11-676 Big Data Analytics
Fall 2015 Syllabus

Instructors	Ravi Starzl, GHC 6701	E-mails	rstarzl@cs.cmu.edu
Office Hours	By appointment	TAs:	Guan Wang guanw@andrew.cmu.edu Xiao Han xiaohan@andrew.cmu.edu

Required Texts:

Recommended Texts:

Other material will be provided as needed during the course.

Description:
This course will teach you the advanced skills required to create and deploy custom Big Data Analytics solutions. You will deploy MapReduce and Spark based solutions that make use of sophisticated machine learning algorithms as well as data sets that are unstructured and large. Throughout the course, you will work with a real problem presently being faced by companies in the Big Data domain. The course can be considered a kind of simulation, where you take on the role of aspiring data scientist, and I take on the role of project manager. You will engineer solutions to the problem presented, then iteratively improve them. There will be a round of team based competitive engineering for each problem before moving to the next one.

Pre-requisites:
Students considering this course should have completed the Introduction to Big Data Systems and Analytics course or alternatively have prior big data systems experience (at least familiar with the installation and configuration of Hadoop in a single-node distribution configuration such as Hortonworks Sandbox). Students should be able to implement MapReduce programs in Java. Experience with Pig and Hive is recommended. Students should also be comfortable with using Unix command line interfaces. This course will use Eclipse for development.

Evaluation:
Grading will be based on the following criteria:

Assignments: 50%
There will be an assignment every week, as well as random in-class exercises that will count as part of the assignment grade. Due dates will be provided at the time the assignment is given.

Team-Based Competitive Engineering Rounds: 10%
There are three general problem focus periods throughout the course, each one approximately four weeks long. Near the end of each period, teams will be formed to competitively engineer a

superior solution that integrates the best ideas and techniques of each team member. Details of team formation and size will be given at the first competitive event.

Mid-Term and Final Projects: 40% (20% each)
These are short, high intensity, end-to-end one week long projects. Students will be given a problem statement and data, then expected to make a presentation on how they have addressed the information need one week later.

Class Information:

- Class meets Tuesday and Thursday from 9:00am to 10:20am in SH 219.
- Tuesday classes will contain a review of important concepts related to the material we will cover that week. Thursday classes will be an open Q&A session with the instructor and TAs will be available for one-on-one assistance.
- Attendance is required on Tuesday, highly recommended Thursday. Every two unexcused absences will lower your grade one letter.
- There is no substitution for hands-on learning with algorithms and coding. For this reason, the majority of material you will be expected to learn will be covered in depth with the readings, examples, and tutorials you will complete outside lecture. Please complete all readings and exercises on time.
- Expect to spend 8-12 hours per week on the work for this course, depending whether your particular skillset and background overlaps significantly with the assigned work of a week.
- Due dates for assignments will be provided at the time the assignment is given. Late assignments will be penalized one grade for every two days it is late. All assignments are required, regardless of how late they are submitted. If you feel that the grade given on an assignment should be reconsidered, please submit the re-grade request in writing by email to the instructor, with a brief description of why you would like the assignment reviewed.

Academic Integrity
Collaboration is expected and encouraged among students – feel free to share notes and hold study groups; however attribution must be given in the form of citation, quotation, or co-authorship on documents.

IF YOU PARTICIPATE IN A STUDY GROUP FOR ANY ASSIGNMENT, YOU MUST LIST ALL MEMBERS OF THE STUDY GROUP ON YOUR ASSIGNMENT.

Failure to list participants of a study group on assignments, or disclose all the participants on a collaborative work, is not acceptable. These violations will result in the elimination of credit for the entire assignment on the first offense, and in the assignment of a failing grade for the course on the second offense
.
Violations of academic integrity are very serious and can result in heavy penalties up to suspension or expulsion. Make sure you review and understand the information at:
http://www.cmu.edu/policies/documents/Cheating.html and
http://www.cmu.edu/policies/documents/GradDisc.html.

Sharing code is explicitly forbidden

Writing is a crucial skill for both academic and commercial success. Therefore we ask you to do your best possible writing both in your presentations and code comments. Please use coherent arguments, good grammar, and correct spelling in your comments and solutions. If you need any assistance in editing or proofing, please make use of the resources at http://www.cmu.edu/acadev/resources/writing.html or see the instructors for help.

Course Schedule:

Week	***Topic***
1	Enumeration of the Information Need and Decision Trees in MapReduce
2	Ensemble Methods – Bagging and Boosting
3	Random Forest and Random Subspace in MapReduce
4	Data Preparation Fundamentals
5	Latent Features and Some Heuristics for Exposing Them
6	NoSQL Data Management and Integration
7	Leveraging NoSQL Graph Oriented Data Stores
8	Analytic Project Planning and Management
9	Scala Fundamentals
10	Scala and Spark: ML
11	Scala and Spark: Feature Extraction and Transformation
12	Scala and Spark: Feature Reduction
13	Focus Sessions: Effectively Presenting and Interpreting your Findings
14	Scala and Spark: Clustering

Assignments:

Week	***Topic***
1	Problem Part 1: Detect Similarity in Documents Enumerate and implement basic comparison strategy
2	Implement decision-tree MR method
3	Implement Random-Forest or Random-Subspace Map-Reduce method
4	(Continued) Implement Random-Forest or Random-Subspace Map-Reduce method
5	Problem Part 2: Improve Scale of Comparisons Optimize implementations and generate improved feature sets
6	
7	Move bigger data set and analytic pipeline sources into NoSQL
8	Implement one alternative approach to measuring similarity
9	MidTerm Project
10	Problem Part 3: Improve Speed of Comparisons Move strategy/implementation to Scala and compare to Hadoop implementation
11	Scala: Explore effectiveness of alternative learning methods

12	Final Project
13	
14	Final long-term problem project presentation