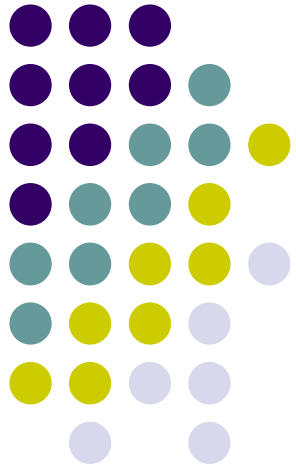
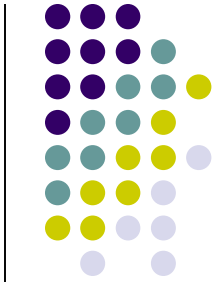
Big Data

November 28, 2020

Mike Spertus

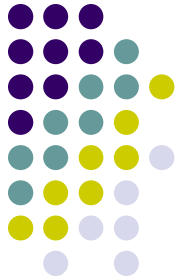
mike_spertus@symantec.com





QUICK INTRO TO SPARK MACHINE LEARNING

If it SparkSQL looks just like hive, why bother?

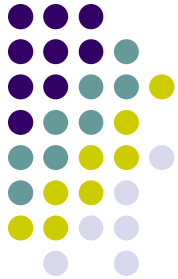


- As pointed out in class, the SQL code in SparkBatchLayer.scala is literally cut and pasted from our hive scripts
- In fact, the WriteToHBase.hql code still will need to be done in hive because Spark doesn't integrate as well with Hbase and multiple SQL stores
- In fact, for our existing batch layer, there is no real reason to do it in Spark rather than Hive (except maybe personal preference)
- So what's the point?

The point: Machine Learning with Spark

- If your batch views are too complex to write in SQL, you can use SparkSQL and supplement it with whatever code and libraries you want
- In particular, Spark comes with a great Machine Learning library call MLlib
- Algorithms that are designed for classification, regression, clustering, and recommendation engines
- Good book: Pentreath, *Machine Learning with Spark*
- Warning: while MLlib is great, it is important to understand what it is and isn't
  - It only contains "Big Data algorithms"
  - I.e., algorithms that run across a cluster
  - If you want to use a standard "non-parallel" algorithm across many small datasets, use `map()` or `parallelize()` to run algorithms from traditional machine learning libraries like WEKA

Machine learning examples



- While in-depth coverage of Machine Learning with Spark would take a whole course, we can walk through a surprisingly powerful example in a few minutes
- We will walk through an email spam detector from Karau, Konwinski, Wendell, and Zaharia's book *Learning Spark*
- Files in Canvas
  - MLLibPipelineInSparkShell.scala to run in Spark shell
  - MLLibPipeline.scala to build and spark-submit
- Far from required, but if you do something like this in your project, I will award some bonus points

Machine Learning: Big Picture

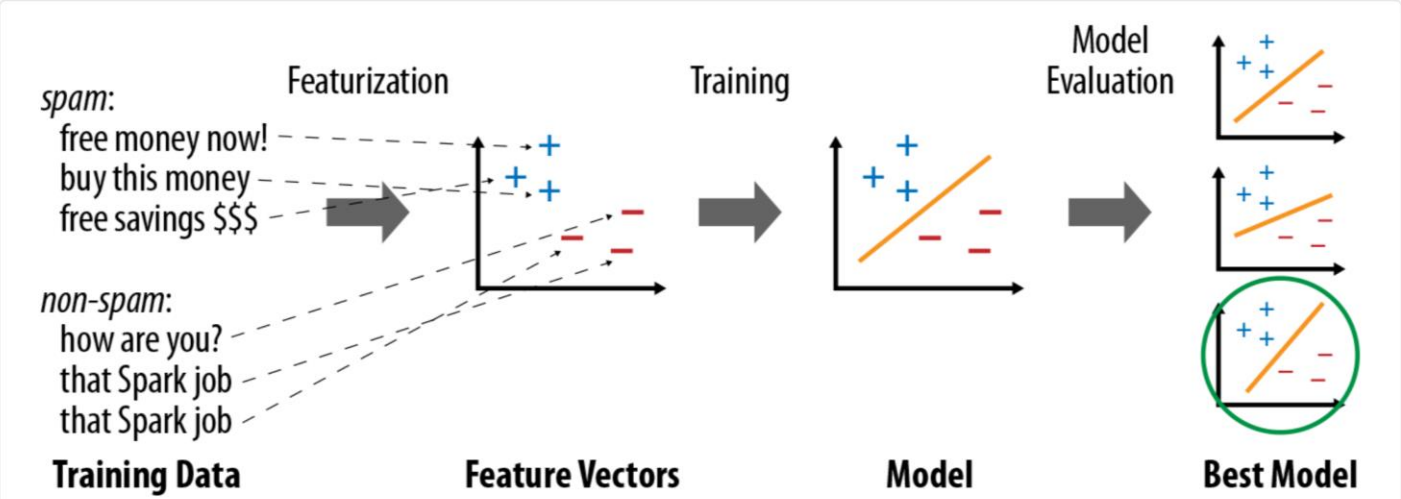
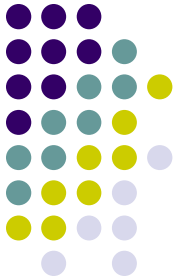


Figure 11-1. Typical steps in a machine learning pipeline

Turning a document into a vector



- In our spam-detector example pipeline, the featurization step was to turn a document into a vector
  - First, turn the document into an array of words
  - Then hash each word to an index and increment it (think of it like a counting Bloom filter)

- Document: “Now is the time”
- Tokenize: (“Now”, “is”, “the”, “time”)
- Replace by hash: (3, 0, 4, 3)
- Frequency vector: