CSCI 6900: Practice Midterm

February 24, 2015

Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page.

Name: _____

Multiple Choice (25 pts)

Clearly circle the choice you think best addresses the prompt.

1. If I wanted to model a power-law graph as accurately as possible, I would start with a/an
    A. undirected graph
    B. Watts-Strogatz graph
    C. Erdős-Rényi graph
    D. Barabasi-Albert graph

2. MapReduce is well-suited for all of the following procedures EXCEPT
    A. breadth-first search
    B. word counting
    C. batch processing
    D. MapReduce is well-suited for any procedure

3. Bayes' Classifier is considered the "optimal" classifier, as it is mathematically guaranteed to induce the lowest statistical risk of misclassification. However, despite its proven optimality, it is never used. Why?
    A. Different classifiers are optimal under different conditions.
    B. Ensemble methods are more optimal than lone classifiers.
    C. Bayes' Classifier is intractable to train.
    D. Naïve Bayes is much faster.

4. $P(A) + P(\neg A) =?$
    A. 1
    B. 0.5
    C. 0
    D. Not enough information

5. Spectral clustering derives its name from
    A. the spectrum of insights it provides
    B. its use of the spectral properties of the data
    C. the fact that it is effective in across a spectrum of applications
    D. its Latin roots meaning *spectre*, explaining its seemingly-magical abilities

Short Answer (25 pts)

Write a couple sentences addressing each question.

1. One paper presented involved the construction of a knowledge vault, a learned repository of general knowledge through supervised and unsupervised term association. What was the primary abstraction used to represent and quantify "knowledge"?

2. Explain "data locality" within the context of handling large quantities of data.

3. What is the primary drawback of operating on graph structures in MapReduce? Assuming other frameworks aren't available, are there any strategies we can use within MapReduce to mitigate this drawback?

4. Describe the difference between MLE and MAP. Name an example of each from your programming assignments.

5. Bayes' Theorem can be derived from the Axioms of Probability related to conditional probability and is as follows: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. Identify the prior, likelihood, and posterior in the equation.

Programming Q1 (25 pts)

Word counting is an ubiquitous tool for text analysis: the bag-of-words model relies on frequency counts of terms throughout a corpus of documents. In some cases, we are interested not in the global counts, but in the most frequent k terms.

In this problem, you will write code to implement the `Mapper` of a distributed Hadoop program which identifies the k most frequent terms in the corpus. The mapper will receive a single document as input. The mapper will provide as output the frequency counts of the words in the document. The integer k indicating the number of most-frequent words we want to identify can be accessed through the `Configuration` object, specifically by invoking `conf.getInt(TOP_WORDS)`, where `TOP_WORDS` is a pre-defined global constant.

Some other points to keep in mind:

- Your code MUST be correct Java.
- At a minimum, you MUST implement the `map` method. You are free to override the `setup` and/or `cleanup` methods if you wish.
- Don't worry about declaring any `Exception`s, but DO make sure method return types are correct.
- You do NOT need to worry about adding `import` statements.
- Each `*Writable` class has a `.get()` method that returns the primitive version of its data type (e.g., `IntWritable.get()` returns an `int`, etc). You MUST invoke this method on every `Writable` before manipulating or otherwise using it.
- ONLY objects of type `Writable` can be written through the Hadoop `Context`.
- Primitive wrapper objects (e.g. `Double`, `Integer`, etc) are annoying and can be ignored for this problem. That is, any method which would return a wrapper object can be assumed to return the corresponding primitive type, if that makes the coding easier. Any method which otherwise requires a wrapper as a parameter (e.g. creating a new `Writable`) can be assumed to take the corresponding primitive as a parameter, instead of the wrapper.

Write your mapper on the following page; the class declaration has been provided.

```
public class TopKMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
```

Programming Q2 (25 pts)

In this problem, you will write the corresponding `Reducer` for the previous word counting algorithm. Regardless of how you implemented the previous question, assume the input to the reducer is a `<key, value>` pair of the format `<word, [count1, count2, ..., countn]>`, where the key is the word and the value is the list of frequency counts of that word across all documents. **Also assume there is only 1 reducer, and that it is invoked once for each word** (this is critical to correctly implementing the reducer). The output should be the k most frequent words and their associated frequency counts (you can extract this value k from the `Configuration` object as in the previous question).

Some other points to keep in mind:

- Your code MUST be correct Java.
- At a minimum, you MUST implement the `reduce` method. You are free to override the `setup` and/or `cleanup` methods if you wish.
- Don't worry about declaring any `Exception`s, but DO make sure method return types are correct.
- You do NOT need to worry about adding `import` statements.
- Each `*Writable` class has a `.get()` method that returns the primitive version of its data type (e.g., `IntWritable.get()` returns an `int`, etc). You MUST invoke this method on every `Writable` before manipulating or otherwise using it.
- ONLY objects of type `Writable` can be written through the Hadoop `Context`.
- Primitive wrapper objects (e.g. `Double`, `Integer`, etc) are annoying and can be ignored for this problem. That is, any method which would return a wrapper object can be assumed to return the corresponding primitive type, if that makes the coding easier. Any method which otherwise requires a wrapper as a parameter (e.g. creating a new `Writable`) can be assumed to take the corresponding primitive as a parameter, instead of the wrapper.

Write your reducer on the following page; the class declaration has been provided.

```
public class TopKReducer extends Reducer<Text, IntWritable, Text, IntWritable>
{
```