AICS Student Lab Notebook: Unsupervised Anomaly Detection for Cybersecurity

Class 10 - AI in Cybersecurity

Table of Contents

Learning Objectives

By the end of this lab, you will:

- ✅ Understand when to use each unsupervised anomaly detection algorithm
- ✅ Implement Isolation Forest, K-Means, and One-Class SVM
- ✅ Compare algorithm performance and interpret results
- ✅ Apply ensemble methods for improved detection
- ✅ Visualize anomaly detection results effectively
- ✅ Understand cybersecurity applications and best practices

Lab Structure

Section 1: Setup and Data Loading

- Import required libraries
- Load and explore cybersecurity dataset
- Perform data preprocessing and scaling
- Create initial data visualizations

Key Code Blocks:

- Library imports and configuration
- Synthetic data generation (if no real dataset available)
- Data cleaning and scaling functions
- Exploratory data analysis plots

Section 2: Isolation Forest

- Understand the isolation principle
- Implement Isolation Forest algorithm
- Experiment with contamination parameters
- Visualize results using PCA
- Complete reflection questions

Student Exercises:

- Parameter experimentation with different contamination values
- Score distribution analysis
- PCA visualization interpretation
- Algorithm strengths/weaknesses discussion

Section 3: K-Means Clustering

- Learn cluster-based anomaly detection
- Implement K-Means with distance-based outlier detection
- Use elbow method to find optimal clusters
- Analyze cluster assignments and distances
- Complete reflection questions

Student Exercises:

- Optimal cluster number determination
- Distance distribution analysis
- Cluster visualization in PCA space
- Comparison of different k values

Section 4: One-Class SVM

- Understand boundary-based anomaly detection
- Implement One-Class SVM with different kernels
- Experiment with nu parameter
- Analyze decision boundaries and scores
- Complete reflection questions

Student Exercises:

- Kernel comparison (linear, RBF, polynomial, sigmoid)
- Parameter sensitivity analysis
- Decision score interpretation
- Boundary visualization

Section 5: Algorithm Comparison

- Compare all three algorithms
- Implement ensemble detection methods
- Calculate performance metrics (if labels available)
- Create comprehensive comparison visualizations
- Analyze method agreements and disagreements

Key Analysis:

- Jaccard similarity between methods
- Ensemble confidence scoring
- High-confidence anomaly identification
- Method-specific strengths assessment

Section 6: Assignment Tasks

- Review assignment requirements (10 points total)
- Understand grading rubric
- Access starter code template
- Plan implementation approach

Assignment Breakdown:

- Task 1: Data Import & Cleaning (2 points)
- Task 2: Algorithm Implementation (3 points)
- Task 3: Performance Evaluation (2 points)
- Task 4: Visualization & Analysis (2 points)
- Task 5: Written Analysis (1 point)

Section 7: Summary and Next Steps

- Review key concepts learned
- Understand cybersecurity applications
- Learn best practices for production deployment
- Explore advanced topics and resources

Required Software and Libraries

```
# Core libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Scikit-learn components
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import IsolationForest
from sklearn.cluster import KMeans
from sklearn.svm import OneClassSVM
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score
```

Dataset Requirements

- Network traffic data with multiple features
- Minimum 1000 samples recommended
- Features should include flow characteristics (duration, packet counts, byte rates, etc.)
- Optional: Ground truth labels for evaluation
- Example datasets: CICIDS2017, NSL-KDD, or synthetic data provided

Expected Outputs

Students will produce:

1. **Jupyter Notebook** with all code, visualizations, and analysis
2. **Visualizations** including PCA plots, performance comparisons, and ensemble analysis
3. **Written Analysis** discussing algorithm comparisons and cybersecurity applications
4. **Parameter Analysis** showing optimal settings for each algorithm
5. **Ensemble Results** demonstrating consensus-based detection

Assessment Criteria

- **Code Quality**: Clean, commented, error-free implementation
- **Analysis Depth**: Thorough comparison and interpretation of results
- **Visualizations**: Clear, informative, properly labeled plots
- **Written Communication**: Clear explanations and insights
- **Technical Understanding**: Correct implementation and parameter tuning

Common Challenges and Solutions

Challenge	Solution
High false positive rate	Adjust contamination parameters, use ensemble methods
Poor algorithm performance	Check data scaling, try different parameters
Visualization issues	Ensure proper data types, check for infinite/NaN values
Slow performance	Use data sampling, optimize parameters
No ground truth labels	Focus on ensemble analysis and domain expertise

Time Management Tips

- **Start Early**: Begin with data loading and exploration
- **Iterate Quickly**: Get basic implementations working first
- **Document Progress**: Add markdown explanations as you go
- **Test Frequently**: Run code cells regularly to catch errors
- **Ask Questions**: Use office hours and discussion forums

Success Checklist

- [ ] All three algorithms implemented and working
- [ ] Parameter tuning completed for each method
- [ ] Comprehensive visualizations created
- [ ] Algorithm comparison analysis completed
- [ ] Written analysis addresses all required points
- [ ] Code is well-commented and organized
- [ ] Results are properly interpreted
- [ ] Assignment submitted on time

🎯 **Remember**: The goal is not just to implement the algorithms, but to understand when and why to use each one in cybersecurity applications. Focus on building intuition about how these methods work and their practical implications.

🚀 **Good luck with your anomaly detection journey!**