CS 6240: Project Report Template

We provide this template for two reasons: (1) It helps you learn a skill—presentation of results—using a format that tends to work well and gives you structure. This also minimizes point loss due to overlooked requirements. (2) It helps us grade reports more efficiently, maximizing the value of feedback we can give you with our limited TA/instructor resources.

For these reasons, it is important that you precisely follow the format, no matter the amount of progress you made on the different challenges. Not following these requirements can result in point deduction.

Make sure you include the “[Task Name]” section for **every major task** of your project, no matter how much progress you made on it. For instance, for a 3-person team, there must be three such sections (unless you work on major tasks that count for multiple tasks).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Team Members

Sai Divya Sangeetha Bhagavatula  
Maanasa Kaza  
Prajakta Rodrigues  
Anugraha Venugopal

Github Classroom Repo: <https://github.com/2020-F-CS6240/project-group3>

State all members of the team. **Also include a link to your Github Classroom repository for the project. (4 points)**

# Project Overview

Start with a summary of your project topic and analysis goals. This could be a single short paragraph. Then present a summary of the highlights of your project results or achievements, again just one or maybe two relatively short paragraphs.

Think of the overview as follows: You are applying for a job and your future employer is only reading this section. How can you tell them in about half a page why they should be interested in you and your project? How can you get them interested to keep reading? (Be confident in your work, but do not oversell it!)

# Input Data

Describe the data you are working with. Include a line of input, if feasible. (Do not include binary data or lines that are too long.)

The data has been produced using Monte Carlo simulations. The first 21 features (columns 2-22) are kinematic properties measured by the particle detectors in the accelerator. The last seven features are functions of the first 21 features; these are high-level features derived by physicists to help discriminate between the two classes. There is an interest in using deep learning methods to obviate the need for physicists to manually develop such features. Benchmark results using Bayesian Decision Trees from a standard physics package and 5-layer neural networks are presented in the original paper. The last 500,000 examples are used as a test set.

The first column is the class label (1 for signal, 0 for background), followed by the 28 features (21 low-level features then 7 high-level features): lepton pT, lepton eta, lepton phi, missing energy magnitude, missing energy phi, jet 1 pt, jet 1 eta, jet 1 phi, jet 1 b-tag, jet 2 pt, jet 2 eta, jet 2 phi, jet 2 b-tag, jet 3 pt, jet 3 eta, jet 3 phi, jet 3 b-tag, jet 4 pt, jet 4 eta, jet 4 phi, jet 4 b-tag, m\_jj, m\_jjj, m\_lv, m\_jlv, m\_bb, m\_wbb, m\_wwbb. For more detailed information about each feature see the original paper.

# [Task Name] (from Project assignment document!) Classification and prediction ensembles using your own local data mining algorithm implementation

## Overview

Summarize in a single short paragraph the goal of the task. The major project task here must match one of those listed in the initial project assignment (unless exception was granted by the instructor).

## Pseudo-Code

Show the pseudo-code for the algorithm you developed. For MapReduce, this should be much more concise and compact than your source code. For Spark, you can copy-and-paste from the actual source code. Make sure your code has comments explaining the main idea and computation aspects. Only provide pseudo-code for the main project tasks.

**Include a link to the Github repository. (4 points)**

## Algorithm and Program Analysis

Recall that any reasonable solution gets up to 90% of the score. A reasonable solution is a program that gets the job done but does not show much effort in terms of using knowledge from this course to improve or analyze performance. For the full 100%, discuss what you did to understand if your program uses the cluster efficiently and/or how you attempted to improve performance. This could be an analysis (e.g., to determine partitioning granularity for 1-Bucket or for matrix-product block partitioning) or experiments and measurements (e.g., varying a parameter that affects performance or memory use).

## Experiments

Present results that show how your program scales. Discuss the AWS cluster size and machine type used. Explain on which dataset you ran the experiments and what the program parameter settings were (if applicable). If you varied one parameter and kept others constant, explain this clearly.

**Include links to log and output files. (4 points)**

### Speedup

Show a graph or a table where you report running time of your program for at least two cluster sizes, e.g., 5 and 10 worker machines. Discuss briefly if you believe the speedup from small to large cluster is a good achievement for your problem. If speedup is low, discuss possible reasons.

### Scalability

If possible, change the size of your input and report how this affects running time. (This is simpler than scaleup. You do not need to achieve the same running time on different cluster sizes. Just report how time changes on a given cluster when you change input size.)

### Result Sample

Show a few lines from the output of the program. Use good judgement to make sure that the reader gets a good idea how the output looks like.

# [If you have more than 1 main project tasks, add the corresponding sections using the same format as above.]

# [Task Name] Classification and prediction in Spark MLlib

## Overview

Summarize in a single short paragraph the goal of the task. The major project task here must match one of those listed in the initial project assignment (unless exception was granted by the instructor).

## Pseudo-Code

Show the pseudo-code for the algorithm you developed. For MapReduce, this should be much more concise and compact than your source code. For Spark, you can copy-and-paste from the actual source code. Make sure your code has comments explaining the main idea and computation aspects. Only provide pseudo-code for the main project tasks.

**Include a link to the Github repository. (4 points)**

## Algorithm and Program Analysis

Recall that any reasonable solution gets up to 90% of the score. A reasonable solution is a program that gets the job done but does not show much effort in terms of using knowledge from this course to improve or analyze performance. For the full 100%, discuss what you did to understand if your program uses the cluster efficiently and/or how you attempted to improve performance. This could be an analysis (e.g., to determine partitioning granularity for 1-Bucket or for matrix-product block partitioning) or experiments and measurements (e.g., varying a parameter that affects performance or memory use).

## Experiments

Present results that show how your program scales. Discuss the AWS cluster size and machine type used. Explain on which dataset you ran the experiments and what the program parameter settings were (if applicable). If you varied one parameter and kept others constant, explain this clearly.

**Include links to log and output files. (4 points)**

### Speedup

Show a graph or a table where you report running time of your program for at least two cluster sizes, e.g., 5 and 10 worker machines. Discuss briefly if you believe the speedup from small to large cluster is a good achievement for your problem. If speedup is low, discuss possible reasons.

### Scalability

If possible, change the size of your input and report how this affects running time. (This is simpler than scaleup. You do not need to achieve the same running time on different cluster sizes. Just report how time changes on a given cluster when you change input size.)

### Result Sample

Show a few lines from the output of the program. Use good judgement to make sure that the reader gets a good idea how the output looks like.

# Conclusions

In one or two paragraphs, summarize your main results and contributions; maybe also point out possible extensions or alternatives that might be worth exploring in future work. This section makes the last impression on the reader. Think about what you would like that person to remember about your project.