CSCI 585 final exam: 6/28/22
Duration: 90+30 minutes (to write/type, submit)

Please read the following carefully, before starting the test:

• the exam is **open** books/notes/devices - feel free to look up whatever you want, from wherever!

• from Q1 through Q10 (worth 5 points each), you can **choose any 7**, for a total of 35; if you want you can do **8, 9, or even, all 10** - we will ADD up ALL your scores (even partial ones) from **all** the questions you answer, then **CAP it at 35**, meaning, a total > 35 will be set to 35 - omg, sweet!! This was exactly like how it was, for the midterm.

• there are no 'trick' questions, or ones with long calculations or formulae, and there's certainly nothing to memorize [it's all OPEN, duh :)] It doesn't mean the questions are trivial! Please do answer carefully: in other words, **answer WHAT IS ASKED**, otherwise you won't get points (eg. if a question is -ABOUT- 2PL, don't DESCRIBE/DEFINE 2PL!) - it's the quality (of your answer) that counts, not quantity (verbosity)...

• **please do NOT cheat** - this means NOT communicating with anyone via any device/medium/channel - you will get a 0, and be reported to SJACS, if you are found to have cheated; ANY attempt to get help from others in any form is a VIOLATION, as per https://policy.usc.edu/scampus-part-b/, sections 11.11 through 11.14 [read it, if you are not familiar with it]

• when the time is up (90 minutes), stop your work, then spend the rest of time (30 minutes) on submission [students with DSP accommodations - your exam duration will be as per DSP determination] - **submitting past the deadline comes with a penalty**, because it is not fair to others if you go over when they don't; **note that you need to submit each answer separately** (not all of them as a single PDF)

• you can write your answers on paper, take pics to submit; or you can type them up, take a screenshot to submit. You can do either, for each question - meaning, you can write out some, type up others.

Fun fact: each question has 'data' in it (it's a DB exam, duh!!).

Good luck! Hope you enjoy answering the questions, hope you find them to be easy.

Q1 [3+2 = 5 points].

Create a valid JSON file (document) that contains KML equivalents of two points inside a folder, two points connected by a line, a convex hull polygon (all these in a single JSON file). You can use your data from HW3 if you like, to make this easier.

A. There are MANY ways to specify the data in JSON format, eg.

```
{
 "myData": [
   {
     "libraries": [
       {
         "lat": ...
         "lon": ...
       },
       {
         "lat": ...
         "lon": ...
       },
     ]
   },
   {
    "ptA": {...},
    "ptB": {...}
   },
   "hullCoords":[
      {...},
      {...},
      ...
   ]
 ]
}
```

The answer does need to contain a key for folder name (for the two points inside it).

Take two pieces of triple-store data, express it as a valid JSON document. The two pieces don't need to be related (connected), and it's ok if they are related.

A. We can make an array with the two triples inside it as objects, eg.

```
{
  "triplesData": [
    {
      "subject": "LosAngeles",
      "predicate": "CityIn",
      "object": "California"
    },
    {
      "subject": "USC",
      "predicate": "schoolLocation",
      "object": "LosAngeles"
    }
  ]
}
```

Q2 [3+2 = 5 points].

Why do we still hold on to SQL, even for analyzing NoSQL data?

A. Because it is the best (most expressive, powerful) language for querying data, compared to declarative alternatives such as C++.

Name three examples of SQL-like syntax we looked at, for data analysis in a non-relational context (ie. in Part 2 of the course).

A. HiveSQL, CQL, Samza query language, HQL...

Q3 [2+3 = 5 points].

In the classic MapReduce (MR) algorithm, what produces the speedup in processing data, compared to a non-MR setup?

A. The speedup comes from processing horizontal fragment data in parallel (in the 'map' step).

What is preferable to 'raw' MR, and why? Explain, using a few sentences.

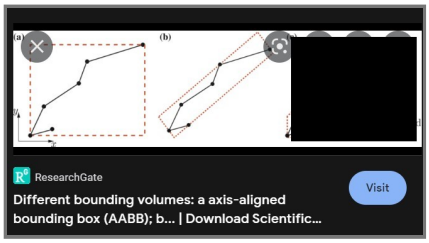
A. A higher-level language such as Pig or HiveSQL - these help us express our query more naturally, ie not requiring us to think in terms of mapping and reducing steps (the associated compiler comes up with those automatically, by analyzing our query).

Q4 [2+3 = 5 points].

MBR-based indexing is the most common way to index spatial data, but it is inefficient in a sense. How? And, what would be the fix? Explain using a diagram.

A. Because MBRs are always parallel to X and Y (they are 'AABBs'), they are not well-suited for features that are angled, eg. a road with a unit slope.

The fix would be to use an oriented bounding box (OBB) instead.



How would the use of circles work, instead of rectangles, for indexing spatial data? Illustrate. Why would this not be better than using rectangles (MBRs)?

A. Circles could work as well, in principle, because they too can form hierarchies (smaller circles inside bigger ones), that can be used for indexing (just like R-trees). But we would need more of them compared to rectangles, so as not to miss (exclude) any feature - that is the drawback.

Q5 [3+1+1 = 5 points].

In what sense are boosting and bagging, flips/inverses/opposites of each other? Explain clearly (the question is NOT asking for their definitions or descriptions!).

A. In boosting, we use multiple DM algorithms on the same data; in bagging, we use multiple (fragmented) pieces of the data on the same algorithm (eg. RandomForest).

In RandomForest, how do we know what the optimal number of trees are?

A. By using int(sqrt(k)), where k is the number of items in the training dataset, for each tree - that means the number of trees is also ~ int(sqrt(k)).

In clustering, how do we know what the ideal cluster count is?

A. We would use the 'elbow' method to find out, eg. as mentioned in https://www.edureka.co/blog/k-means-clustering/ [linked in https://bytes.usc.edu/cs585/m22_datadatadata/lectures/DM/slides.html#(9)].

Q6 [3+2 = 5 points].

Explain the role of ML in a data pipeline, in terms of querying data. In other words, data has always been 'queryable' - how does ML improve on this, ie what new capabilities does it provide?

A. While we have always been able to query data, we have never been able to use it to make predictions on new data, based on learning the patterns in it - ML provides that. Via ML, existing data becomes usable in a new way - to handle new (unseen thus far) data as well.

What makes ML, the most flexible data mining algorithm?

A. The activation function (usually sigmoid()) in each neuron - by having multiple (deep) layers, and multiple neurons in each layer, the small 'tweaks' (non-linear mappings) from each neuron are able to collectively learn the pattern in ANY signal (data), however complex. A neural network is a 'universal function approximator' in other words.

Q7 [2+3 = 5 points].

'Big Data' will only get bigger... What can we do, to cope? This is a practical question - you can answer in many ways. Think about storage, and computing - how can both be improved (what new technologies could help)?

For computing, there could be spintronics, optical computing etc. that boost computing speeds; there could be holographic, or DNA-based technologies - for many more orders of storage. Also, network speeds can be improved, for higher throughput.

List and discuss (a line each) three non-tabular data formats you used, in your homeworks.

A. KML, JSON, hd5 (NN format). BECAUSE HW5 was not due before the exam - for this question, we can accept just two (eg KML, JSON) or even just ONE answer!

Q8 [1+2+2 = 5 points].

Traditional data warehouse design is based on the classic 'star' schema. Briefly explain what this means.

A. From the slides - fact table in the center, 'star' points - dimension tables - that surround.

Why (ie for what use case) would we expand the star into a snowflake?

A. When we need to drill in or zoom out, ie. analyze data at a fine-grained or aggregrate levels along attribute hierarchies.

Why (again, what use case) would we 'blunt' the star, ie. totally get rid of it?

A. When we require speed - by having heavily denormalized, redundant data all in the fact table, we avoid (usually expensive) joins altogether.

Q9 [2+3 = 5 points].

What is the rationale behind 'iterative refinement', ie. why is it a good technique (whether it's for software design, or product design, or data analysis, etc.)?

A. Because we can always 'start somewhere' - a rough design, initial draft, incorrect parameters... and REFINE, ie. modify it incrementally to create increasingly better versions. In other words, the right answer, ideal model... can be arrived at, over time - by starting with a 'poor' version.

In DML/ML, which three algorithms make use of iterative refinement?

A. NN, k-means (for cluster centroids), E-M (iterative refinement of model and also latent params), etc.

Q10 [3+2 = 5 points].

What does the sigmoid curve do, to data that is input to it?

The sigmoid tweaks it, ie. produces a non-linear response ('S' curve shaped - hence the name 'sigmoid'). Also, it maps large ranges of input (-inf to inf) to a finite 'y' value (eg. -0.5 to 0.5, 0 to 1, etc).

In DM/ML, sigmoid shows up in two different contexts - what are they? Explain, in a few sentences.

From the notes - sigmoid is used in logistic regression, to classify input into 'n' classes (usually two); it is also used as the activation function in NNs.
