

124

+35

A



Organización de Datos 75.06. Segundo Cuatrimestre de 2018. 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, está prohibido hablar desde el lugar, fumar o cualquier actividad que pueda molestar a los demás. El criterio de corrección de este examen está disponible en forma pública en el grupo de la materia.

"You're strong. But I could snap my fingers, and you'd all cease to exist." - Thanos, Avengers: Infinity War

#	1	2	3.1 A	3.2 B	4	5	6	7	Entrega Hojas:
Corrección	B	B-	B	B	B-	B-	B	B	Total:
Puntos	15/15	12/15	10/10	10/10	15/15	10/15	7/10	10/10	89+35/100

Nombre: Federico del Mazo
Padrón: 100029
Corregido por: NATI
124

1) (***) Tenemos información sobre recetas en 3 RDD de Spark	2) (***) Dada la exitosa convocatoria de los Juegos Olímpicos de la Juventud por parte del público, sus organizadores realizan distintos análisis para planificar las jornadas finales del certamen. Es por ello que cuentan con información en los siguientes archivos csv: eventos.csv (id_evento, fecha, id_locacion, nombre_evento, id_categoria_deportiva, cantidad_espectadores) locacion.csv (id_locacion, nombre, capacidad, capacidad_extendida, sede, latitud, longitud) categorias_deportivas.csv (id_categoria_deportiva, nombre, año_de_adopcion) El primer archivo cuenta con información de los eventos, indicando la fecha (en formato "YYYY-mm-dd hh:mm:ss"), el lugar donde ocurrió (id_locacion) y la cantidad de espectadores que asistieron. Además se aporta información sobre la categoría deportiva a la cual pertenece el evento. Por otro lado se tiene información sobre las distintas locaciones en la sedes del certamen en las que ocurrieron los eventos. Contamos con información de su capacidad total de espectadores así como de su capacidad extendida (cuantos asientos extras se pueden brindar sobre la capacidad de la locación). Se desea obtener: a) Nombre de la sede que acumuló la mayor cantidad de espectadores en eventos durante el certamen del 14 al 15 de octubre inclusive. Esto es de vital importancia para distribuir el merchandising oficial del evento, para las fechas finales (7 pts) b) Nombre del evento y nombre de la categoría deportiva de aquellos eventos cuya cantidad de espectadores superó la capacidad de la locación, más allá de la capacidad extendida. Esto es de vital importancia para detectar problemas de seguridad o si es necesario realizar algún cambio de locación. (8 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) Sea un archivo que contiene cinco millones de dígitos de Pi, no se puede saber si es random porque $K(X)$ es intractable. (*) (10 pts)	b) Para poder suponer que un archivo es random debe verificarse que la entropía de Shannon sea máxima (*) (10 pts)	c) Las tablas de frecuencias de los compresores dinámicos de orden 3 o superior pueden ocupar tanto espacio que la compresión termina siendo inefficiente. (*) (10 pts)	d) Solo con compresores aritméticos podemos alcanzar la longitud ideal indicada por la entropía ya que permiten codificar un mensaje en cantidades no enteras de bits (*) (10 pts)
---	--	---	--

4) Desafortunadamente, tenemos un set de datos con muchos puntos y necesitamos utilizar LSH para buscar los puntos más cercanos. Contamos con el siguiente set: {22,14,10,12} y las siguientes 4 funciones de hashing: $h_1(x) = (3x \bmod 7) \bmod 4$, $h_2(x) = (2x \bmod 7) \bmod 4$, $h_3(x) = (2x+1) \bmod 7) \bmod 4$ y $h_4(x) = ((x+3) \bmod 7) \bmod 4$. Se pide: a. Usando $b=2$ y $r=2$, indique cómo quedan las tablas b. ¿Cuáles puntos deberíamos comparar si nuestro query es el {16}? Explique c. ¿Qué podrímos hacer para reducir la cantidad de falsos negativos? Y si quisieramos reducir la cantidad de falsos positivos? (***) (15pts)	5) (**) Se tienen los siguientes documentos: D1: CORDERO SAL PIMIENTA ROMERO D2: CERDO CORDERO SAL CORDERO D3: SAL CERDO LIMON D4: CORDERO ENTRAÑA D5: PIMIENTA PAPA CORDERO PIMIENTA D6: CORDERO CORDERO CORDERO CORDERO Dada la consulta "CORDERO PIMIENTA" dar el resultado de la consulta rankeadas utilizando TF IDF. (10 pts) Considerando como relevante los documentos que no tengan otra carne que no sea CORDERO, calcular la Precisión, Recall y F1 Score. (5 pts)
--	---

6) Se tiene una matriz muy grande donde cada fila representa una imagen de una cara. Se quiere aplicar algún algoritmo de reconocimiento facial utilizando la SVD. a. ¿Cómo podemos determinar el valor de k (cant. de dimensiones a utilizar)? Justifique b. ¿Se podría reducir el espacio que ocupa la matriz sin perder información? c. Una vez obtenido k, ¿Cómo podemos reducir los puntos originales a k dimensiones? d. Si ahora quisiera reconocer una imagen, ¿Cómo podría usar la SVD para ello? (***) (10pts)	7) El COI desea evaluar la aceptación de las nuevas categorías deportivas que se sumaron en el año 2018 a los Juegos Olímpicos de la Juventud. Para ello es necesario que nuestra área de análisis de datos prepare una visualización que muestre a lo largo del tiempo de duración del certamen como fue evolucionando la cantidad de público que han tenido estas nuevas categorías Para desarrollar el punto debe partir como base de la información que cuenta en el punto 2, ampliando con otras posibles fuentes de datos, el contenido de la misma. (***) (10 pts)
--	--

①

RDD {

recetas → (id, nombre, tiempo, dificultad)
 ingredientes → (id-ing, nombre-ing)
 infxreceta → (id, id-ing, cant)

a) Nombres de recetas con corderos.

① cordero = ingredientes.Filter(lambda x: x[1] == 'cordero')

② recetas con cordero = infxreceta.map(lambda x: (x[1], x[0])).
 $\lambda \rightarrow \text{lambda } x:$

. join(cordero)

. map(lambda x: (x[1][0], None))

→ Agrego None
para tener TUPLOS
de Clave, Valor

③ nombresrecetasCordero = recetas.map(lambda x: (x[0], x[1]))

. join(recetasconcordero)

. map(lambda x: (x[1][0]))

. collect()

— Explicación

① Filtra ingredientes para tener solo cordero como un RDD y en un JOIN (inner) Poder filtrar los recetas

1, cordero → 1, cordero.
2, Cordero

② Me quedo con la tupla (id-ingrediente, id-receta) para despues hacer un join y solo quedarme con las recetas con corderos. Despues, me quedo con los id de recetas

NOTA

1, 1, 3	→	1, 1	→	1, (1, Cordero)	→	1, None
1, 3, 3	→	3, 1	→	1, (1, Cordero)	→	1, None
2, 2, 3	→	2, 2	→	1, (2, Cordero)	→	1, None

1, 1, 3
1, 3, 3
2, 2, 3

1, 1
3, 1
2, 2

1, (1, Cordero)
1, (2, Cordero)

1, None
1, None
1, None

id-receta
id-receta
id-receta

③ De los recetas solo me quedo con su id y nombre. Luego, otra vez haciendo un inner join como filtro, me quedo con los recetas de corderos. De ahí, me quedo con los hombres y filalo con collect.

1, Portuguesa, 3m, Fácil → 1, Portuguesa → 1 (Portuguesa, None)
2, Española, 3m, Difícil → 2, Española
→ Portuguesa

||
b) Cantidad de ingredientes para recetas fáciles con cordero.

① $\text{recetasFaciles} = \text{recetas}. \text{filter}(\lambda x\{x[3] == \text{'Fácil'}\})$
. map($\lambda (x[0], \text{None})$)

en punto anterior.

② $\text{RecetasFaciles} = \text{recetasFaciles}. \text{join}(\text{recetas con cordero})$
con cordero
. map($\lambda (x[0], \text{None})$)

③ $\text{Ingredientes} = \text{infXreceta}. \text{join}(\text{recetasFaciles con cordero})$
recetas faciles
con cordero
. map($\lambda (x[1][0], x[1][1])$)
. reduceByKey(lambda x, y: x+y)

④ ~~Ingredientes~~ Ingredientes
~~Recetas Faciles~~
con cordero - con
Nombre = Ingredientes
Recetas
Faciles con cordero
. join(ingredientes)
. map($\lambda (x[1][0], x[1][1])$)
. collect()

- ① Me quedo con los recetas fáciles
② Inner join me sirve de filtro → tengo recetas fáciles con cordero
③ Me quedo con la tupla (id-ing, cantidad) y sumo las cantidades por ing
④ Me quedo con la tupla (nombre-ing, cantidad)

NOTA

(2)

Pandas.read_csv('eventos.csv')

Eventos → (id, Fecha, id-loc, nombre,er, id-cat, cat, cant)

Locacion → (id-loc, nombre, cap, cap-exc, Sede, lat, lon)

Cat_deportivas → (id-cat, nombre, año)

a) Sete con más espectadores en el 24 y 25 de Oct

eventos = eventos[["^{id-loc}fecha", "cant"]]

eventos['Fecha'] = pd.to_datetime(eventos['Fecha'])

Si: no
Parsear,
usar
opcion
format

eventos = eventos.loc[eventos['Fecha'].dt.day == 14]
eventos[eventos['Fecha'].dt.day == 15]

eventos = eventos[['id-loc', 'cant']]

locacion = locacion[['id-loc', 'sede']]

Asumo que son datos
de estos 2000
(del 2 al 18 de Oct).
Si: no, validar año y mes.

eventos x sede = eventos.merge(locacion, left_on='^{id-loc}', right_on='id-loc')

↳ |id-loc|cant|sede|

eventos x sede = eventos x sede[['sede', 'cant']]

↳ sede|cant

eventos x sede.groupby('sede').agg({'cant': 'sum'})

.to_frame()

.sort_values('cant')

.head(1)

me devolverá
serie

con multíples
porque tiene
dos columnas

mece un
solo para
quedarse
en el 1º

b) Eventos y categorios con más espectadores que la cantidad
y la extiende.

(ahorita discute al punto d, uso variables originales)

locacion ['cap+ext'] = locacion ['cap'] + locacion ['cap-ext']

locacion = locacion ['id-loc', 'cap+ext']

eventos con loc = eventos [['id-loc', 'nombre-ev', 'id-cat', 'cant']]

merge (locacion, left_on='id-loc', right_on='id-loc')

↳ id-loc | nombre-ev | id-cat | cant cap+ext

eventos con loc = eventos con loc . loc [eventos [~~['id-loc']~~] ^{Cant} > merge (locacion, left_on='id-loc', right_on='id-loc')
[['id-cat', 'nombre-ev']]

eventos con loc . merge (cat_deportivos, left_on='id-cat', right_on='id-cat')

[['nombre-ev', 'nombre']]

↳ nombre de
cat deportiva.

③

b) Verdadero.

Si la entropía de Shannon se maximiza, los caracteres son equiprobables, y por ende se puede suponer (pero no asumir) que es random.

La entropía luego puede incrementar reducirse (por ej con la transformación de Burrows Wheeler) y de esa forma ver si había algún patrón o secuencia con posibilidades de compresión (mostrando que no era random).

Si es random \rightarrow entropía máxima (Si no entropía \rightarrow no es random)
~~entropía~~ Si entropía max \rightarrow Puede ser random.
~~entropía~~

~~descomprimible~~ ~~no posee redundancia~~

✓

c) Falso.

Si bien $K(X)$ si es ilimitable, hay diversos formas ~~de~~ de ver si un archivo que sigue un patrón es random o no.
Si simplificamos haciendo un programa que calcule los dígitos de π y lo verificamos contra el archivo dato, ya verás como este sigue un patrón (serán dígitos de π) y ya puedes ser el menos sorprendido el programahero, teniendo que $K(X) \neq |X|$, lo cual significa que no es random.

✓

(4)

	22	14	10	12		<u>16</u>
$n_1 (3x \bmod 7) \bmod 4$	3	0	2	1	table 1	2
$n_2 (2x \bmod 7) \bmod 4$	2	0	2	3	coincide en ambos	0
$n_3 (2x+1) \bmod 7 \bmod 4$	3	1	0	0	table 2	1
$n_4 ((x+3) \bmod 7) \bmod 4$	0	3	2	1	coinciden en los	1

OR

AND

Usando $b=2 = \text{tablas}$ y $r=2 = \text{Funciones}$ por tabla pedimos que los puntos sean similares si coincide su minhash en 2 funciones en una de las dos tablas.

Si usamos $h_1 \cup h_2$ como una tabla y $h_3 \cup h_4$ en otra, ningún punto será similar a 16.

En cambio, si usamos $h_1 \cup h_4$ y $h_2 \cup h_3$, vemos que el 14 es similar al 16, ya que coinciden tanto en h_2 como en h_3 .

Reducir
minhash

Para reducir la cantidad de falsos negativos se puede incrementar b , la cantidad de tablas en las que busquen coincidencias, ya que basta que dos mh coincidan en una o la otra (hecho una unión). Esto hace que ~~reduzca los falsos positivos~~ aumente los falsos positivos.

Otra forma de reducir FN es bajando k (~~entrevistado~~)

Reducir
falsos positivos

Para reducir la cantidad de falsos positivos se puede incrementar r , la cantidad de funciones donde pidan que coincidan los mh (hecho intersección). Esto aumenta los FN.

⑤

	CORDERO	PIMENTA
D1	1 · 0,49	1 · 1,81
D2	2 · 0,49	0
D3	0	0
D4	1 · 0,49	0
D5	1 · 0,49	2 · 1,81
D6	4 · 0,49	0

$$\begin{array}{ll} D1 \rightarrow 2,3 & D4 \rightarrow 0,49 \\ D2 \rightarrow 0,98 & D5 \rightarrow 4,11 \\ D3 \rightarrow 0 & D6 \rightarrow 1,96 \end{array}$$

Consulta: D5, D1, D6, D2, D4

Relevanzas	recuperados			no recuperados
	D1	D5	D6	
no relevanzas	D2	D4		D3

$$\text{Precision} \rightarrow \frac{\text{rei rec}}{\text{rec}} = \frac{3}{5}$$

$$\text{Recall} \rightarrow \frac{\text{rei rec}}{\text{rei}} = \frac{3}{3}$$

$$\text{F1Score} \rightarrow \frac{(\beta^2 + 1) P.R}{P+R} \rightarrow \frac{2 P.R}{P+R} = \frac{3}{4}$$

$$\text{con } \beta=1$$

log₂($\frac{N_{docs}+1}{F_d}$)

2. Usar base
Cant de docs donde aparece

TF * IDF



Cantidad de veces que aparece termino en doc

$$IDF(\text{cordero}) = \log_2\left(\frac{7}{5}\right) = 0,49$$

$$IDF(\text{pimenta}) = \log_2\left(\frac{7}{2}\right) = 1,81$$

mal uso IDF

(b) baki

⑥ a) Para saber cuantas dimensiones utilizar, podemos hacer SVD(M) la cual nos da las 3 matrices U, Σ y V^T .

Luego, en Σ tenemos los autovalores de la matriz original, ordenados de mayor a menor importancia.

Tomando la energía (información) de la matriz como la suma de sus autovalores del cuadrado, y teniendo eso en cuenta como el 100% de mi información, podemos ir viendo cuanta energía acomoda hoy según la cantidad de autovalores usados. Podemos fijar un valor de info a tener (ej: 90%) y usar $K = \text{autovalores necesarios}$

Ejemplos, si probamos los autovalores y su parte de la energía es cero, probablemente encontraremos un 'codo' de cortar y usar donde hay un salto considerable de energía disponible.

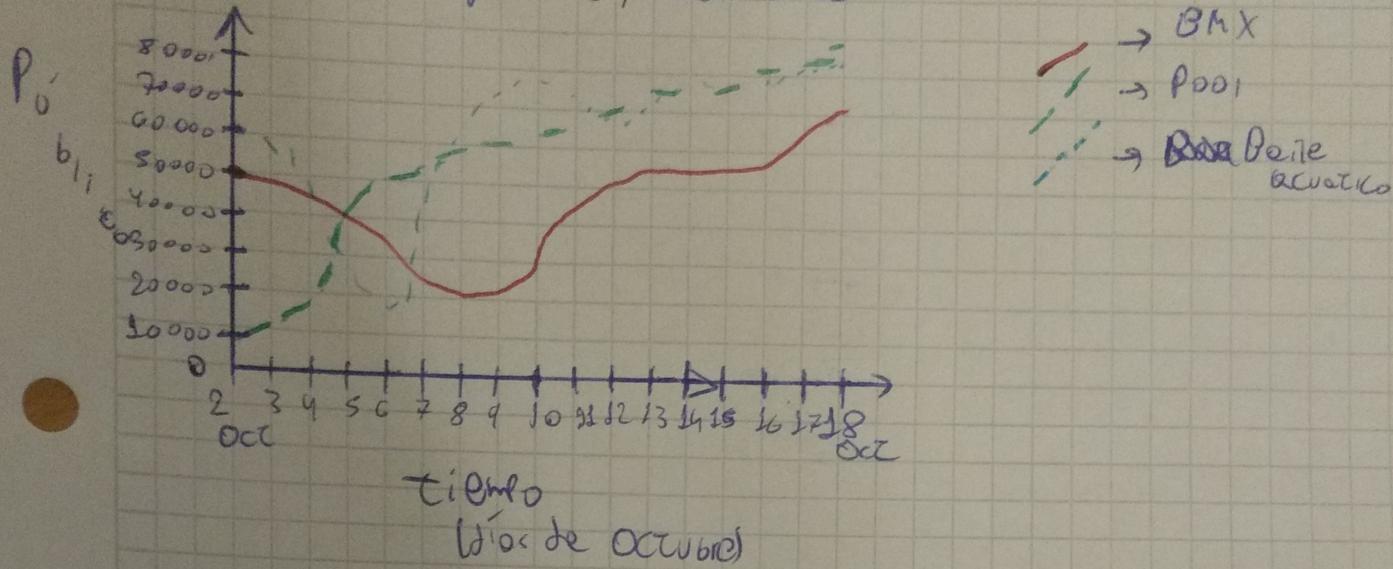
b) Para no perder info, simplemente podemos usar DVS reducida, que retira los autovalores nulos, quitando dimensiones sin perder info (porque los que retira son los que tienen la resta)

c) Si queremos reconocer la info, podemos reconstruir la M y conservarla a la original no reducida.

d)

(7)

Especadores de Nuevos
Categorías, 55002018



Graficando en el mismo gráfico los niveles los nuevos categorías y usando los espectadores en función del tiempo, se puede comparar fácilmente los altos y bajos de cada categoría. Pudiendo ver que se le puede atribuir al atractivo innato del deporte (Por ej.: crecer establecimientos) como que se le atribuye al golpe de innovación (Por ej.: un gran primer día, y luego decayer).

Otra alternativa sería usar un heatmap, normalizado para considerar solamente el crecimiento / decrecimiento relativo de cada actividad en si misma.



(Hubiese estado mejor en la piz)