architectures for further performance gains.

7. Streamlit Implementation: