**Project Overview**

In this project, you will be provided with a real-world dataset, extracted from Kaggle, on San Francisco crime incidents, and you will provide statistical analyses of the data using Apache Spark Structured Streaming. You will draw on the skills and knowledge you've learned in this course to create a Kafka server to produce data, and ingest data through Spark Structured Streaming.

**Development Environment**

You may choose to create your project in the workspace we provide here, or if you wish to develop your project locally, you will need to set up your environment properly as described below:

* Spark 2.4.3
* Scala 2.11.x
* Java 1.8.x
* Kafka build with Scala 2.11.x
* Python 3.6.x or 3.7.x

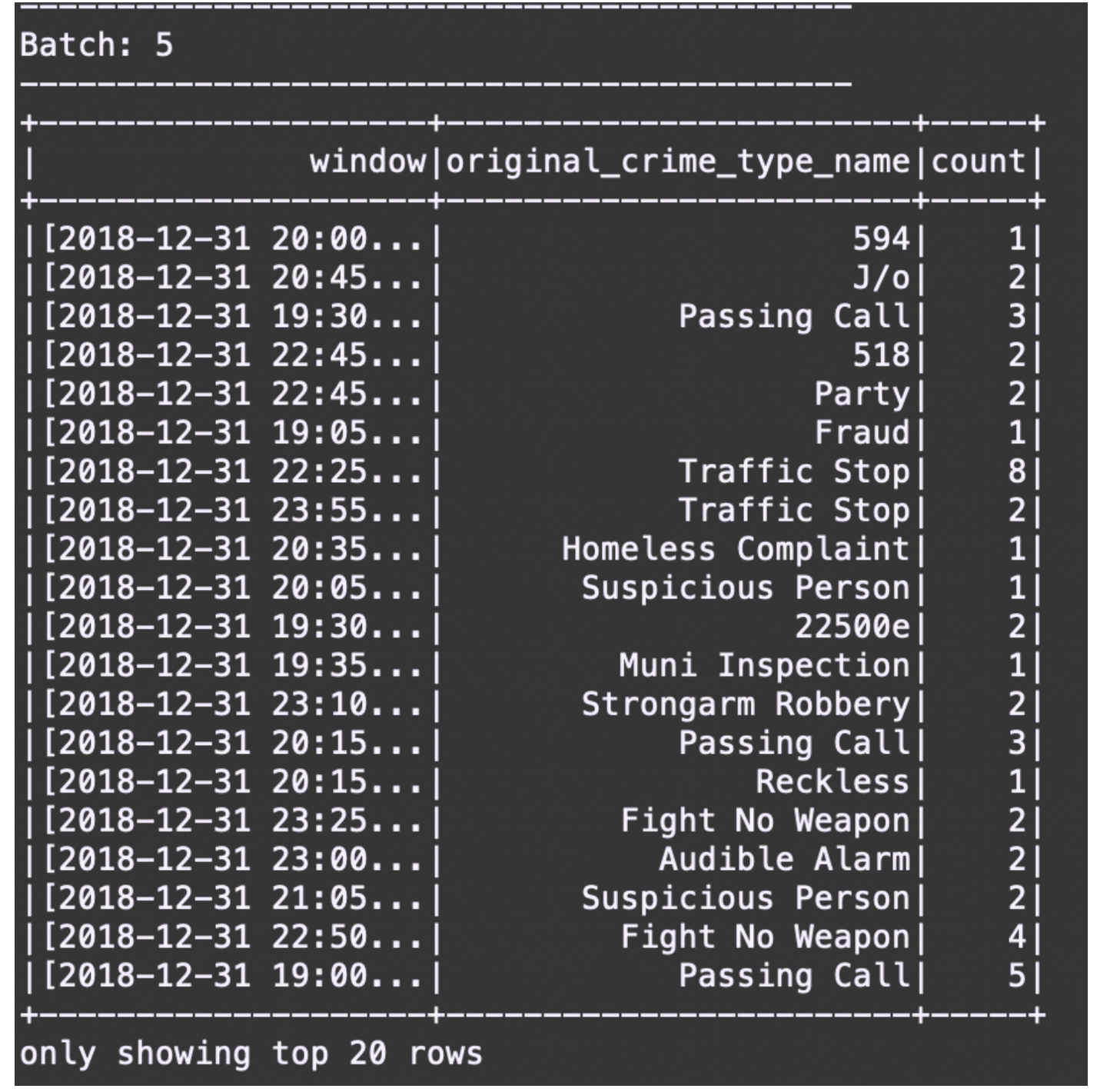
**Environment Setup (Only Necessary if You Want to Work on the Project Locally on Your Own Machine)**

**For Macs or Linux:**

* Download Spark from [**https://spark.apache.org/downloads.html**](https://spark.apache.org/downloads.html). Choose "Prebuilt for Apache Hadoop 2.7 and later."
* Unpack Spark in one of your folders (I usually put all my dev requirements in /home/users/user/dev).
* Download binary for Kafka from this location [**https://kafka.apache.org/downloads**](https://kafka.apache.org/downloads), with Scala 2.11, version 2.3.0. Unzip in your local directory where you unzipped your Spark binary as well. Exploring the Kafka folder, you’ll see the scripts to execute in bin folders, and config files under config folder. You’ll need to modify zookeeper.properties and server.properties.
* Download Scala from the official site, or for Mac users, you can also use brew install scala, but make sure you download version 2.11.x.
* Run below to verify correct versions:
* java -version
* scala -version
* Make sure your ~/.bash\_profile looks like below (might be different depending on your directory):
* export SPARK\_HOME=/Users/dev/spark-2.4.3-bin-hadoop2.7
* export JAVA\_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0\_181.jdk/Contents/Home
* export SCALA\_HOME=/usr/local/scala/
* export PATH=$JAVA\_HOME/bin:$SPARK\_HOME/bin:$SCALA\_HOME/bin:$PATH

**For Windows:**

Please follow the directions found in this helpful StackOverflow post: [**https://stackoverflow.com/questions/25481325/how-to-set-up-spark-on-windows**](https://stackoverflow.com/questions/25481325/how-to-set-up-spark-on-windows)



## Starter Code

You can find three Python files that are starter code, the project dataset, and some other necessary resources in a zip file called "SF Crime Data Project Files" in the Resources tab in the left sidebar of your classroom:

* producer\_server.py
* kafka\_server.py
* data\_stream.py
* police-department-calls-for-service.json
* radio\_code.json
* start.sh
* requirements.txt

These files are also included in the Project Workspace.

## Files You Need to Edit in Your Project Work

These starter code files should be edited:

* producer\_server.py
* data\_stream.py
* kafka\_server.py

The following file should be created separately for you to check if your kafka\_server.py is working properly:

* consumer\_server.py

## Create a GitHub Repository

Create a new repo that will contain all these files for your project. You will submit a link to this repo as a key part of your project submission. If you complete the project in the classroom workspace here, just download the files you worked on and add them to your repo.

## Beginning the Project

This project requires creating topics, starting Zookeeper and Kafka servers, and your Kafka bootstrap server. You’ll need to choose a port number (e.g., 9092, 9093..) for your Kafka topic, and come up with a Kafka topic name and modify the zookeeper.properties and server.properties appropriately.

#### Local Environment

* Install requirements using ./start.sh if you use conda for Python. If you use pip rather than conda, then use pip install -r requirements.txt.
* Use the commands below to start the Zookeeper and Kafka servers. You can find the bin and config folder in the Kafka binary that you have downloaded and unzipped.
* bin/zookeeper-server-**start**.sh config/zookeeper.properties
* **bin**/kafka-**server**-**start**.sh config/**server**.properties
* You can start the bootstrap server using this Python command: python producer\_server.py.

#### Workspace Environment

* Modify the zookeeper.properties and producer.properties given to suit your topic and port number of your choice. Start up these servers in the terminal using the commands:
* /usr/bin/zookeeper-server-**start** zookeeper.properties
* /usr/**bin**/kafka-**server**-**start** producer.properties
* You’ll need to open up two terminal tabs to execute each command.
* Install requirements using the provided ./start.sh script. **This needs to be done every time you re-open the workspace, or anytime after you've refreshed, or woken up, or reset data, or used the "Get New Content" button in this workspace.**
* In the terminal, to install other packages that you think are necessary to complete the project, use conda install <package\_name>. You may need to reinstall these packages every time you re-open the workspace, or anytime after you've refreshed, or woken up, or reset data, or used the "Get New Content" button in this workspace.

## Step 1

* The first step is to build a simple Kafka server.
* Complete the code for the server in producer\_server.py and kafka\_server.py.

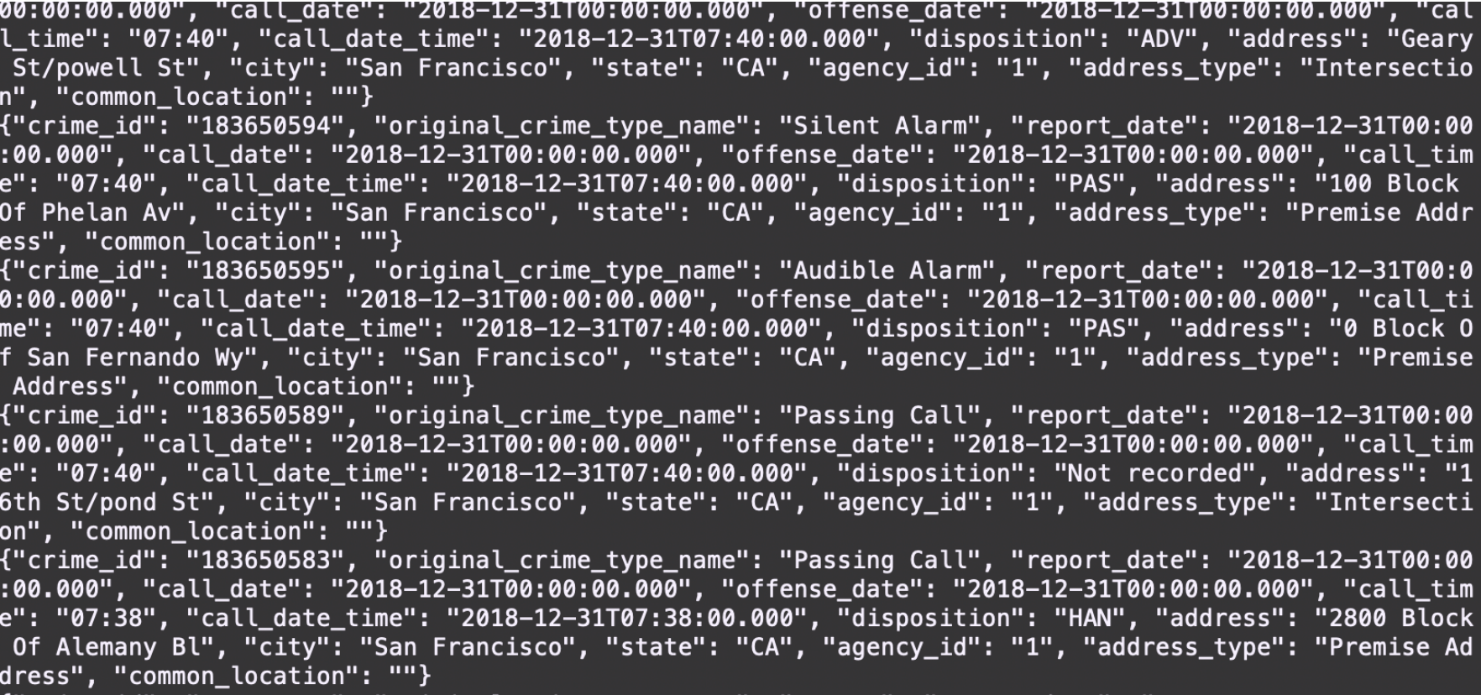
#### Local Environment

* To see if you correctly implemented the server, use the command bin/kafka-console-consumer.sh --bootstrap-server localhost:<your-port-number> --topic <your-topic-name> --from-beginning to see your output.

#### Workspace Environment

* To start kafka-consumer-console, use the command /usr/bin/kafka-consumer-console.

**Take a screenshot of your kafka-consumer-console output. You will need to include this screenshot as part of your project submission.**

[[](https://classroom.udacity.com/nanodegrees/nd029/parts/d3d2cbfa-dc05-44c3-8db8-84e1d931170d/modules/36ddbb88-7b71-4e78-983c-5a0890eb1ec2/lessons/9a62b3fd-3586-47a1-a1c9-5c727d4ec3b4/concepts/0165e25d-9da1-4c92-a3cb-ffc56c5eef4f)](https://classroom.udacity.com/nanodegrees/nd029/parts/d3d2cbfa-dc05-44c3-8db8-84e1d931170d/modules/36ddbb88-7b71-4e78-983c-5a0890eb1ec2/lessons/9a62b3fd-3586-47a1-a1c9-5c727d4ec3b4/concepts/0165e25d-9da1-4c92-a3cb-ffc56c5eef4f)

**[Sample Kafka Consumer Console Output](https://classroom.udacity.com/nanodegrees/nd029/parts/d3d2cbfa-dc05-44c3-8db8-84e1d931170d/modules/36ddbb88-7b71-4e78-983c-5a0890eb1ec2/lessons/9a62b3fd-3586-47a1-a1c9-5c727d4ec3b4/concepts/0165e25d-9da1-4c92-a3cb-ffc56c5eef4f)**

## Step 2

* Apache Spark already has an integration with Kafka brokers, so we would not normally need a separate Kafka consumer. However, we are going to ask you to create one anyway. Why? We'd like you to create the consumer to demonstrate your understanding of creating a complete Kafka Module (producer and consumer) from scratch. In production, you might have to create a dummy producer or consumer to just test out your theory and this will be great practice for that.
* Implement all the TODO items in data\_stream.py. You may need to explore the dataset beforehand using a Jupyter Notebook.
* Do a spark-submit using this command: spark-submit --packages org.apache.spark:spark-sql-kafka-0-10\_2.11:2.3.4 --master local[\*] data\_stream.py.
* Take a screenshot of your progress reporter after executing a Spark job. **You will need to include this screenshot as part of your project submission.**
* Take a screenshot of the Spark Streaming UI as the streaming continues. **You will need to include this screenshot as part of your project submission.**

## Step 3

Write the answers to these questions in the README.md doc of your GitHub repo:

1. How did changing values on the SparkSession property parameters affect the throughput and latency of the data?
2. What were the 2-3 most efficient SparkSession property key/value pairs? Through testing multiple variations on values, how can you tell these were the most optimal?

### Project Submission

You will submit a link to your GitHub repo, with the files you've created: producer\_server.py, kafka\_server.py, data\_stream.py, and consumer\_server.py. The README.md doc in your GitHub repo should contain your responses to the two questions from Step 3.

Your project submission should also include a zip file containing the three screenshots you've taken.