

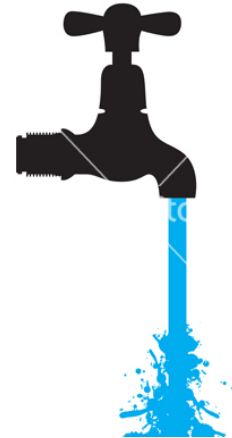
Streaming Architecture for Predictive Modeling

Keira Zhou

Data Engineer @ Capital One Labs

Batch vs. Streaming

- Batch
 - Run a process in a scheduled way
 - Good for computation on huge amount of historical data
 - Complex analysis, hard to compute in real time
- Streaming
 - Continuous
 - “Life as it happens”: process data as it comes in
 - Traffic information
 - Heart beat monitoring
 - Capital One: double swipe



Task and Dataset

- Task: Predict Phishing website
 - e.g. Your password for x website is expiring, update at: <http://abc.com/update>
 - Predict if a website URL is a phishing website
- Phishing Websites Data Set
 - From UCI: <https://archive.ics.uci.edu/ml/datasets/Phishing+Websites#>
 - Extracted features + Labeled data points

Feature Selection

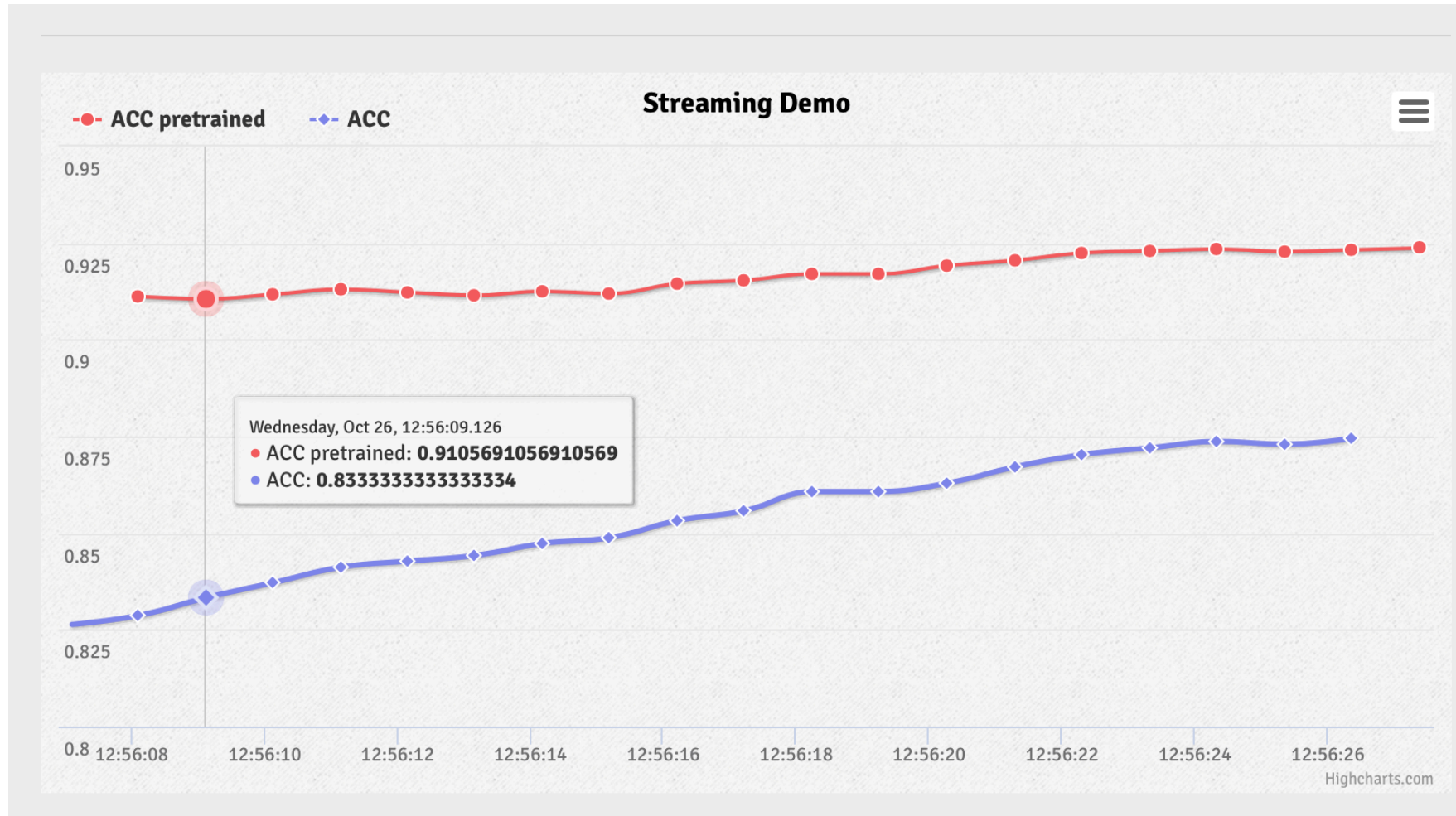
- Description provided in the dataset
- Some examples
 - Using URL Shortening Services “TinyURL”:
 - bit.ly/19DXSk4
 - Age of Domain:
 - minimum age of the legitimate domain is 6 months
 - Adding Prefix or Suffix Separated by (-) to the Domain:
 - <http://www.confirme-paypal.com/>

Train and Predict

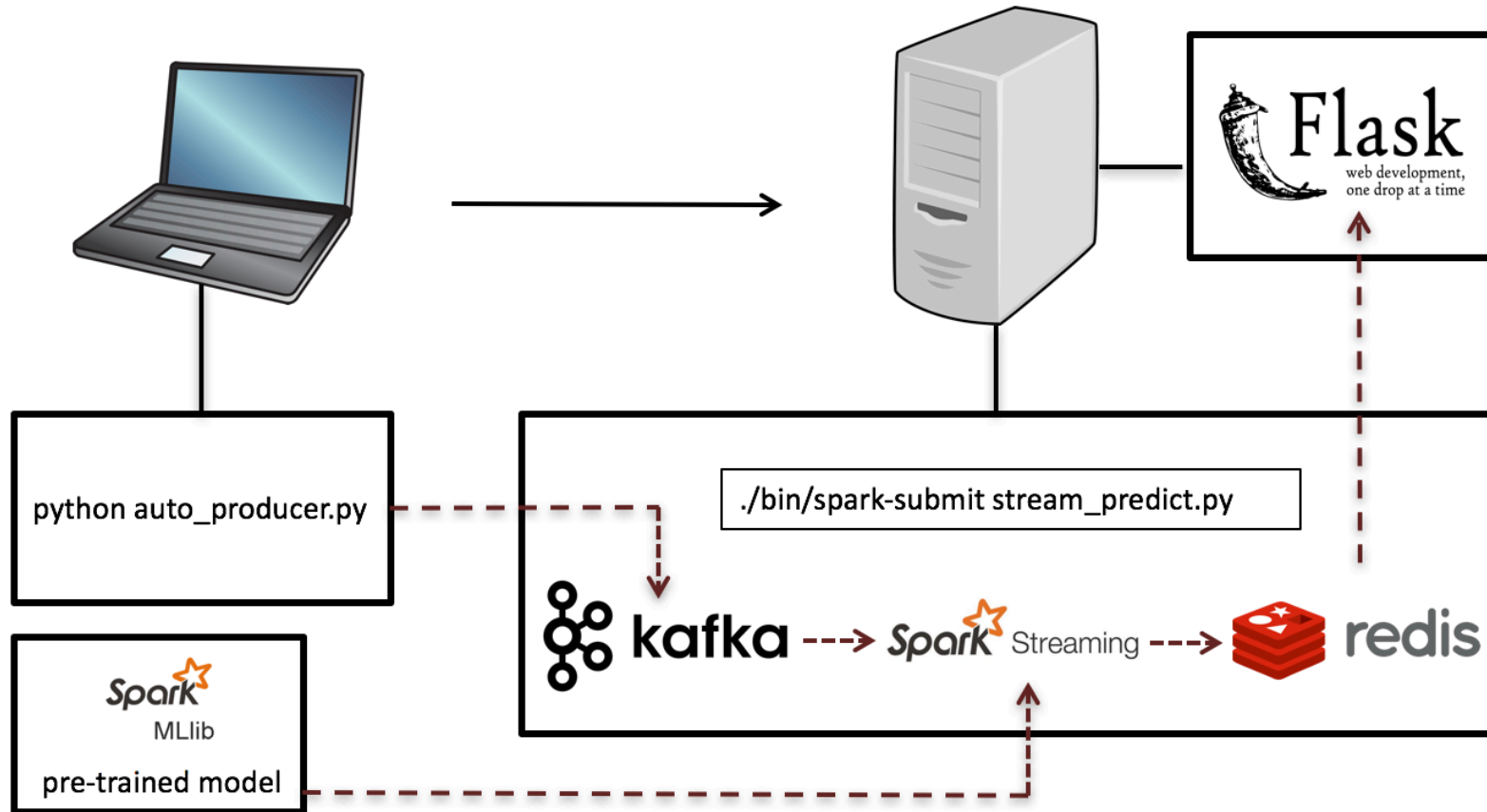
	Pre-trained Model	Online Model Updating	Online Prediction
Method 1	YES	NO	YES
Method 2	NO	YES	YES
Method 3	YES	YES	YES

Demo: Method 2 & Method 3

Online updating with and without a pre-trained model

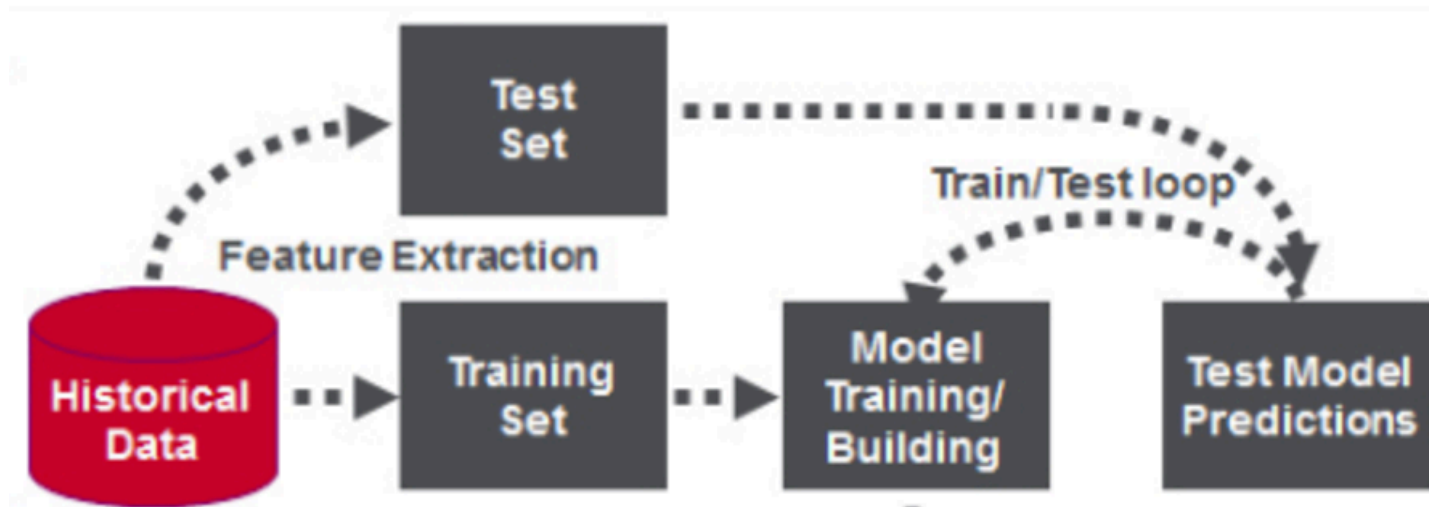


Backend



Train on Historical Data

- Classification problem
- Logistic Regression with Stochastic Gradient Descent



Code Snippet #1

- Spark 2.0 feature: Save model to file

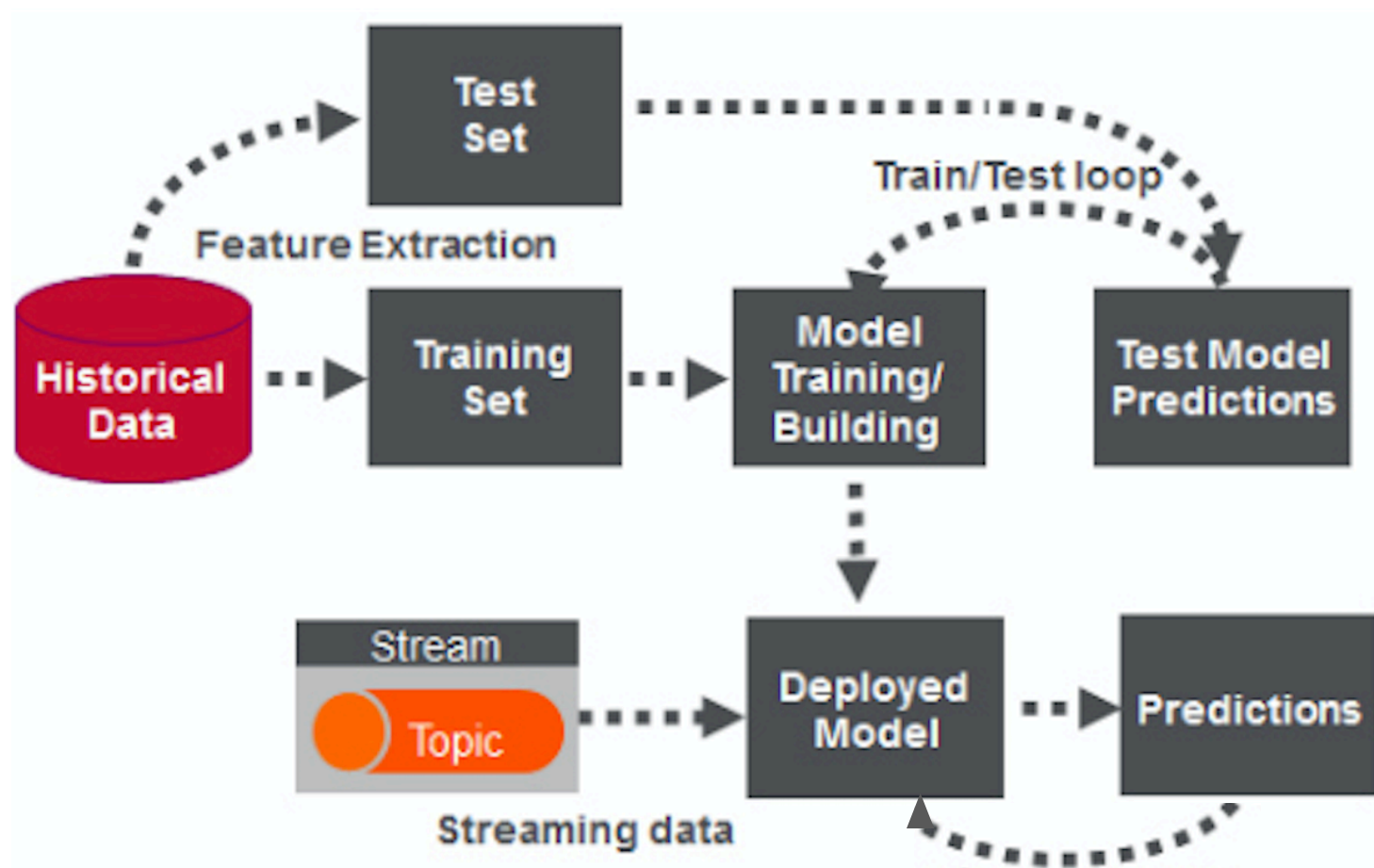
```
model = LogisticRegressionWithSGD.train(train_data, regType="l2")  
model.save(sc, "model/SGD")
```

- Load model file

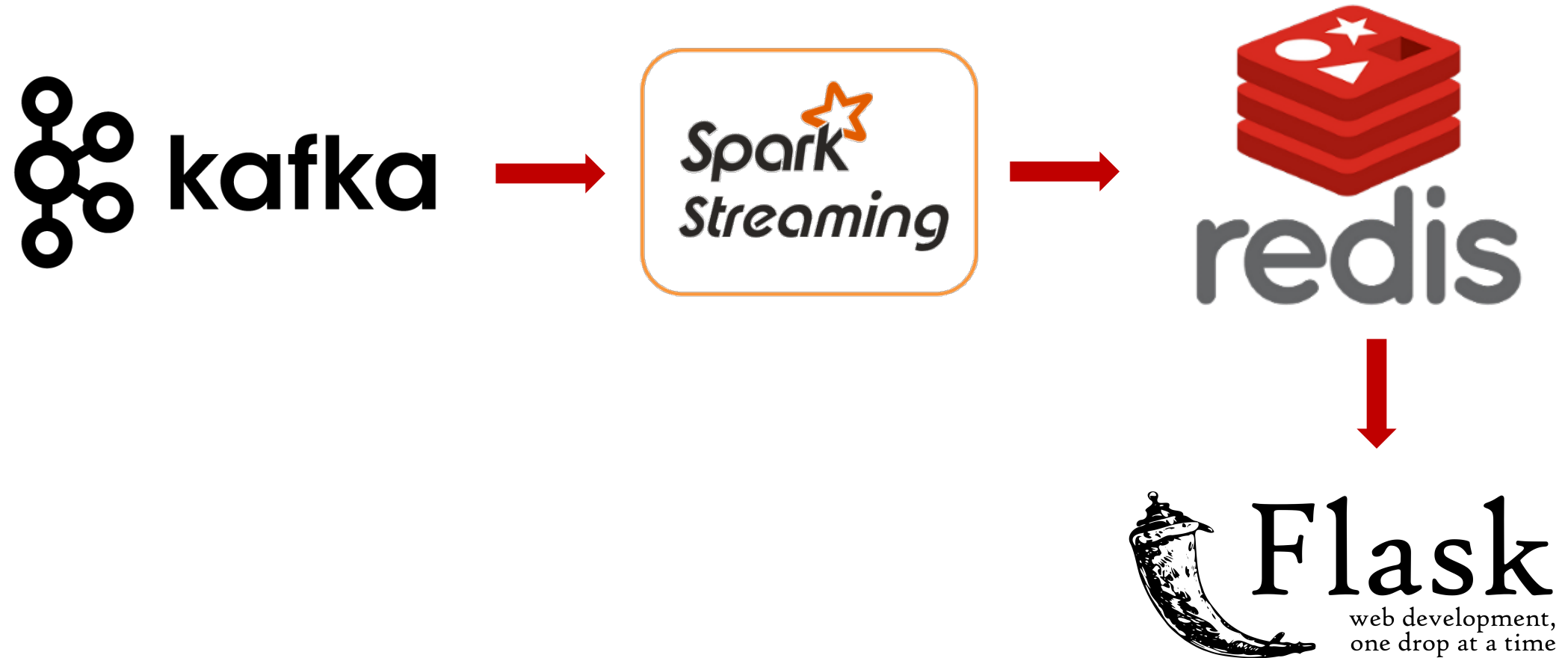
```
trained_model = LogisticRegressionModel.load(sc, "model/SGD")
```

Predict & Update

- Streaming Logistic Regression Model with Stochastic Gradient Descent



Build the Streaming Pipeline



Code Snippet #2

- Kafka -> Spark Streaming: Spark Streaming Kafka connector

```
# load data from Kafka
directKafkaStream = KafkaUtils.createDirectStream(
    ssc,
    ["auto_trnx"],
    {"metadata.broker.list": KAFKA_BROKERS}
)
```

- **Dependency:** spark-streaming-kafka-0-8_2.11-2.0.1.jar
 - Use the right version
 - Make sure you add the jar and its dependencies to your project

Parallelism in Spark

- Parallelism in Data Receiving
 - For example, split Kafka input DStream based on topics
 - One Kafka Input stream for one topic to receive data in parallel
 - numStreams = 5
 - kafkaStreams = [KafkaUtils.createStream(...) **for** _ **in** range (numStreams)]
- Parallelism in Data Processing
 - conf = SparkConf().setMaster("local[2]")
 - Meaning - 2 threads: “minimal” parallelism
 - Spark Cluster: YARN, Mesos, Standalone

Code Snippet #3

- Spark Streaming -> Redis: Python Redis connector (pip install redis)

```
def redisSink(rdd):  
    def _add_redis(prediction):  
        conn = redis.StrictRedis(  
            host='localhost',  
            port=6379,  
            db=0,  
            decode_responses=True  
        ) # seperate connection for each RDD  
        cache_key = "%s:%s" % (prediction[0], prediction[1]) # key "predicted_result:actual_label"  
        value = 1  
        if conn.exists(cache_key):  
            value = conn.get(cache_key)  
            value = int(value) + 1  
        conn.set(cache_key, value)  
    rdd.foreach(_add_redis)
```

Things to Consider

- Data Science
 - Find the right features
 - Get labeled data
 - Manual labeling
- Data Engineering
 - Connectors: Spark connector or Python connector
 - Tuning Spark:
 - # of drivers
 - # of executors
 - Memory
 - level of parallelism

#whoami

- Data Engineer @ Capital One Labs
- Previous:
 - Fellow @ Insight Data Engineering
 - BS + MS in Systems Engineering @UVA
- Github Repo:
 - <https://github.com/keiraqz/SparkModeling>

Questions?