Big Data Analytics

Stony Brook University  
CSE545 - Spring 2022

**Assignment 3**

**Assigned: 4/9/2022;   Due: 4/24/2022 11:59pm**

[**Overview**](https://docs.google.com/document/u/3/d/e/2PACX-1vTT8lW5ClrbjQnOJxHnS_uGJI5nzd6LE2WxAq_W2FpjBRa7Wo_w882Dtk9cuHIgPrsEtBSTLWVYU08L/pub#h.k7js7on60mn7)

[**Part I. Hello World from Cluster (10 points)**](https://docs.google.com/document/u/3/d/e/2PACX-1vTT8lW5ClrbjQnOJxHnS_uGJI5nzd6LE2WxAq_W2FpjBRa7Wo_w882Dtk9cuHIgPrsEtBSTLWVYU08L/pub#h.pvn219uxc1h4)

[**Part II. Restaurant Recommendation System (45 Points)**](https://docs.google.com/document/u/3/d/e/2PACX-1vTT8lW5ClrbjQnOJxHnS_uGJI5nzd6LE2WxAq_W2FpjBRa7Wo_w882Dtk9cuHIgPrsEtBSTLWVYU08L/pub#h.mfqja726bk8j)

[**Part III. Hospital-Reviews Hypothesis Testing (45 Points)**](https://docs.google.com/document/u/3/d/e/2PACX-1vTT8lW5ClrbjQnOJxHnS_uGJI5nzd6LE2WxAq_W2FpjBRa7Wo_w882Dtk9cuHIgPrsEtBSTLWVYU08L/pub#h.izblg9espjua)

[**Submission**](https://docs.google.com/document/u/3/d/e/2PACX-1vTT8lW5ClrbjQnOJxHnS_uGJI5nzd6LE2WxAq_W2FpjBRa7Wo_w882Dtk9cuHIgPrsEtBSTLWVYU08L/pub#h.vswnf3261km)

**Overview**

|  |
| --- |
| **Objectives:**   * Gain experience with a live hadoop-style (hdfs,  spark) cluster. * Implement a basic collaborative filtering recommendation system. * Implement hypothesis testing with multi-test correction at scale. * Gain experience navigating a cloud console to spin up a cluster. * Work with moderately large data. * Gain experience problem solving non-theoretical, practical issue of big data.   **Requirements.** You must use Python version 3.6 or later and PySpark 3.0 or later **over a GCloud cluster as specified in Part I below**.  You must also keep the data in Spark RDDs – you may not use Spark DataFrames (mastering spark RDDs makes using dataframes efficiently easy; people who learn dataframes without first learning RDDs often implement very inefficient approaches).  **Python Libraries.**  Acceptable data science, machine learning, or statistics libraries are listed below. Other data science, machine learning, or statistics related libraries are prohibited unless listed below --  **ask if unsure.** The intention is for you to implement the algorithms we have gone over and problem solve in order to best understand the concepts of this course and their practical application.  You may not use any pre-existing implementations of such algorithms even if they are provided in the following libraries.  Additional approved libraries that are not in the template will be listed here (if any):  import random  import numpy as np  import math  import hashlib  import csv |
| **Copying code from other students, online or other resources is prohibited** and will result in at least a zero on the assignment and report to graduate program director with possibility for more consequences. Please see syllabus for additional policies.  A word to the wise: As is tradition in CSE545 at SBU, parts of this assignment are completely novel, never given before. |

**Part I. Hello World from Cluster (10 points)**

|  |  |
| --- | --- |
| Your objective is to spinup a cluster with multiple worker nodes.  **Output:** Place checkpoint output into: a3\_p1\_<lastname>\_<id>.pdf  (Please do not submit any format except a pdf; use print to pdf in your favorite document editor).  **Spin up a cluster and take a screenshot.**  Follow this tutorial:   * Use either your @cs.stonybrook.edu email or your personal gmail (note: we get a list of all emails redeeming credit; one email per student). * [Steps to getting GCloud Access](https://www.google.com/url?q=https://docs.google.com/document/d/1uNQ5J3L-pi2CEiVhq4-eErnmdhY_Bh2wZHD8xgqmg6U/edit?usp%3Dsharing&sa=D&source=editors&ust=1654368309662918&usg=AOvVaw0MiOX5PGlUR8CxE6koaKEn) You will be asked to provide your school email address and name. An email will be sent to you to confirm these details before a coupon is sent to you. * [Video: Steps to create clusters in Google Cloud Platform](https://www.google.com/url?q=https://drive.google.com/file/d/1KzvXPL66_D3zCvln8O8NqSy5XgQpMUxP/view?usp%3Dsharing&sa=D&source=editors&ust=1654368309663357&usg=AOvVaw2BC5no02B5LtYIW0Y60Jf4)   Use the following configuration (You can use any "east" region and zone as long as the zone matches the region):  ***(...........................................................version 1: 4/08/2022……….……………………………………….)*** Table  Description automatically generated  ***(..............................................................update: 4/08/2022……….……………………………………………….)***  It’s useful to get Google SDK for your local machine: [https://cloud.google.com/sdk/docs/](https://www.google.com/url?q=https://cloud.google.com/sdk/docs/&sa=D&source=editors&ust=1654368309664378&usg=AOvVaw0hMFDJmeGOB7myGNu66MI_)  **Checkpoint 1.1**  Take a screenshot of console.cloud.google.com/dataproc/clusters to show your cluster “running”.  **Test the cluster.** ~~Set pyspark to use python3:~~~~add “export PYSPARK\_PYTHON=python3” to the bottom of ".bashrc":                 (you can edit with “nano .bashrc” or install your preferred editor)         run “source .bashrc”~~ Launch pyspark: “pyspark” and try a few things: sc.\_jsc.sc().getExecutorMemoryStatus().size() #returns the number of nodes  sc.parallelize([1, 2, 3, 4, 5]).take(2)  **To install a library (i.e. numpy):**  **#update the cluster now comes with all libraries you need**   |  | | --- | | **~~#first setup python3 for gcloud and pip:~~**  ~~sudo apt-get -y install python3 python-dev build-essential python3-pip~~  ~~sudo pip3 install --upgrade google-cloud sudo pip3 install --upgrade google-api-python-client sudo pip3 install --upgrade pytz~~  ~~sudo echo "export PYSPARK\_PYTHON=python3" | sudo tee -a /etc/profile.d/spark\_config.sh /etc/\*bashrc /usr/lib/spark/conf/spark-env.sh sudo echo "export PYTHONHASHSEED=0" | sudo tee -a /etc/profile.d/spark\_config.sh /etc/\*bashrc /usr/lib/spark/conf/spark-env.sh nano /etc/spark/conf/spark-defaults.conf # add to bottom: "spark.executorEnv.PYTHONHASHSEED=0"~~  **~~#now install libraries as such:~~**  ~~sudo pip3 install~~**~~numpy~~**  **~~#On each worker VM instance~~**~~(goto console.cloud.google.com/compute/instances):~~  ~~sudo apt-get -y install python3 python-dev build-essential python3-pip sudo pip3 install~~**~~numpy~~** |   (This is an updated version of: [https://stackoverflow.com/questions/45843960/how-to-run-python3-on-googles-dataproc-pyspark](https://www.google.com/url?q=https://stackoverflow.com/questions/45843960/how-to-run-python3-on-googles-dataproc-pyspark&sa=D&source=editors&ust=1654368309667646&usg=AOvVaw2S9Rc2VhLiU5Qgqox2HFA7))  **Note: once you create a cluster, it should work by default without package installation above.**  **Importing data into hdfs.** hadoop fs -put FILENAME FILENAME\_ON\_HDFS  **Tell hdfs to replicate a data file 4 times (increases read throughput):** hadoop fs -setrep -w 4 FILENAME\_ON\_HDFS  **Checkpoint 1.2**  Put any file into hdfs, and then run the following and paste the output into a file along with checkpoint 1.1 output:  hadoop fs -ls FILENAME\_ON\_HDFS(i.e. the goal is to show that you have put a file into your hdfs)  **Shutting down the cluster.** Do not leave the cluster up while you are not using it! It will costs you credits. If you’re not using it, stop the VM instances to save credits: [Console](https://www.google.com/url?q=https://console.cloud.google.com/dataproc/cluster&sa=D&source=editors&ust=1654368309669138&usg=AOvVaw07I52LWumUyVf3LFRIz3r6) -> Compute Engine -> VM Instances -> Select all of the Instances -> “Stop”. Once done with the assignment and you have saved your code to your own machine, then go to [https://console.cloud.google.com/dataproc/cluster](https://www.google.com/url?q=https://console.cloud.google.com/dataproc/cluster&sa=D&source=editors&ust=1654368309669325&usg=AOvVaw3FjaBgw5ioC2EJytdHI6BE) and delete the cluster to make sure you do not use up credits on it. **Tip:**If you plan to do a lot of development on the cluster, start with only 2 worker nodes on a subset of the data and then upgrade to test on 4. |

**Part II. Restaurant Recommendation System (45 Points)**

|  |
| --- |
| Your objective is to help 5 people from near Las Vegas and elsewhere to find a good place to eat. You will do this by creating a couple collaborative filtering-based recommendation systems over on Yelp data.  **Filename:**a3\_p2\_<lastname>\_<id>.py  **Data:**  [Trial review.json Data](https://www.google.com/url?q=https://www3.cs.stonybrook.edu/~has/CSE545/a3/trial_yelp_dataset_review.json&sa=D&source=editors&ust=1654368309671127&usg=AOvVaw2LhoiyEB1B5qpr8Nyfq1Rm) (just for getting familiar with format)  [Full review.json Data](https://www.google.com/url?q=http://www3.cs.stonybrook.edu/~cse545/review.json.gz&sa=D&source=editors&ust=1654368309671479&usg=AOvVaw0FwCwV1X8RnCQTRQTPpsvC)  *Tip:*You do not need to download the data to your personal machine. Send it directly to your cluster. From the name node of your cluster, run:  wget http://www3.cs.stonybrook.edu/~cse545/review.json.gz  zcat review.json.gz | hadoop fs -put - /data/review.json  **Input:** Your code should take one command line parameter for the review dataset location.          Example: spark-submit a3\_p2\_lastname\_id.py 'hdfs:/data/review.json'  **Output:** Place checkpoint output into: a3\_p2\_<lastname>\_<id>\_OUTPUT.txt  **Step 2.1: Create a utility matrix, represented in sparse format as an RDD.**   1. Reach each line into an RDD and get rid of everything except the user\_id, business\_id, stars (rating), and date. (hint use: .map(json.loads) to get a dictionary per record). Note in this context, "items" are restaurants which are identified by the business\_id field in the data. 2. Filter to only one rating per user per item by taking their most recent record. (hint: this is the last time you will need the "date" field) 3. From there, filter to items (i.e. business\_ids) associated with at least 30 distinct users. 4. From there, filter to users associated with at least 5 distinct items. 5. Extract the following *target users* along with their business\_id ratings into a broadcast variable named "target\_users".   PomQayG1WhMxeSl1zohAUA  uEvusDwoSymbJJ0auR3muQ  q6XnQNNOEgvZaeizUgHTSw  n00anwqzOaR52zgMRaZLVQ  qOdmye8UQdqloVNE059PkQ  ^ target\_users subject to change until 4/18 (**updated on 4/18**)  **Checkpoint 2.1**  Print the first 10 business\_ids and ratings for the *target users.*For each user, sort alphanumeric by business\_id (this can be done outside the RDD) and only include the first 10 of them.  **Step 2.2: Perform user-user collaborative filtering.**   1. Read the following step (2.2.2) and transform your RDD into a format(s) appropriate for finding all similar users to the given *target\_usees.*(tip: using business\_ids as a key makes it easy to find all those who rated a particular business; using another RDD with the user\_id as key makes it easy to find all other businesses rated by a given user). 2. For each of the target users, find up to 50 most similar neighbors (i.e. other users). Neighbors must:         (a) have at least two ratings for the same businesses as the target\_users         (b) have a positive, non-zero similarity with the target users Use the consine similarity of mean-centered ratings as the similarity metric. 3. Make predictions of how the target\_user would rate other resteraunts. For each target user, make predictions for all restaurants for which at least three neighbors have ratings.   **Checkpoint 2.2**  Print the first 10 business\_ids that received a predicted rating along with their predicted rating for the *target users.*For each user, sort alphanumeric by business\_id (this can be done outside the RDD) and only include the first 10.  **Step 2.3: Perform item-item collaborative filtering.**   1. Transform your RDD (from 2.1) into a format(s) appropriate for finding all similar items to the given all of the items that *target\_usees* reviewed*.* 2. For each item in the entire dataset, find up to 50 most similar neighbors (i.e. other items). Neighbors must:         (a) have at least one rating from one of the *target users*(if not, do not consider the item as a potential neighbor).         (b) have a positive, non-zero similarity with the target item.         Use the consine similarity of mean-centered ratings as the similarity metric. 3. Make predictions of how the user would rate other resteraunts based on how the user rated similar restaurants. Only make predictions for resteraunts with at least 3 neighbors.   **Checkpoint 2.3**  Print the first 10 business\_ids that received a predicted rating along with their predicted rating for the *target users.*For each user, sort alphanumeric by business\_id (this can be done outside the RDD) and only include the first 10.  **\*\*Remember to save your code and delete/terminate your cluster when you're not using it.\*\*** |

**Part III. Hospital-Reviews Hypothesis Testing (45 Points)**

|  |
| --- |
| Restaurants and what we eat differ by and form characterizations of a community.  At the same time the percent of hospital beds used is a strong indicator of burden on a community's health system.  Your objective is to compute the association between 184 ingestion words (case insensitive) and the healthcare burden on a community as measured by mean bed usage percentage of hospitals. You will do this by testing the hypothesis that each of the 184 words is correlated with bed usage, correcting for multiple tests, as well as recording the effect size (Pearson correlation).  **Data:**  [review.json](https://www.google.com/url?q=http://www3.cs.stonybrook.edu/~cse545/review.json.gz&sa=D&source=editors&ust=1654368309681720&usg=AOvVaw0cIlxbcQws7sIXAWYk3O7N) – Contains restaurant reviews per business\_id (among other things not needed).  [business.json](https://www.google.com/url?q=http://www3.cs.stonybrook.edu/~cse545/business.json&sa=D&source=editors&ust=1654368309682223&usg=AOvVaw2INWfodv-CmWE32OAsT6Um) – Contains mapping of business\_id to zip (among other things not needed))  [test\_COVID-19\_Hospital\_Impact.csv](https://www.google.com/url?q=http://www3.cs.stonybrook.edu/~has/CSE545/a2/test_COVID-19_Hospital_Impact.csv&sa=D&source=editors&ust=1654368309682573&usg=AOvVaw1mjbw2-JTuwxTpK9aGK4g9) – Contains reports of beds used per hospital (same data as used in assignment 2).  [dictionary.csv](https://www.google.com/url?q=http://www3.cs.stonybrook.edu/~cse545/dictionary.csv&sa=D&source=editors&ust=1654368309683005&usg=AOvVaw2UhfGuK_KThCJ_dUjwxrrd) – list of 184 "ingestion words" from Linguistic Inquiry and Word Count (Pennebaker et al., 2015).  **Input:** Your code should take one command line parameter for the review dataset location.          Example: spark-submit a3\_p2\_lastname\_id.py 'hdfs:/data/review.json' 'hdfs:/data/business.json' 'hdfs:/data/test\_COVID-19\_Hospital\_Impact.csv' 'dictionary.csv'  **Output:** Place checkpoint output into: a3\_p3\_<lastname>\_<id>\_OUTPUT.txt  **Filename:** a3\_p3\_<lastname>\_<id>.py  **Step 3.1: Aggregate outcome data to zip codes.**   1. Filter to hospital records with at least 30 for total\_beds\_7\_day\_avg (for those with smaller total beds, percentages will not be reliable). 2. Find the mean bed usage percentage per hospital. To do this, first calculate the bed\_usage\_percent for each record: bed\_usage\_pct = inpatient\_beds\_used\_7\_day\_avg / total\_beds\_7\_day\_avg and then aggregate it to the hospital level as the mean of all bed\_usage\_pct s. mean\_bed\_usage\_pct = mean([bed\_usage\_pct1,bed\_usage\_pct2, …]) 3. Next, aggregate the mean\_bed\_usage\_pct per zip code by taking the mean across all hospitals within that zip code.   **Checkpoint 3.1**  Print the mean\_bed\_usage\_pct for zip codes: 89109, 89118, 15237, 44122, 44106  **Step 3.2: Aggregate word usage per zip code.**   1. Add zip code to reviews by looking up the zip code of the business\_id in business.json. 2. Filter to reviews with at least 256 characters. This is to make sure enough language has been observed to bother considering the review. (This can be done in conjunction with 3.2.1 or 3.2.3). 3. Aggregate the percentage of reviews mentioning each of the 184 dictionary words per zip code. You should not consider how many times a word was mentioned in a particular review but just whether it was mentioned at all. usage\_score = sum([1 if word mentioned else 0 for all reviews in zip]) / (number of reviews in zip) *Note: the dictionary allows for prefix matching. Words ending in "\*" should be matched to any remaining characters. For example*'alcohol\*' *matches*'alcohol', 'alcohols'*, and*'alcoholic' *among others*.   **Checkpoint 3.2**  Print the 5 most frequent words, along with their usage\_score for zip codes: 89109, 89118, 15237, 44122, 44106  **Step 3.3: Calculate correlations between word usage and mean\_bed\_usage\_pct.**   1. For each word in the dictionary, correlate its usage with mean\_bed\_usage\_pct. 2. Calculate the p-value and Bonferonni corrected p-value for each word.   **Checkpoint 3.3**  Print the top 20 most positively correlated words along with their p-value and corrected p-value.  Print the top 20 most negatively correlated words along with their p-value and corrected p-value.  **\*\*Remember to save your code and stop or terminate your cluster when you're not using it.\*\*** |

**Submission**

|  |
| --- |
| Submit the following 5 files containing the output of your code as well as your code itself. Please use blackboard to submit two files each with your lastname and student id:   1. a3\_**p1**\_lastname\_id\_OUTPUT**.pdf** 2. a3\_**p2**\_lastname\_id**.py** 3. a3\_**p2**\_lastname\_id\_OUTPUT**.txt** 4. a3\_**p3**\_lastname\_id**.py** 5. a3\_**p3**\_lastname\_id\_OUTPUT**.txt**   **Please do not upload a zip file. Double-check that your files are there and correct after uploading and make sure to submit.** Uploading files that are zips or any other type than python code or txt files will result in the submission being considered invalid. Partially uploaded files or non-submitted files will count as unsubmitted.  **Runtimes (added 4/14):**Your code for each part should run in under 10 minutes on the test data and within cluster-mode on the specified gcloud cluster.    **Questions**: Please post questions to the course [Piazza](https://www.google.com/url?q=https://piazza.com/stonybrook/spring2022/cse545&sa=D&source=editors&ust=1654368309691061&usg=AOvVaw1GNRccck3RvYUSVzqUBA_X) page.  See Piazza posts tagged "assignment3".  As noted in syllabus and past assignments: questions sent within 48 hours of the assignment deadline are not guaranteed a response. |