AICS Assignment #3: Basic Setup & Data Import with Anomaly Detection

AICS Instructor: Steve Smith
Due Date: August 5, 11:59 PM ET
Grade: 15 points

🎯 Objective

Apply the anomaly detection techniques learned in Classes 4 and 5 to your preprocessed malware dataset from Assignment #2. This assignment bridges data preprocessing with unsupervised machine learning, preparing you for supervised classification in later assignments.

📋 Requirements

1️⃣ Environment Setup & Data Import (3 points)

- Load your preprocessed dataset from Assignment #2
- Import required libraries (scikit-learn, pandas, numpy, matplotlib, seaborn)
- Display dataset summary statistics and confirm preprocessing was successful
- Verify data types and shape are appropriate for ML algorithms

2️⃣ LLM-Assisted Code Generation (3 points)

- **Demonstrate LLM prompting skills** by including at least 3 different prompts you used to generate Python code
- Document the prompts in markdown cells before the generated code
- Show how you refined prompts to get better results
- Examples of good prompts:
    - "Generate Python code to implement Isolation Forest for malware detection with contamination parameter tuning"
    - "Create a function to visualize anomaly detection results using matplotlib with proper labels and legends"
    - "Write code to compare three anomaly detection algorithms and create a summary table of results"

3️⃣ Anomaly Detection Implementation (6 points)

Implement all three algorithms covered in Class 5:

K-Means Clustering (2 points)

- Implement K-means with appropriate number of clusters (test k=2,3,4,5)
- Calculate distances from cluster centers to identify outliers
- Use elbow method or silhouette analysis to choose optimal k

Isolation Forest (2 points)

- Implement with contamination parameter testing (0.1, 0.2, 0.3)
- Generate anomaly scores for all samples
- Identify samples flagged as anomalies (-1)

One-Class SVM (2 points)

- Implement with nu parameter testing (0.1, 0.2, 0.3)
- Generate decision function scores
- Compare boundary definition approaches

4️⃣ Feature Analysis for Cybersecurity Context (2 points)

- Select and analyze the most relevant features for malware detection
- Create correlation heatmap of top 10 features
- Analyze feature importance in the context of PE file characteristics:
    - Entropy measures
    - Import function counts
    - Section characteristics
    - File size metrics

5️⃣ Results Comparison & Visualization (1 point)

- Create comparison table showing:
    - Algorithm name
    - Number of anomalies detected
    - Percentage of dataset flagged as anomalous
    - Parameter settings used
- Generate visualizations using PCA for 2D representation of anomalies
- Include confusion matrix comparison (if you have ground truth labels)

📊 Deliverables

- **Jupyter notebook** with all code, outputs, and analysis
- **LLM prompts documentation** in markdown cells
- **Visualization outputs** embedded in notebook
- **Summary analysis** of which algorithm performed best and why

💡 LLM Prompting Guidelines

✅ Effective Prompt Examples:

"Create Python code to implement Isolation Forest anomaly detection for a malware dataset with 69 features. Include parameter tuning for contamination values [0.1, 0.2, 0.3] and visualization of results using matplotlib."

"Generate a function that compares K-means, Isolation Forest, and One-Class SVM anomaly detection algorithms. Return a pandas DataFrame with algorithm names, number of anomalies detected, and contamination rates."

"Write code to create a 2D visualization of anomaly detection results using PCA dimensionality reduction. Color-code normal vs anomalous samples and add a legend."

🚫 Avoid These Prompt Mistakes:

- ❌ "Write some anomaly detection code"
- ❌ "Make a plot"
- ❌ "Fix this error" (without providing context)

🔬 Technical Requirements

Data Preparation:

- Use StandardScaler on numerical features before anomaly detection
- Handle any remaining missing values appropriately
- Ensure data is in proper format for scikit-learn algorithms

Algorithm Implementation:

```
# Example structure - get specific code from LLM prompts
from sklearn.cluster import KMeans
from sklearn.ensemble import IsolationForest
from sklearn.svm import OneClassSVM
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

Expected Output Format:

- Algorithm comparison table
- PCA visualization plots
- Parameter tuning results
- At least 3 documented LLM prompts with resulting code

🎯 Learning Outcomes

By completing this assignment, you will:

- ✅ Apply unsupervised learning algorithms to cybersecurity data
- ✅ Understand the differences between clustering and anomaly detection approaches
- ✅ Practice effective LLM prompting for code generation
- ✅ Interpret anomaly detection results in cybersecurity context
- ✅ Prepare preprocessed data for supervised learning in future assignments

💯 Grading Rubric

Component	Excellent (90-100%)	Good (80-89%)	Satisfactory (70-79%)	Needs Improvement (<70%)
Environment & Import (3pts)	Perfect setup, clear data verification, comprehensive imports	Minor issues with setup or verification	Basic setup with some missing elements	Significant setup problems
LLM Prompting (3pts)	3+ well-documented prompts with iterative refinement shown	3 prompts documented but limited refinement	2-3 prompts with basic documentation	Inadequate prompt documentation
Algorithm Implementation (6pts)	All 3 algorithms correctly implemented with parameter tuning	2-3 algorithms with minor issues	2 algorithms working properly	Only 1 algorithm or major implementation errors
Feature Analysis (2pts)	Comprehensive cybersecurity-focused feature analysis	Good analysis with minor gaps	Basic feature analysis	Limited or incorrect analysis
Results & Visualization (1pt)	Clear, professional visualizations and comprehensive comparison	Good visualizations with minor issues	Basic but adequate visualizations	Poor or missing visualizations

🚨 Common Pitfalls to Avoid

- ❌ Not scaling features before applying anomaly detection algorithms
- ❌ Using inappropriate contamination parameters without justification
- ❌ Failing to interpret results in cybersecurity context
- ❌ Poor documentation of LLM prompting process
- ❌ Not connecting results back to malware detection goals

🔗 Helpful Resources

- Scikit-learn Anomaly Detection Documentation
- Course slides from Classes 4 & 5
- LLM prompting best practices from course materials
- Your Assignment #2 preprocessed dataset

Note: This assignment builds directly toward your capstone malware detection project. The anomaly detection insights you gain here will inform your supervised learning approach in later assignments.