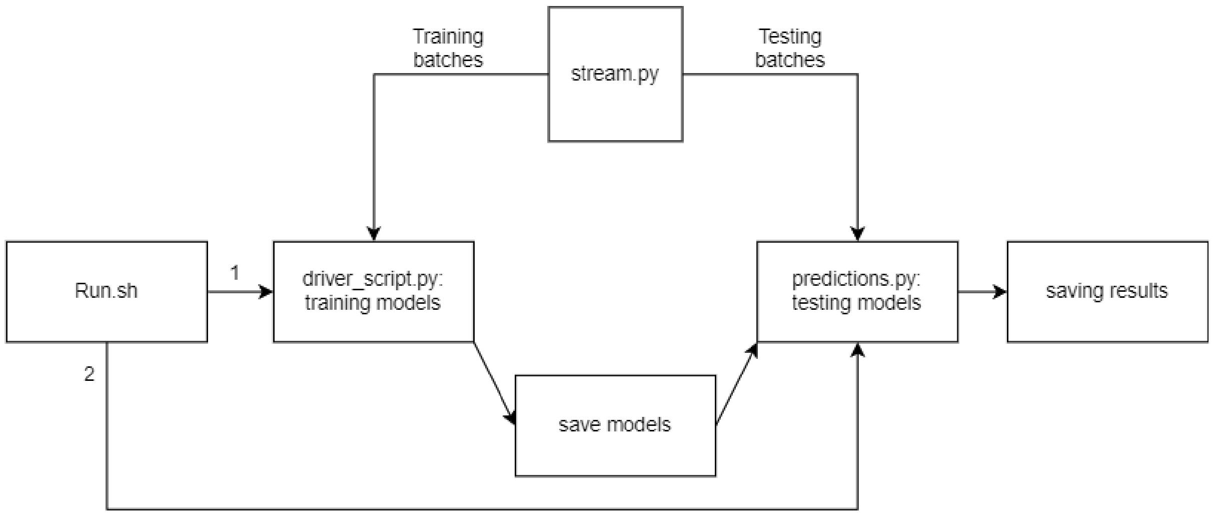
Project titles: Machine Learning with Spark MLlib using CIFAR 10 dataset

Team name: BD2_047_055_060
Team members: Anjali Praveen, Anusha M S, Apurva Pothumarthi

Design details:
The Project is run from the src folder by executing the run.sh script. This executes the driver_script.py first to train the machine learning and clustering models and then executes the predictions.py to evaluate the models and save the results.

Surface levels implementation:



As the above diagram indicates, The run.sh first executes driver_script.py along with stream.py for training batches. This was done by modifying the stream.py file to suit our needs. In this step, the driver_script initializes all the required models, preprocesses the data, trains, and then saves the models into individual files. After this, run.sh executes predictions.py along with stream.py for the testing batch. The script evaluates all the models and saves the results for future use. All the required custom functions that were written from scratch are stored in a separate file called custom_funcs.py.

Details about each unit:

1. **driver_script.py:**
    - SparkContext() is used to initialize Apache Spark. StreamingContext() makes a connection to spark. We will be taking data in batches with intervals of 5. socketTextStream() receives data from localhost and port number 6100.
    - Global models are created with a perceptron, SGD classifier, passive-aggressive classifier as our machine learning models whereas our cluster model is a sequential K-means cluster.
    - The driver_function() is defined and checks if the rdd is empty. If not, the JSON strings are converted into a NumPy array. These are divided into x_train and y_train. X_train is further processed to result in normal forms.
    - The normal forms and the y_train values are incrementally fit into the models decided above. Next, these are fitted in our cluster model.

    - The function above is finally executed with the socket stream. Models are exported with different prefixes to the models folder.

2. **predictions.py:**
    - Pickle is imported and used to convert the objects from the models into the stream.
    - A function model_prediction is defined which again checks if rdd is empty. If not the JSON is converted to NumPy arrays. Data is split into x_test and y_test and passed through the models.
    - Additionally, the metrics which are confusion matrix, accuracy, precisions, recalls, and F1 scores are evaluated.
    - The rdd is also passed through function cluster_predictions and arrays are stacked vertically and saved to the file which was initially opened. This will be closed at the end.
3. **custom_funcs.py:**
    - batch_convert() takes batches of data and converts them into JSON which is further turned into NumPy arrays in the above stages.
    - image_normalise() normalises NumPy arrays from batch_convert().
    - image_centers() brings the mean to zero.
    - image_standardise() brings the standard deviation to 1.
    - image_greyscale() converts the image from RGB to greyscale. Dimension is reduced to 2D.
    - image_preprocess() calls the above processing functions one after another.
    - Model functions are made to export them, evaluate metrics.
    - Finally, a class is created for sequential k-means clustering.
4. **stream.py:**
    - Added command-line arguments train_count for number of training batches(takes all training batches by default) and testing (boolean function) for testing batch.
5. **run.sh:**
    - The file is run first where the previous models are removed.
    - Next, it calls on stream.py and driver_script.py simultaneously.
    - Finally, it calls on stream.py and predictions.py.

The reason behind design decisions:
- run.sh is kept as the starting point.
- Functions from scratch are all grouped.
- The predictions and driver files are kept separate to train and test datasets separately.

Takeaway from project:
- Understanding how Apache Stream works.
- Working with processing and designing predicting models with image datasets.
- Understanding incremental learning models.
- Exploring different libraries and functions.