```
import json
import base64
import requests
import boto3
import io
import csv
import logging
import os

# Initialize Kinesis client
kinesis_client = boto3.client(’kinesis’)

# Replace with your Kinesis Data Stream name
KINESIS_STREAM_NAME = os.environ.get(’KINESIS_STREAM_NAME’,
’SimpleLearnProject2-Adult-Data-Awskinesis’)
BATCH_SIZE = int(os.environ.get(’BATCH_SIZE’, 500))

# Set up logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    dataset_url = ’https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data’

    try:
        response = requests.get(dataset_url, timeout=10)
        response.raise_for_status()
    except requests.RequestException as e:
        logger.error(f"Request failed: {e}")
        return {
            ’statusCode’: 500,
            ’body’: json.dumps(’Failed to fetch dataset’)
        }

    data = response.text
    data_io = io.StringIO(data)
    reader = csv.reader(data_io, delimiter=’,’)

    records = []
    total_records = 0
    valid_records = 0

    for row in reader:
        total_records += 1
        record_json = json.dumps(row)

        # Validate JSON
        try:
            json.loads(record_json)  # Check if JSON is valid
            encoded_record = base64.b64encode(record_json.encode(’utf-8’)).decode(’utf-8’)
            records.append({
                ’Data’: encoded_record,
                ’PartitionKey’: ’Adult’
            })
            valid_records += 1
        except json.JSONDecodeError as e:
            logger.warning(f"Invalid JSON record skipped: {record_json} - Error: {e}")

        if len(records) >= BATCH_SIZE:
            send_to_kinesis(records)
            records = []

    if records:
        send_to_kinesis(records)

    logger.info(f"Processed {total_records} records, with {valid_records} valid records sent to Kinesis")
    return {
        ’statusCode’: 200,
        ’body’: json.dumps(f"Processed {total_records} records, with {valid_records} valid records sent to
Kinesis")
    }

def send_to_kinesis(records):
    try:
        kinesis_client.put_records(
            StreamName=KINESIS_STREAM_NAME,
            Records=records
        )
        logger.info(f"Successfully sent {len(records)} records to Kinesis")
    except Exception as e:
        logger.error(f"Failed to send records to Kinesis: {e}")
```