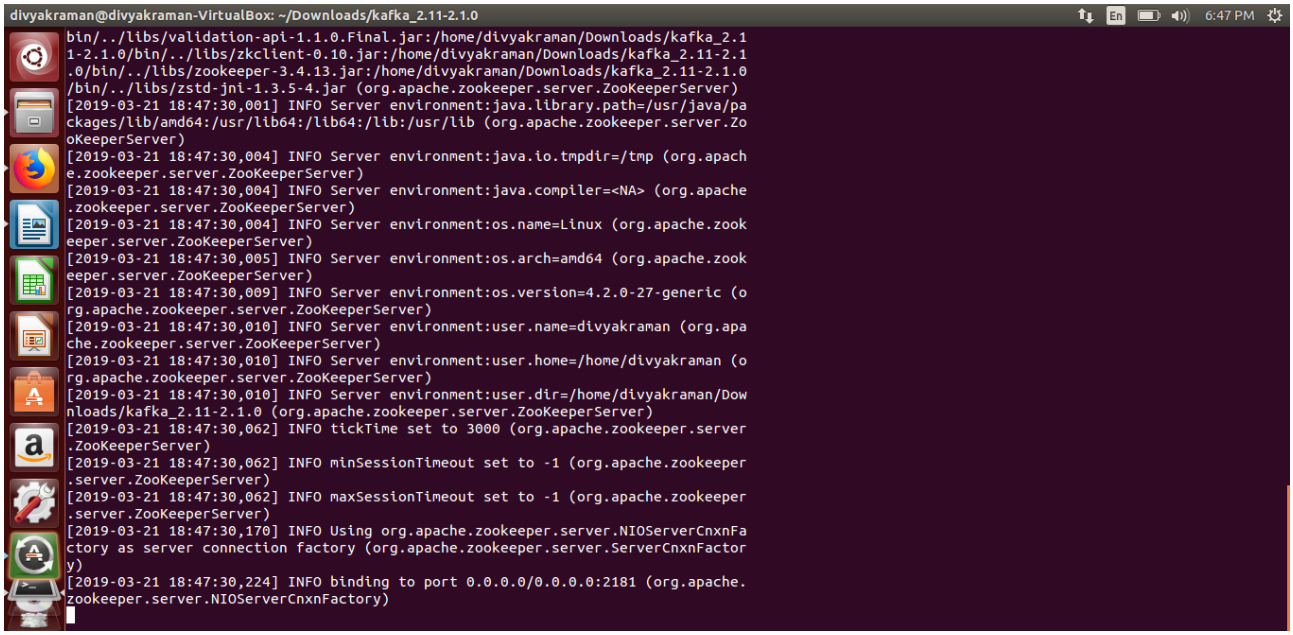
CS4830: Big Data Laboratory
Divya K Raman, EE15B085
Lab 6

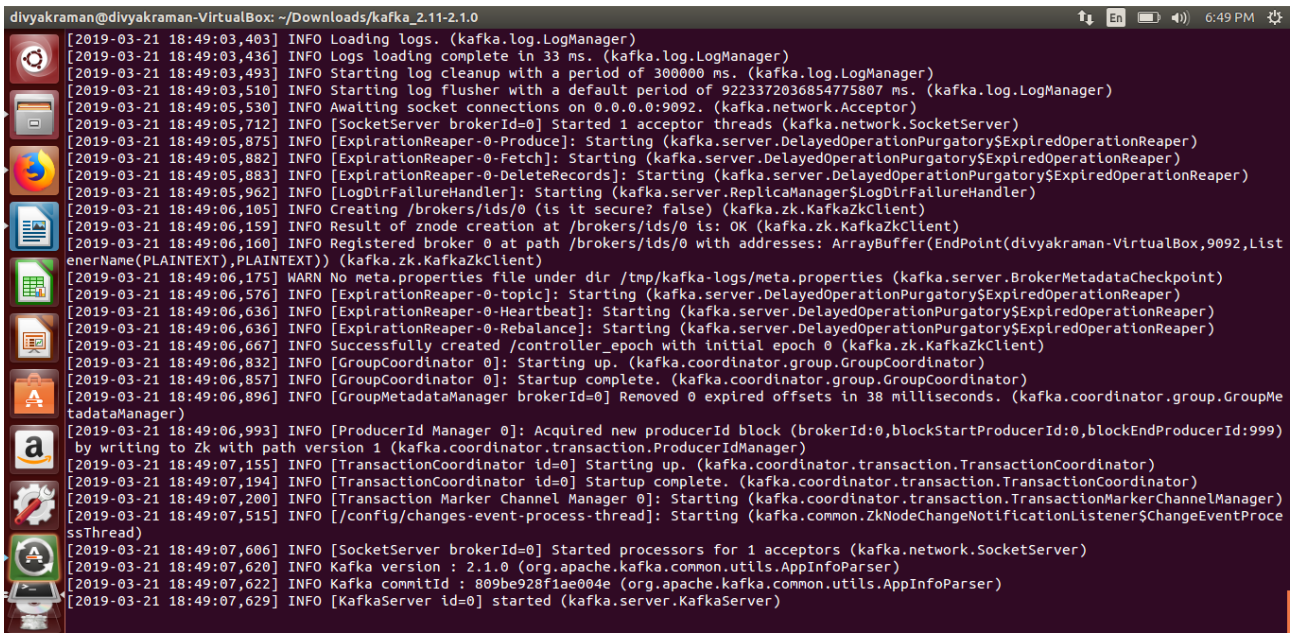
In this assignment, we work with kafka and kafka-python. The end goal is to build a producer consumer model for kafka.

We first install kafka locally and install kafka-python(python connector of kafka) using pip. The latest version of kafka is downloaded and we un-tar it. All work is to be done inside this folder and hence change the directory using the command cd kafka_2.11-2.1.0. Note that the local machine needs to have java installed.

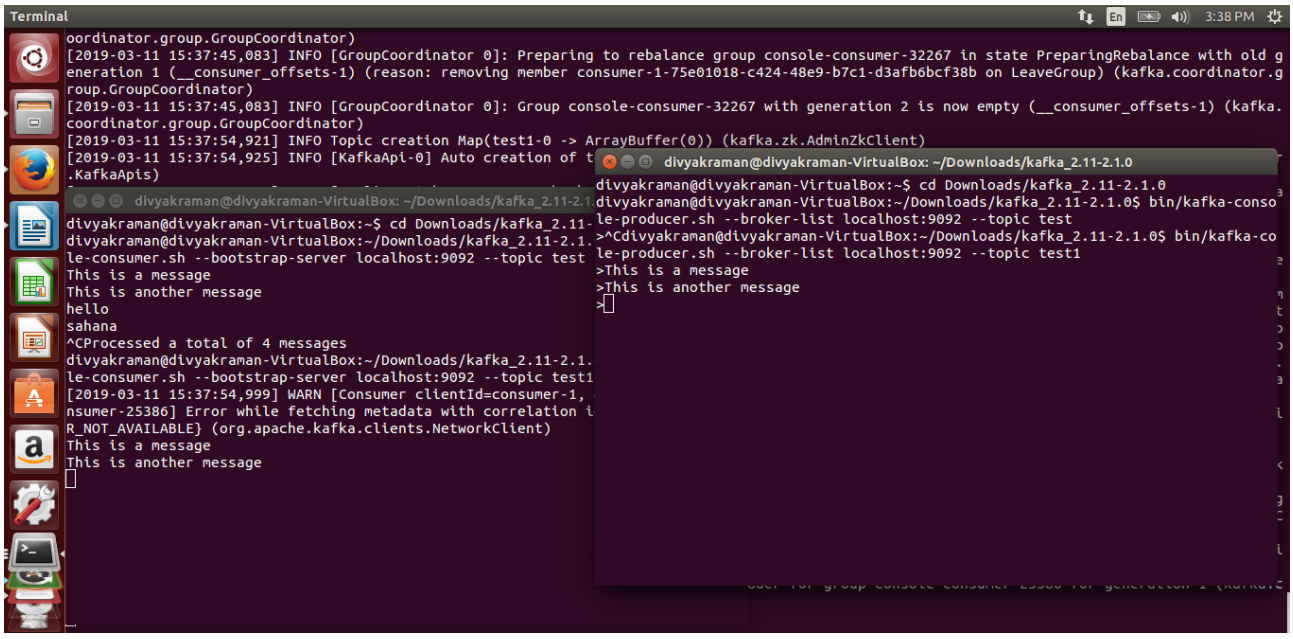
We next start zookeeper and keep it running in one terminal. Command: bin/zookeeper-server-start.sh config/zookeeper.properties



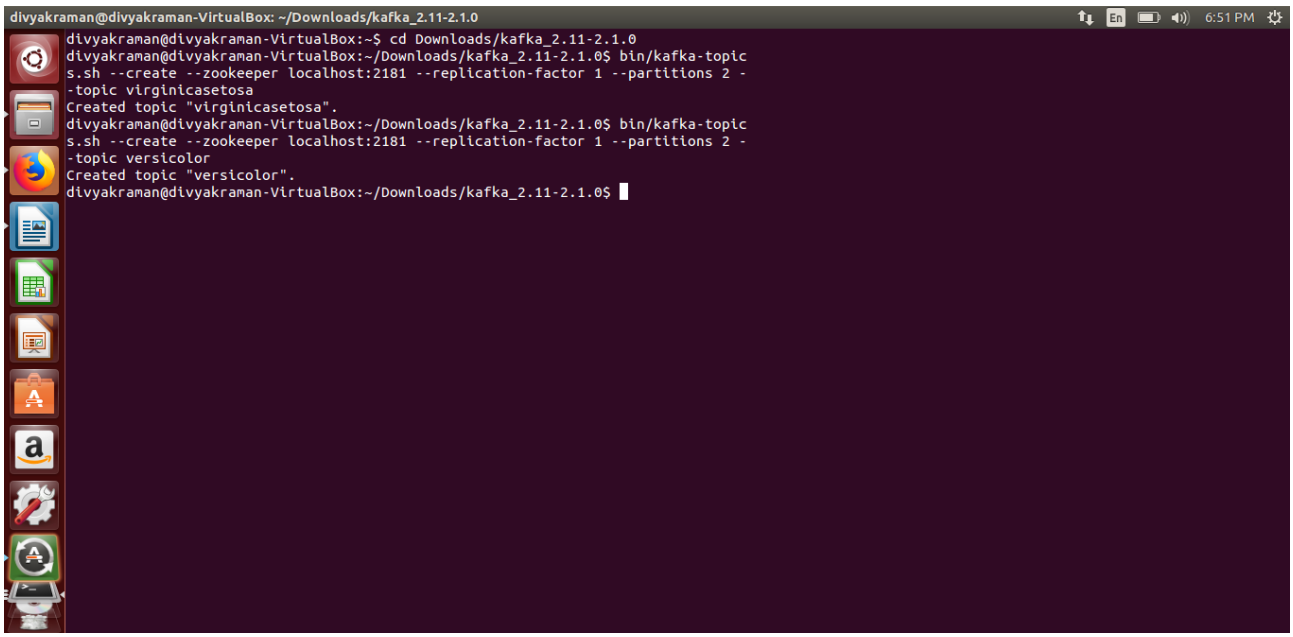
Next, in another terminal, start the kafka server. bin/zookeeper-server-start.sh config/zookeeper.properties



We next send some messages and receive some messages:



We now start a single kafka broker and create a topic called virginicasetosa with 2 partitions, create another topic called versicolor with 2 partitions for both which the replication factor is 1



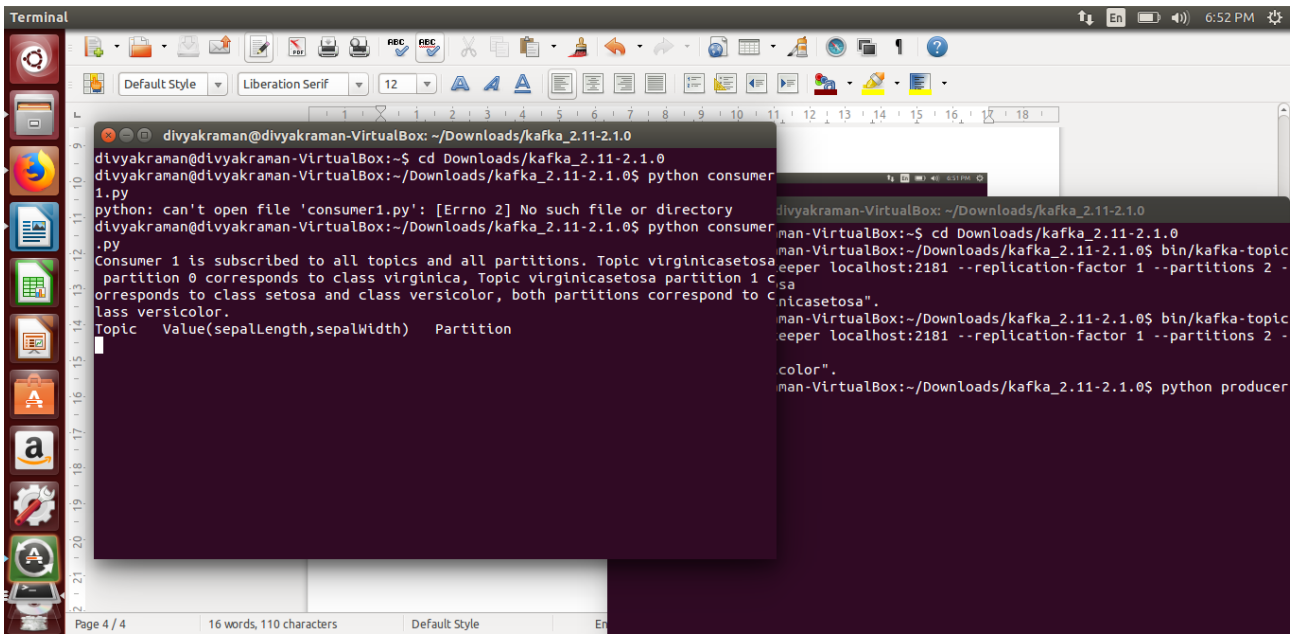
consumer.py has the code for the consumer as per the specifications given.
Producer1.py has the code for the producer as per the specifications given.

In the consumer code, each consumer has a timeout of 10 seconds and will exit from the consumer message loop if no message arrives within the next 10 seconds. This is to ensure that the rest of the program below the consumer loop gets executed. Seek_to_beginning ensures that the consumer takes action on all messages from the beginning on the topic and partition it is assigned to and not just on the messages that arrive at the moment.

In the producer code, we have a numpy encoder class which configures to send json as only binary strings can be sent and interpreted.

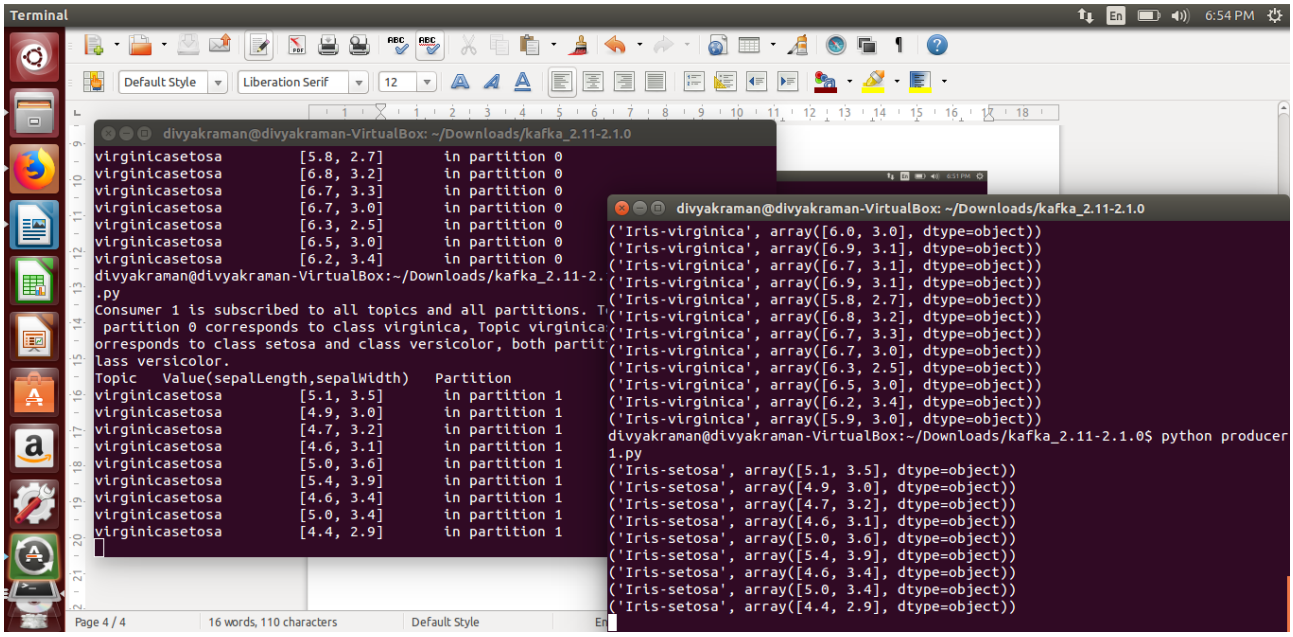
In one terminal, run consumer.py and in another terminal, within the next 10 seconds, run producer1.py. If the producer script is not run within the next 10 seconds, the consumer script will exit assuming that no messages will arrive any further. This time lag can be increased by changing the value in consumer_timeout_ms while defining a consumer.

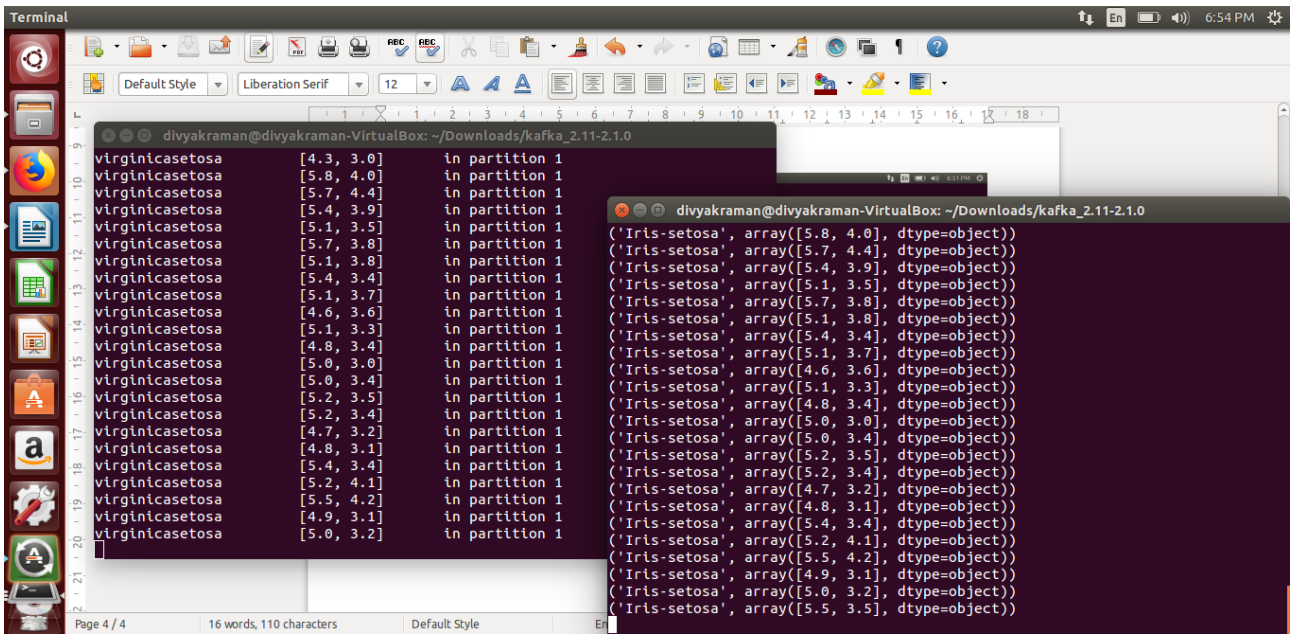
Just when it starts running:

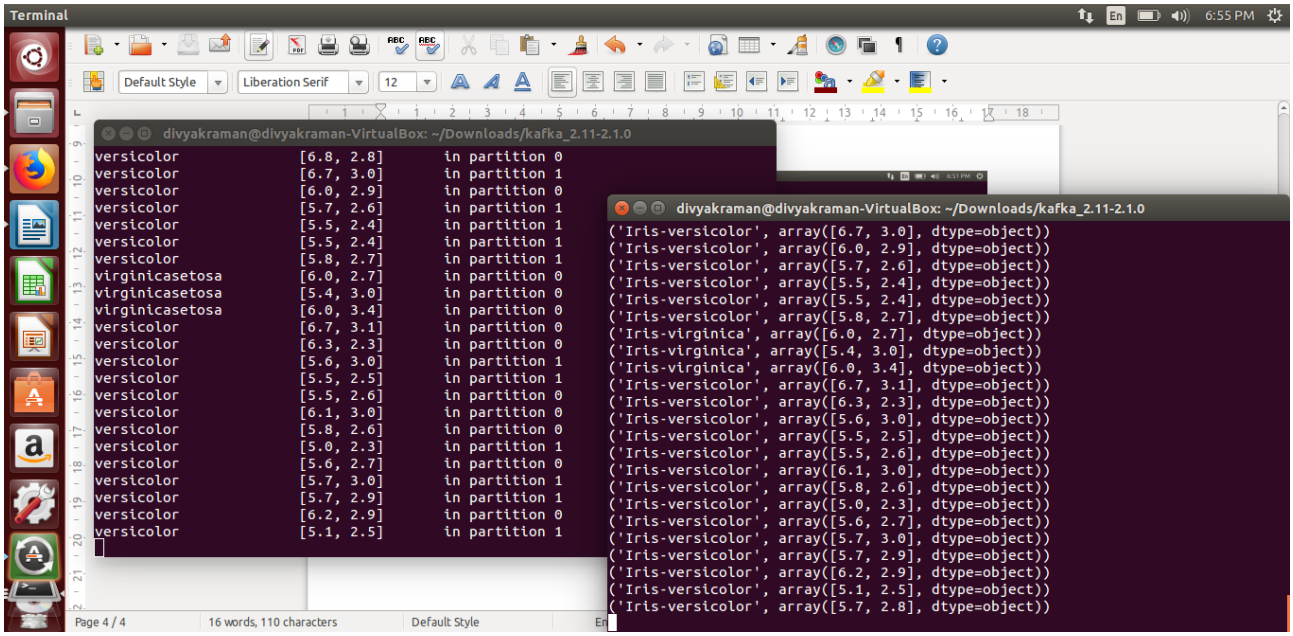


Consumer 1:

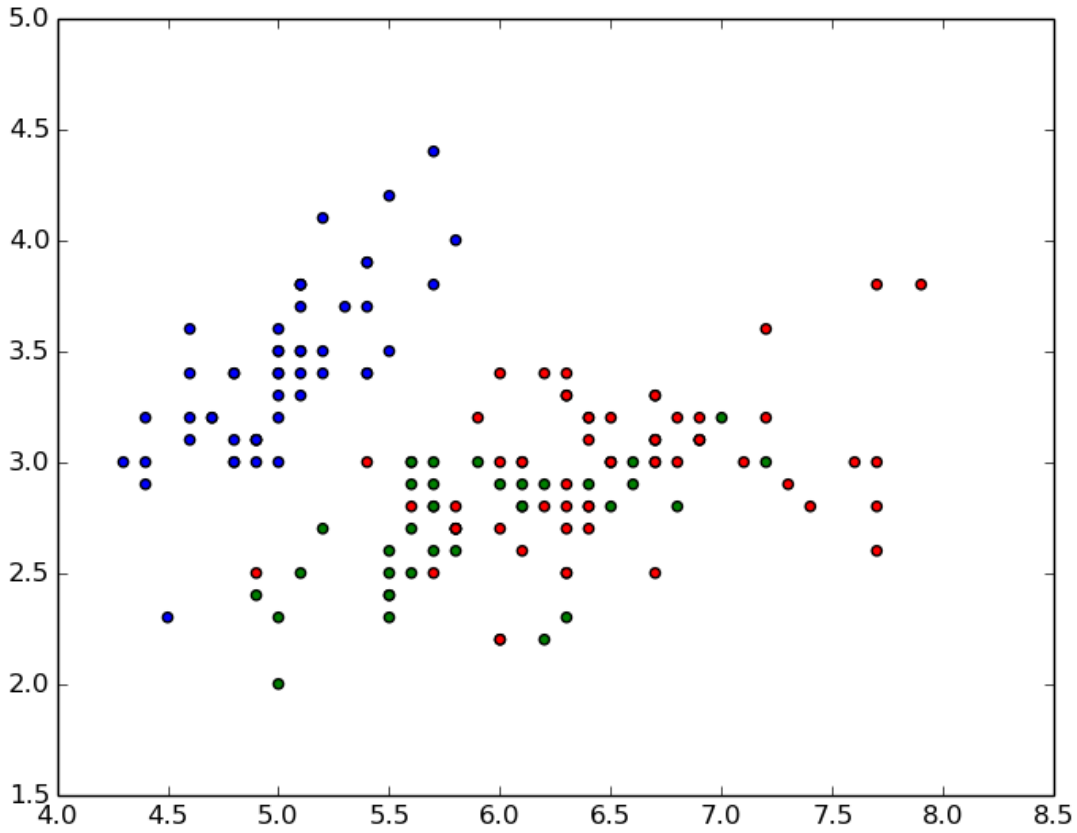
Consumer 1 is subscribed to all topics and all partitions. Topic virginicasetosa partition 0 corresponds to class virginica, Topic virginicasetosa partition 1 corresponds to class setosa and class versicolor, both partitions correspond to class versicolor.





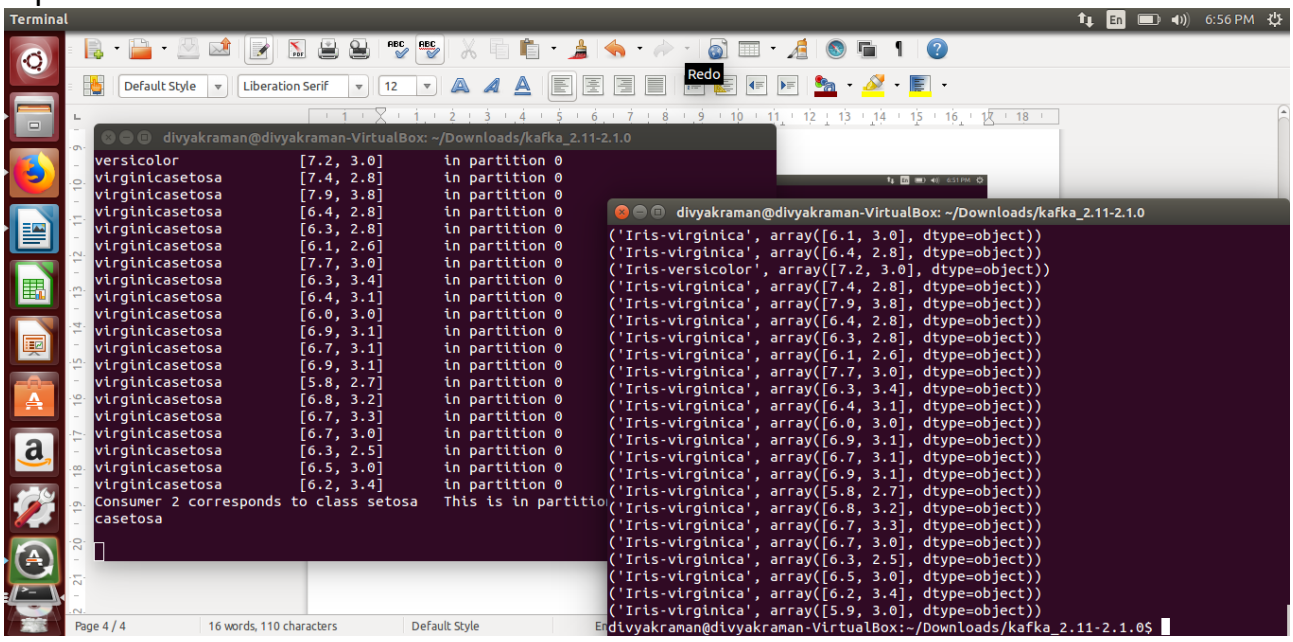


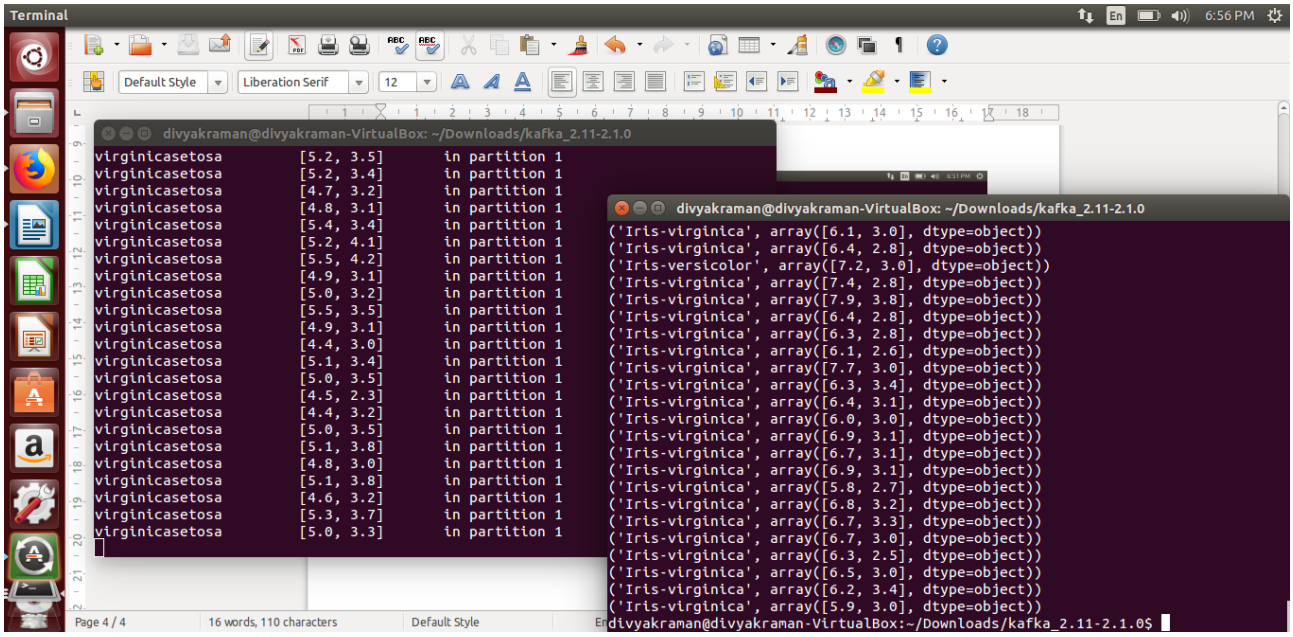
Consumer 1 also creates a plot of sepal Length vs Sepal Width. Class virginica is in red, class setosa is in blue and class versicolor is in green. This is a plot of sepal length vs sepal width.



Consumer 2:

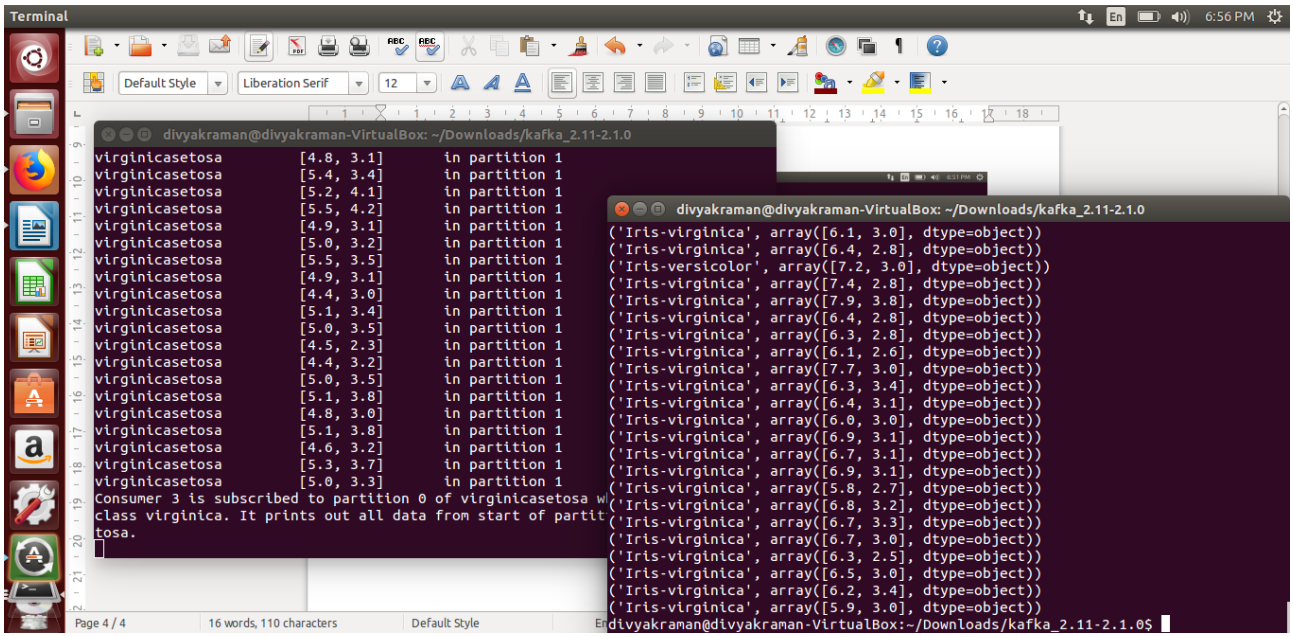
Consumer 2 corresponds to class setosa. This is in partition 1 of topic virginicasetosa. It prints a table to stdout.

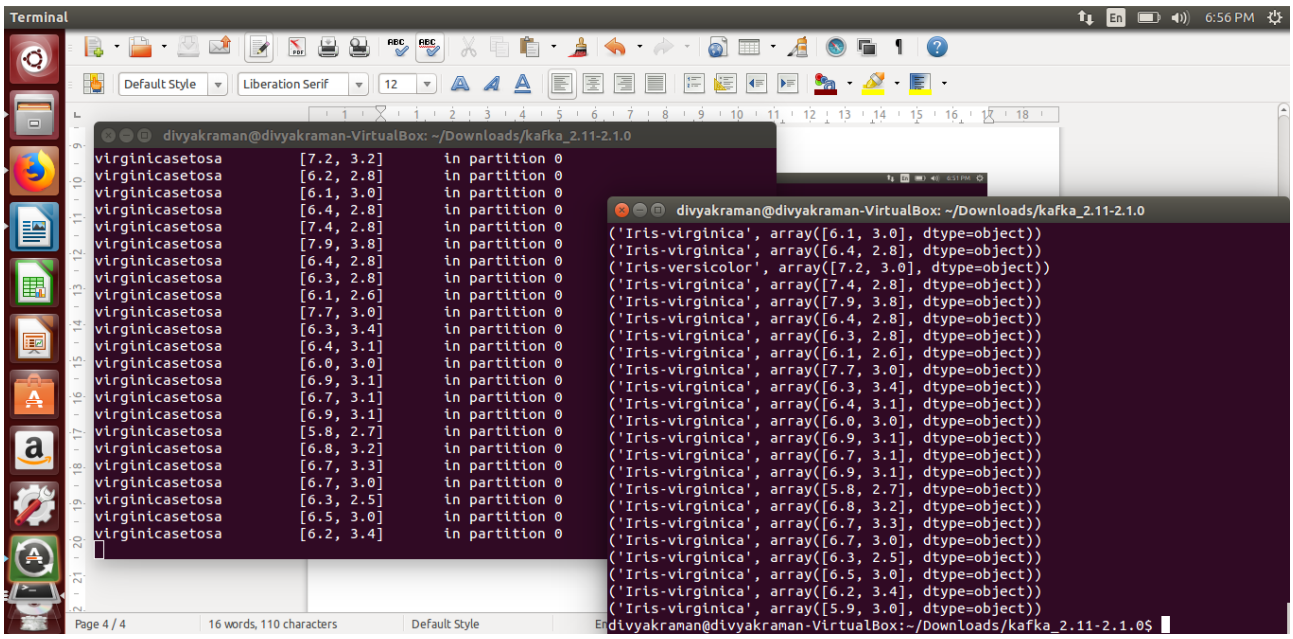


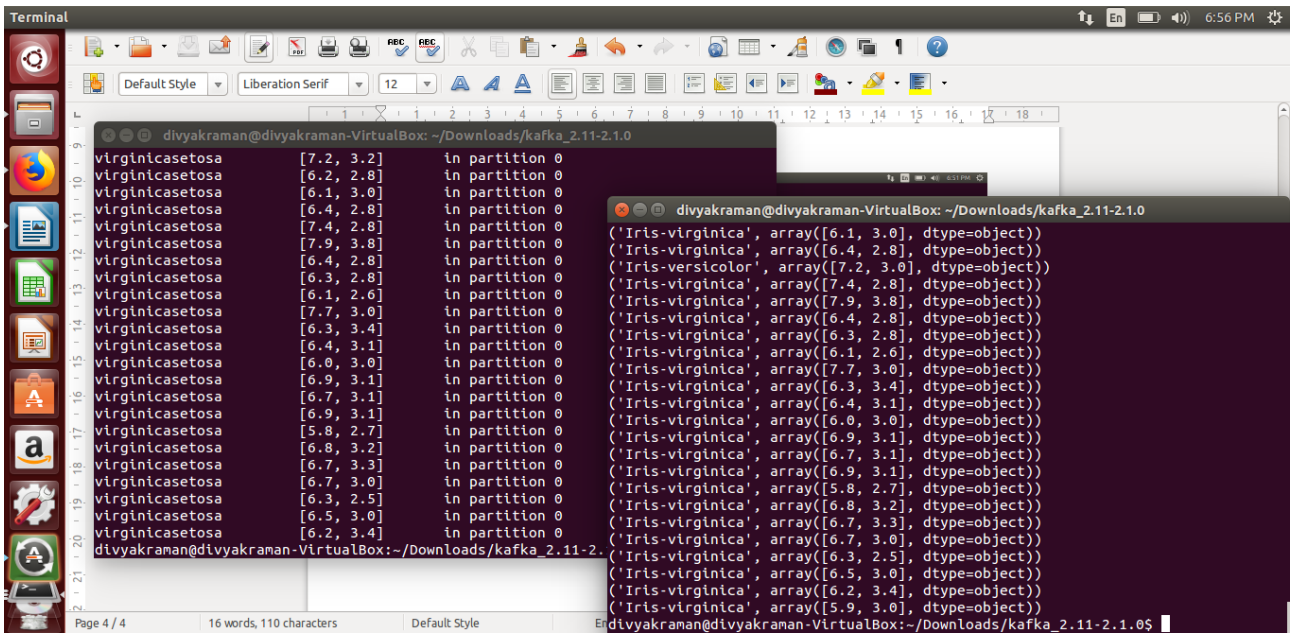


Consumer 3:

Consumer 3 is subscribed to partition 0 of virginicasetosa which corresponds to class virginica. It prints out all data from start of partition 0 of virginicasetosa.







Question: The Producer and Consumer APIs have a collection of other configurations/keyword arguments that determine how they function. Describe any 2 such configurable variable for the Producer and Consumer (two for Producer and two for Consumer)

Producer:

key_serializer – used to convert user-supplied keys to bytes. If not None, it should return bytes. Default value is none.

Partitioner –used to determine which partition each message is assigned to. When a key is None, the message is delivered to a random partition.

Consumer:

group_id – The name of the consumer group to join for dynamic partition assignment, and to use for fetching and committing offsets. If the value is None, auto-partition assignment (via group coordinator) and offset commits are disabled. The deafault value is None.

consumer_timeout_ms – number of milliseconds to block during message iteration before raising StopIteration (i.e., ending the iterator). Default value is forever i.e. [float('inf')].

References:

1. https://kafka-python.readthedocs.io/en/master/apidoc/KafkaConsumer.html
2. https://kafka-python.readthedocs.io/en/master/apidoc/KafkaProducer.html

Question: What is the role of zookeeper?

Kafka is a distributed system and uses Zookeeper to track status of kafka cluster nodes. It also keeps track of Kafka topics, partitions etc. Kafka uses Zookeeper for electing a controller. The controller is one of the brokers and is responsible for maintaining the leader/follower relationship for all the partitions. When a node shuts down, it is the controller that tells other replicas to become partition leaders to replace the partition leaders on the node that is going away. Zookeeper is used to elect a controller, make sure there is only one and elect a new one it if it crashes. Zookeeper also manages cluster membership (tracking which brokers are alive and part of the cluster). Zookeeper also does topic configuration which tracks what topics exist, the number of partitions each has, where the replicas are, the preferred leader and what configuration overrides are set for each topic.

Reference:

https://stackoverflow.com/questions/23751708/is-zookeeper-a-must-for-kafka