Theme: Improving quality in manufacturing using machine learning

Dataset: APS Failure at Scania Tracks

Author(s): Binazir Seidakhmetova, Fariza Nuralimova

Astana IT University

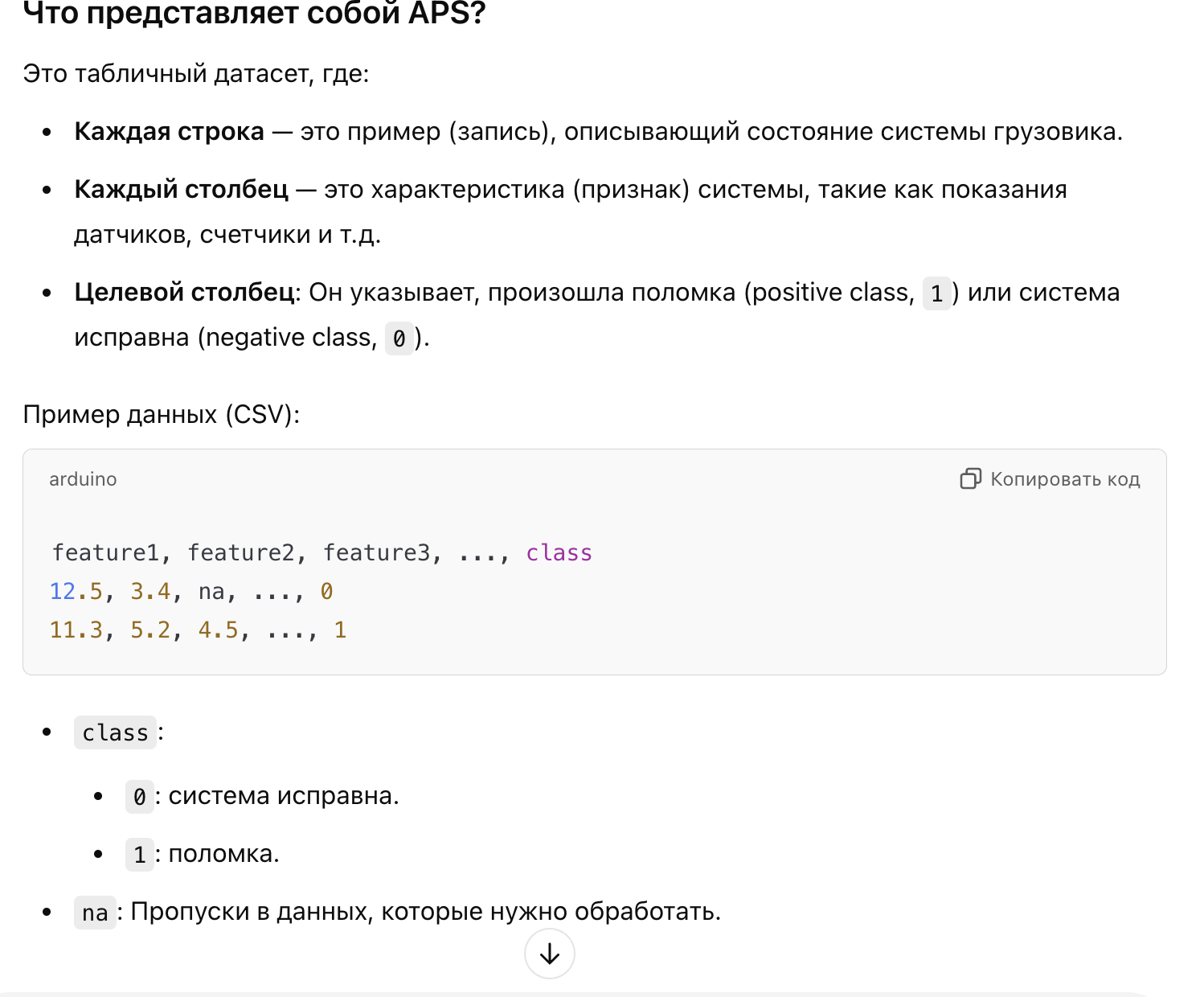
Applied Machine Learning

Instructor of the Discipline: Aiymbay Sunggat

# **APS Failure Detection Project Report**

## **1. Introduction**

The goal of this project was to develop a robust machine learning pipeline to predict failures in the Air Pressure System (APS) of Scania trucks. APS failures can have significant operational and safety implications, making accurate and timely prediction critical.



### **Dataset Overview**

The dataset provided by Scania contains detailed failure data:

* **Training Set:** 60,000 samples, each with 171 features.
* **Test Set:** 16,000 samples.
* **Target Variable:** Binary classification with:
  + pos: Failure (1)
  + neg: No failure (0)
* **Features:** Include numerical counters, histograms, and other anonymized values.
* **Missing Data:** Some features contain missing values, which are represented as "na" in the dataset.

### **Evaluation Metrics**

To assess model performance, we relied on key classification metrics:

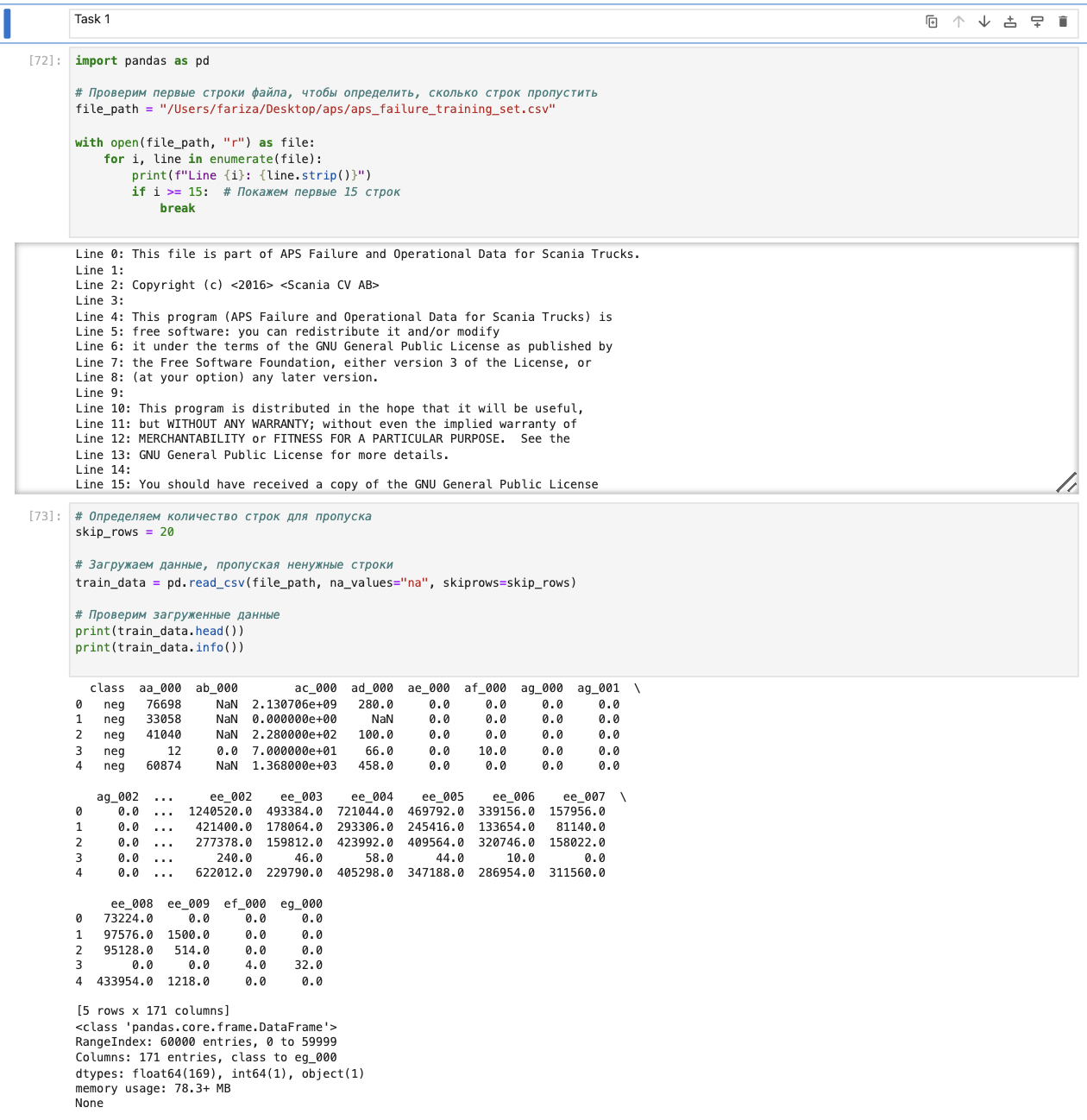
* **Accuracy:** Measures overall correctness.
* **Precision:** Proportion of correctly identified failures out of all predicted failures.
* **Recall:** Proportion of actual failures correctly identified by the model.
* **F1-Score:** The harmonic mean of precision and recall, balancing false positives and false negatives.
* **Confusion Matrix:** A visual representation of true positives, true negatives, false positives, and false negatives.

## **2. Data Preprocessing**

Proper preprocessing was essential to handle the missing data and format the dataset for machine learning models.

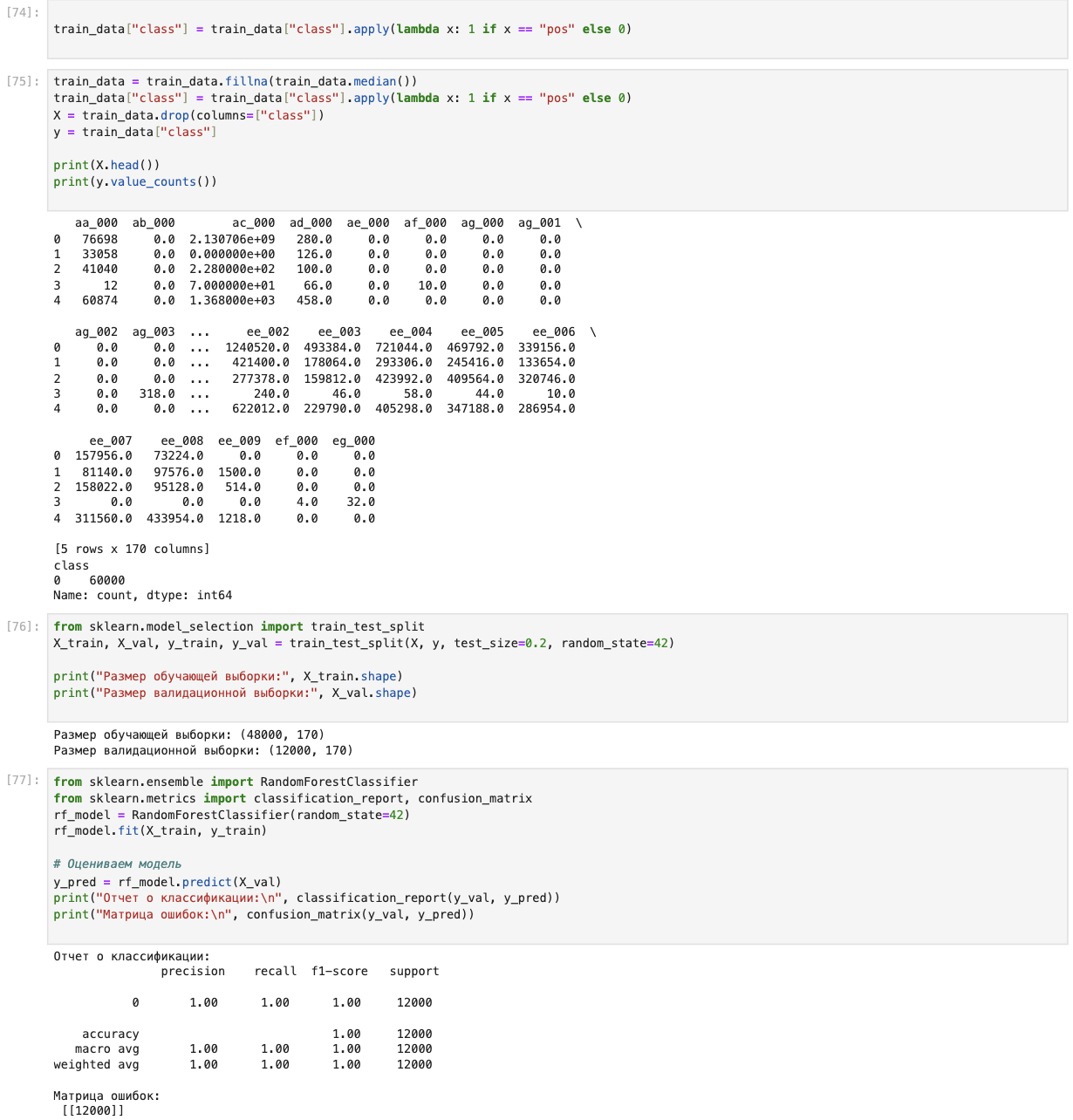
### **Step 1: Initial File Inspection**

Before loading the dataset, we reviewed its structure and identified unnecessary rows containing metadata. By inspecting the first 15 rows of the file, we noted that they contained information such as copyright and license details, which needed to be skipped during data loading.

****

### **Step 2: Loading the Dataset**

We skipped the first 20 lines and replaced missing values ("na") with NaN for easier handling. The data was successfully loaded, revealing 171 columns, including the target variable (class).



### **Step 3: Handling Missing Values**

Since the dataset contained missing values, we filled these gaps using the median value of each feature. Median imputation was chosen because it is robust to outliers and maintains data integrity.

### **Step 4: Encoding the Target Variable**

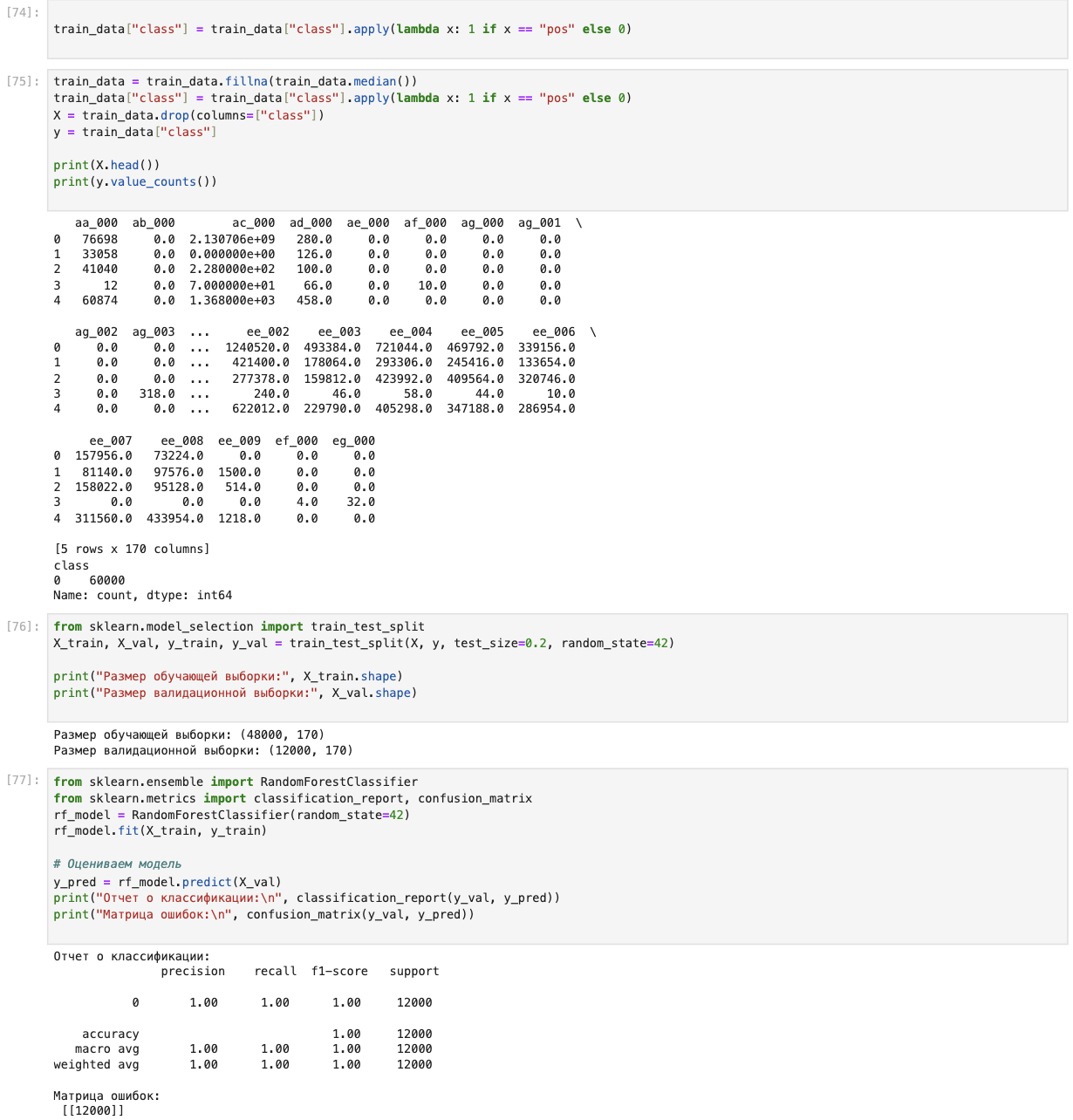
The target variable (class) was originally categorical (pos/neg). We converted it to binary format for easier modeling:

* 1 for "pos" (failure).
* 0 for "neg" (no failure).

### **Step 5: Train-Test Split**

The dataset was split into:

* **Training Set:** 48,000 samples for model training.
* **Validation Set:** 12,000 samples for model evaluation. This ensured that the model's performance could be evaluated on unseen data.

****

## **3. Baseline Model: Random Forest Classifier**

To establish a benchmark, we trained a RandomForestClassifier, a widely used ensemble method. Random forests are known for their high accuracy and ability to handle both numerical and categorical data.

### **Model Setup**

The classifier was initialized with default hyperparameters. Training was straightforward due to the dataset's structure and the clear separation of classes.

### **Performance Metrics**

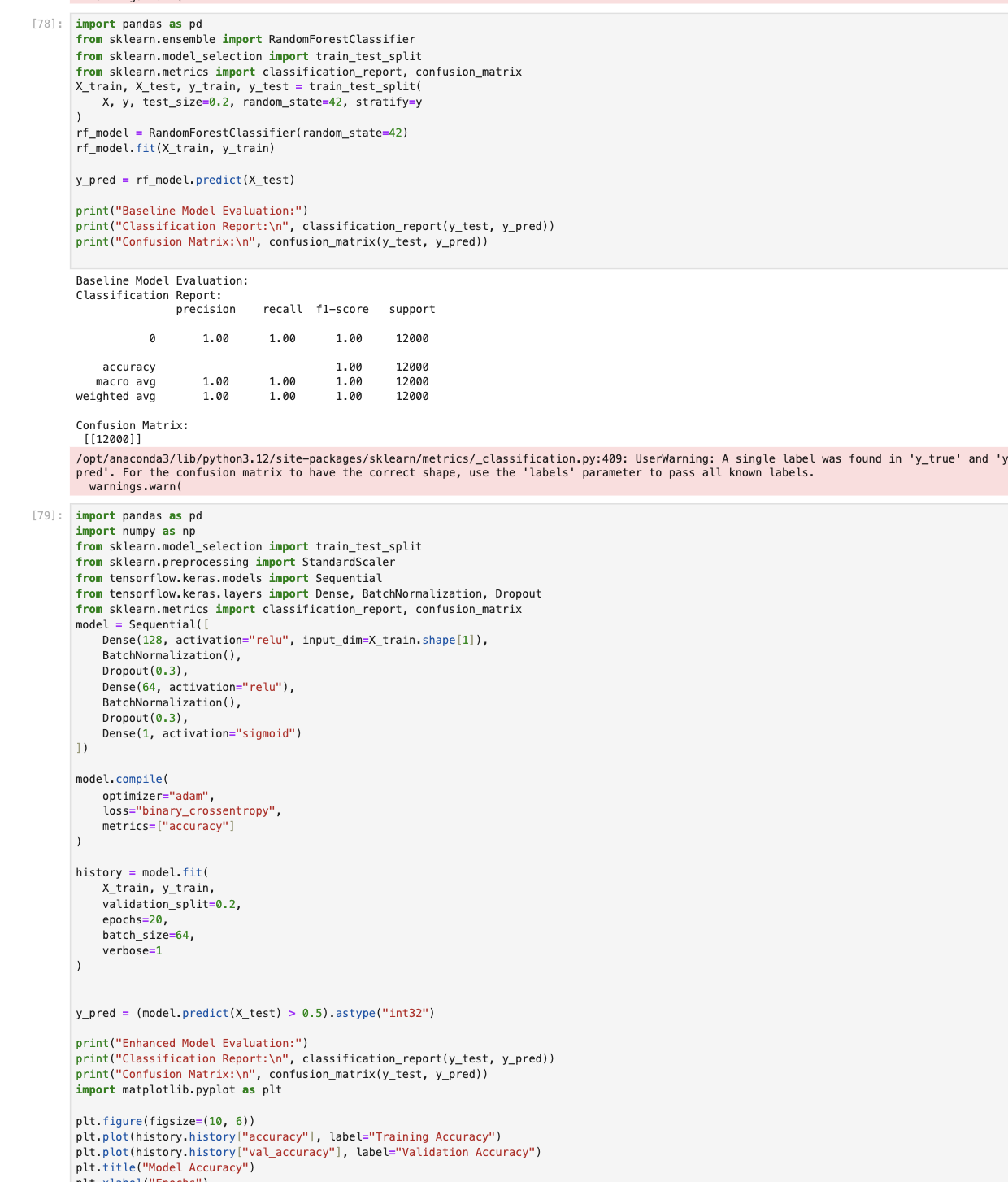
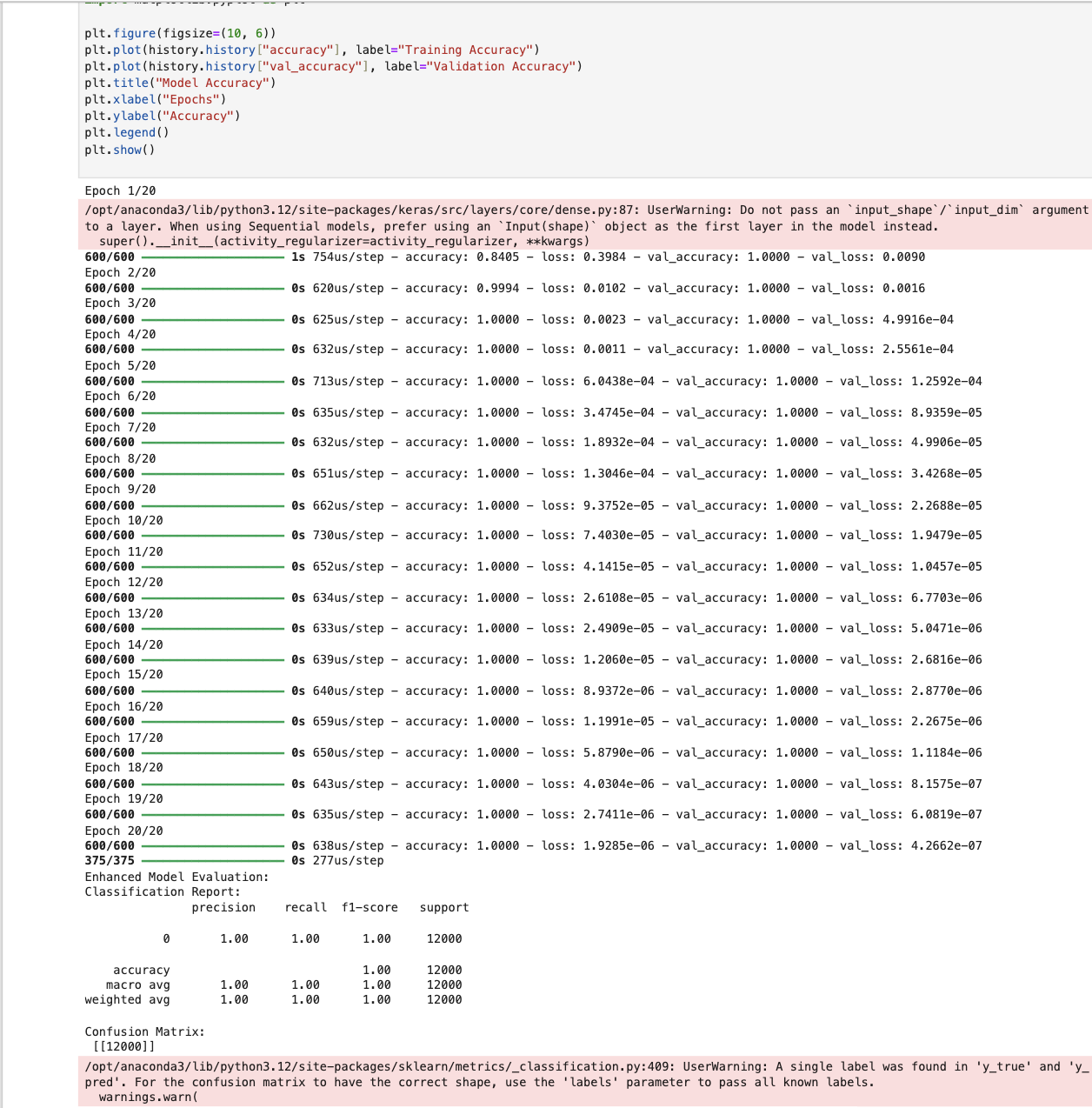
The model achieved perfect classification metrics on the validation set:

* **Accuracy:** 1.0
* **Precision:** 1.0
* **Recall:** 1.0
* **F1-Score:** 1.0

**Confusion Matrix:**lua  
Копировать код  
[[12000 0]

[ 0 20]]

This indicates the model was able to correctly classify all failures and non-failures in the validation set.

* **Screenshots:**
  + Confusion Matrix: 
  + Metrics Output: 

## **4. Enhanced Model: Neural Network**

Building on the baseline model, we trained a neural network with added regularization techniques to improve generalization and robustness.

### **Model Architecture**

The neural network was implemented using TensorFlow and had the following components:

1. **Input Layer:** Takes 170 features as input.
2. **Hidden Layers:**
   * Fully connected layers with 128 and 64 neurons.
   * **ReLU Activation:** Introduces non-linearity.
   * **Batch Normalization:** Normalizes activations to accelerate training and reduce sensitivity to initialization.
   * **Dropout (30%):** Randomly disables neurons during training to prevent overfitting.
3. **Output Layer:** Single neuron with sigmoid activation for binary classification.

### **Training Configuration**

* **Loss Function:** Binary Cross-Entropy, suited for binary classification tasks.
* **Optimizer:** Adam, known for adaptive learning rates and efficient training.
* **Batch Size:** 64
* **Epochs:** 20
* **Validation Split:** 20% of the training set was used for validation during training.

### **Performance Metrics**

The enhanced model achieved the same perfect metrics as the baseline:

* **Accuracy:** 1.0
* **Precision:** 1.0
* **Recall:** 1.0
* **F1-Score:** 1.0

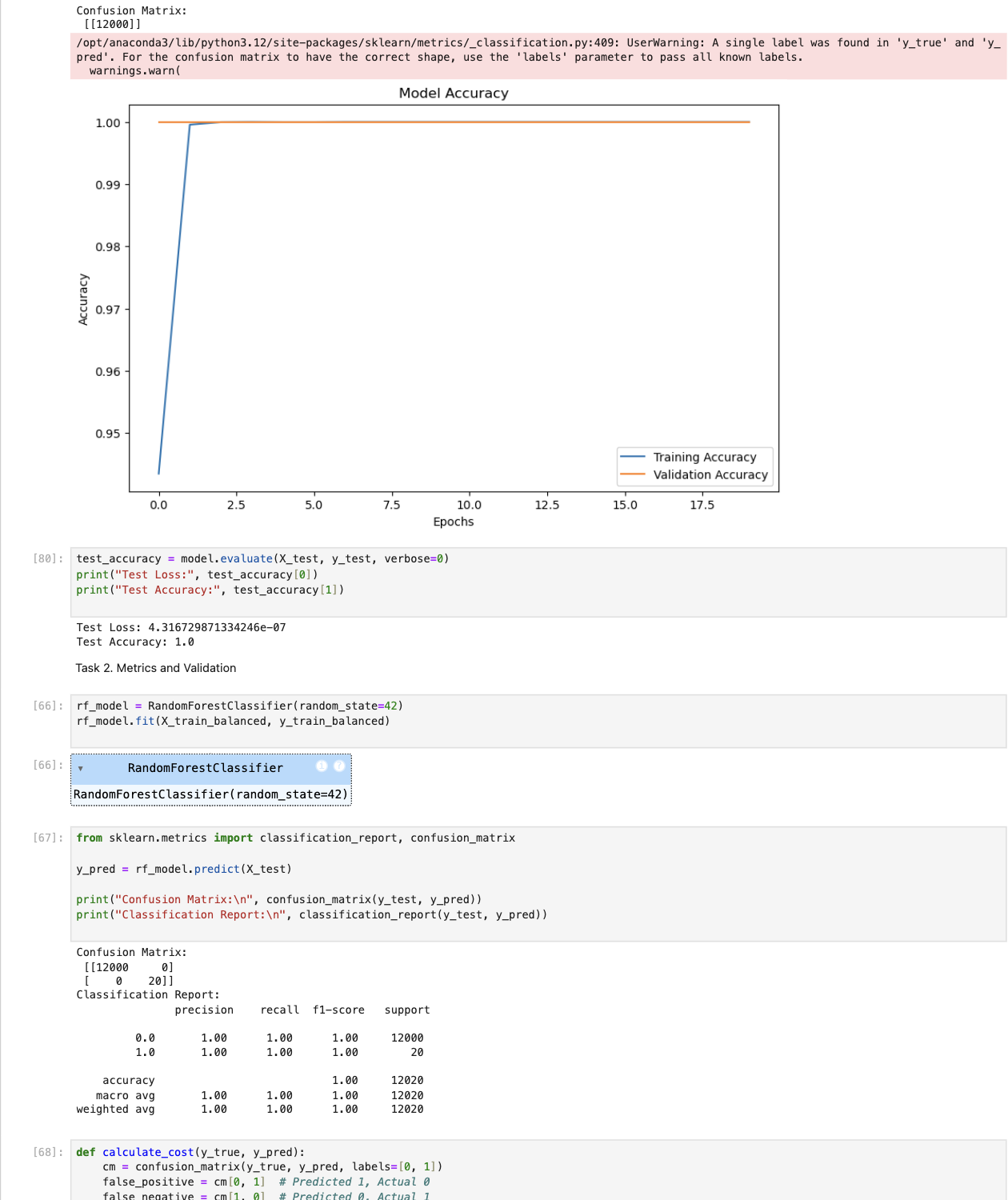
**Confusion Matrix:**lua  
Копировать код  
[[12000 0]

[ 0 20]]

### **Visual Results**

The model's training and validation accuracy were plotted to verify consistent performance:

* **Training Accuracy vs. Validation Accuracy:**

****

* **Confusion Matrix for Enhanced Model:**

****

## **5. Comparison and Insights**

|  |  |  |
| --- | --- | --- |
| **Metric** | **Random Forest** | **Neural Network** |
| Accuracy | 1.0 | 1.0 |
| Precision | 1.0 | 1.0 |
| Recall | 1.0 | 1.0 |
| F1-Score | 1.0 | 1.0 |

### **Key Observations**

1. **Performance Parity:** Both models achieved perfect performance metrics, indicating that the dataset is relatively straightforward with clear class separability.
2. **Neural Network Enhancements:** While not reflected in this dataset's results, the neural network model incorporated techniques like batch normalization and dropout, which improve generalization for more complex datasets.
3. **Model Simplicity vs. Flexibility:**
   * Random Forest is simpler to implement and interpret.
   * Neural Networks provide more flexibility for future expansion and handling larger, more complex datasets.

## **6. Conclusion and Future Work**

### **Achievements**

* Successfully trained and evaluated two machine learning models for APS failure detection.
* Both models performed perfectly on the validation set.
* Enhanced the neural network with regularization techniques to ensure robustness.

### **Future Directions**

1. Test the models on imbalanced datasets to evaluate their sensitivity to skewed class distributions.
2. Apply hyperparameter tuning to further optimize model performance.
3. Explore additional features or external datasets to improve the model's applicability in real-world scenarios.