**Code Structure**

The implementation includes four scripts, aligned with the research paper:

1. **featur\_selection2.py**: Executes SBOA for feature selection on Spark.
2. **classification.py**: Trains the OGRU model using TensorFlow.
3. **s3\_producer.py**: Streams data to Kafka.
4. **s3\_consumer.py**: Consumes Kafka messages for real-time predictions.

**Modules Used**

* **featur\_selection2.py**:
  + pyspark.sql, pyspark.ml: Spark data processing and Logistic Regression.
  + numpy: Numerical computations.
  + pickle: Saves feature indices.
* **classification.py**:
  + tensorflow.keras: OGRU model with GRU, Dense, Dropout, and Adam optimizer.
  + pyarrow.parquet: Reads Parquet files.
  + sklearn.metrics: AUC computation.
  + numpy: Data manipulation.
* **s3\_producer.py**:
  + kafka.KafkaProducer: Sends data to Kafka.
  + pyarrow.parquet: Reads Parquet files.
  + numpy, json: Data serialization.
* **s3\_consumer.py**:
  + kafka.KafkaConsumer: Consumes Kafka messages.
  + tensorflow.keras: Loads OGRU model.
  + sklearn.preprocessing: StandardScaler for normalization.
  + numpy: Data handling.

**Execution Steps**

1. **Environment Setup**:
   * Configure a multi-node Hadoop cluster with HDFS (hdfs://master:9000).
   * Install Kafka with brokers at localhost:9092.
   * Install dependencies: Python 3.8+, TensorFlow, PySpark, PyArrow, Kafka-Python, NumPy, Scikit-learn.
2. **Feature Selection**:
   * Run featur\_selection2.py to apply SBOA.
   * Input: epsilon\_normalized.t in HDFS (hdfs://master:9000/user/ubuntu/mydir/).
   * Output: epsilon\_selected.parquet, selected\_features.pkl.
   * Uses 0.5% sampling to reduce runtime from 4-5 hours (per your concern).
3. **Model Training**:
   * Run classification.py to train OGRU.
   * Input: epsilon\_selected.parquet.
   * Output: optimized\_ogru\_model.h5.
   * Configured for 50 epochs to approach 96.87% AUC (addressing 77% accuracy issue).
4. **Data Streaming**:
   * Run s3\_producer.py to stream data to Kafka topic y\_topic.
   * Input: epsilon\_selected.parquet.
5. **Real-Time Prediction**:
   * Run s3\_consumer.py to predict labels from Kafka messages.
   * Input: optimized\_ogru\_model.h5, y\_topic.
   * Output: Predicted labels and confidence scores.

**Dependencies and Environment Setup**

* **Hardware**: Multi-node Hadoop cluster (1 master, multiple workers).
* **Software**:
  + Hadoop 3.x with HDFS.
  + Apache Spark 3.x with PySpark.
  + Kafka 2.x (localhost:9092).
  + Python 3.8+.
* **Python Libraries**:

pip install tensorflow==2.10.0 pyspark==3.3.0 pyarrow==8.0.0 kafka-python==2.0.2 numpy==1.22.0 scikit-learn==1.0.2

* **Configuration**:
  + Spark: spark.driver.maxResultSize=4g, spark.executor.memory=8g, spark.executor.cores=6.
  + HDFS: Use hdfs://master:9000/user/ubuntu/mydir/.
  + Kafka: Ensure y\_topic exists.
  + Replaced ubuntu1 with ubuntu in HDFS paths.