Organización de Datos 75.06. Primer Cuatrimestre de 2017. Examen parcial, primera oportunidad:

Importante: Antes de empezar complete nombre y padrón en el recuadro. Lea bien todo el enunciado antes de empezar. Para aprobar se requiere un mínimo de 60 puntos (60 puntos = 4) con al menos 20 puntos entre los ejercicios 1 y 2. Este enunciado debe ser entregado junto con el parcial si quiere una copia del mismo puede bajarla del grupo de la materia. En el ejercicio 3 elija 2 de los 4 ejercicios y resuelva única y exclusivamente 2 ejercicios. Si tiene dudas o consulta levante la mano, esta prohibido hablar desde el lugar, fumar o cualquier actividad que pueda molestar a los demás. El criterio de corrección de este examen esta disponible en forma pública en el grupo de la materia.

"It's a trap!" (Admiral Ackbar, Return of the Jedi)

#	*1*	*2*	*3.1[ ]*	*3.2[ ]*	*4*	*5*	*6*	*7*	**Entrega Hojas:**
Corrección									Total:
Puntos	/15	/15	/10	/10	/15	/15	/10	/10	/100

Nombre:
Padrón:
Corregido por:

1) Se tiene información estadística de la temporada regular de todos los jugadores de la NBA en un RDD de tuplas con el siguiente formato: (id_jugador, nombre, promedio_puntos, promedio_asistencias, promedio_robos, promedio_bloqueos, promedio_rebotes, promedio_faltas). Un analista de la cadena ESPN está trabajando con un RDD que corresponde a la primera ronda de playoffs y que tiene el siguiente formato: (id_jugador, id_partido, timestamp, cantidad_puntos, cantidad_rebotes, cantidad_bloqueos, cantidad_robos, cantidad_asistencias, cantidad_faltas). En base a estos RDDs se quiere programar en PySpark un programa que genere un RDD con los nombres (sin duplicados) de los jugadores que lograron en algún partido de playoffs una cantidad de asistencias mayor a su promedio histórico. (****) (15 pts)

2) Un sitio de Ebooks tiene información sobre los reviews que los usuarios hacen de sus libros en un DataFrame con formato (user_id, book_id, rating, timestamp). Por otro lado tenemos información en otro DataFrame que bajamos de GoodReads: (book_id, book_name, avg_rating). Podemos suponer que los Ids de los libros son compatibles. Se pide usar Python Pandas para:

a) Obtener un DataFrame que indique el TOP5 de Ebooks en el sitio de Ebooks. (Para este punto se puede ignorar el segundo DataFrame). (****) (7.5 pts)

b) Obtener un DataFrame que indique qué libros tienen una diferencia de rating promedio mayor al 20% entre el sitio de Ebooks y GoodReads. (***) (7.5 pts)

3) Resolver 2 (dos) y solo 2 de los siguientes ejercicios a elección (si resuelve mas de 2 el ejercicio vale 0 puntos, sin excepciones). En cada caso indicar V o F justificando adecuadamente sus respuestas. Si no justifica vale 0 puntos sin excepciones.

a) Cuando k tiende a infinito el resultado de ranquear los documentos utilizando BM25 es similar al que se puede obtener con TF-IDF.(**) (10 pts)

b) Utilizando front coding parcial logramos que el indice ocupe menos espacio pero nos vemos obligados a realizar búsquedas secuenciales. (*) (10 pts)

c) Agregando log(N+1/fti) en la fromula de TF-IDF logramos eliminar el peso de los documentos largos sobre los cortos. (*) (10 pts)

D) Cuanto mas caracteres tomemos para el armado de n-gramas, menos falsos positivos nos arrojará el indice. (*) (10 pts)

4) Sean los siguientes vectores en 5 dimensiones: v1 = [4 4 -5 -2 3] ; v2= [-3 -2 -4 5 0]; v3=[3 2 -1 -2 1]. Y sean los siguientes 6 hiperplanos aleatorios: r1=[1 1 1 1 -1]; r2=[-1 1 1 -1 -1]; r3=[1 -1 -1 -1 -1]; r4=[1 -1 -1 -1 1]; r5=[1 -1 -1 -1 1]; r6=[-1 1 1 -1 1]. Se pide comparar las alternativas r=3, b=2 vs r=2, b=3 indicando en cada caso que colisiones se producirían. (15 pts) (**)

5) Dado el archivo "AAAAB" se pide comprimirlo usando PPMC de orden máximo=2 y considerando al símbolo de EOF como terminación del archivo. (***) (15 pts)

6) Luego de aplicar la SVD a un cierto set de datos obtenemos las siguientes matrices (por filas): U=[-0.36, 0.25, -0.68, -0.57, -0.08; -0.29 0.51 0.60 -0.36 0.39; -0.51 0.11 0.27 0.18 -0.79; -0.59 -0.74 0.09 -0.10 0.29; -0.40 0.33 -0.29 0.70 0.37] S = [2.41 0 0 0 0;0 0.90 0 0 0;0 0 0.78 0 0;0 0 0 0.25 0;0 0 0 0 0.001] V=[-0.43 -0.41 0.44 -0.33 -0.58; -0.55 -0.03 -0.54 0.56 -0.30; -0.40 0.46 0.64 0.38 0.24; -0.34 0.65 -0.28 -0.60 -0.10; -0.49 -0.43 -0.13 -0.24 0.71]. Se pide:

a) Graficar el set de datos reducido en 2d. [4 pts] (*)
b) Indicar la reconstrucción de la matriz original si usamos k=1 [3 pts] (*)
c) ¿Son todos los números del dataset original positivos? ¿Por qué? [3 pts] (*)

7) Dada la siguiente función de hashing que pertenece a la familia Universal de Carter-Wegman para números enteros: h(x) = [(4*x + 3) mod 13] mod 5 . Usamos h para construir un esquema FKS para las siguientes claves: 20,40,70,10,100. Indicar la estructura final resultante y la en caso de ser necesario la segunda función de hashing a usar para el segundo nivel teniendo en cuenta que debe ser pertenecer a la familia [(a*x + b) mod 13] mod m (***) (10 pts)