TREBALL DE FI DE GRAU

Grau en Estadística

Títol: Optimització dels trajectes de les comandes d'una cadena de restaurants de Melbourne

Autor: Marcel Canals Codina

Director: Julia de Frutos Cachorro

Departament: Matemàtica Econòmica, Financera i Actuarial

Convocatòria: Juny 2022

UNIVERSITAT DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

Resum

Aquest treball consisteix en l'estudi d'un problema d'optimització dels trajectes que realitzen els treballadors que transporten menjar a domicili d'una cadena de restaurants de Melbourne durant l'any 2018.
Aquesta ciutat situada al sud-est d'Austràlia, compta amb tres restaurants anomenats: Bakers, Nickolson i Thompson situats en el centre de l'àrea urbana, també coneguda com a Melbourne City Centre.

Donada una base de dades, s'ha dut a terme un procediment detallat des de zero seguint els següents passos: un ampli preprocessament de les dades, un anàlisi descriptiu gràfic i una optimització dels trajectes de transport de les comandes tenint en compte els restaurants com a nodes d'inici i la posició geogràfica dels domicilis dels clients com a nodes destí.
Per tal de resoldre diferents escenaris segons la comanda, s'ha portat a cap unes subdivisions en la mostra estratificant per tipus d'àpat, dies laborals o festius i caps de setmana o dies entre setmana.

El problema d'optimització consisteix en la minimització d'aquestes distàncies; s'ha realitzat mitjançant un programa d'optimització lineal amb restriccions seguint el concepte d'arbre d'expansió mínima, és a dir, enllaçar tots els nodes d'una xarxa de forma directa o indirecta amb la mínima longitud dels arcs d'enllaç. S'ha dut a terme a través d'una matriu binària on les variables han sigut els arcs entre nodes i sempre complint els requisits de passar exactament una vegada per cada domicili i retornar al restaurant d'inici en finalitzar la ruta.

Paraules clau: Optimització lineal amb restriccions, Melbourne, Minimització de distàncies, Arcs i nodes, Anàlisis descriptiu gràfic, Preprocessament, Arbre d'expansió mínima.

“Optimization of the deliveries of orders from a Melbourne restaurant chain”

This work consists of the study of an optimization problem. The main idea is optimizing the transport route of workers that carry food orders from a Melbourne restaurant chain during the year 2018.
Located in southeast Australia, this city has three restaurants named Bakers, Nickolson and Thompson located in the centre of the urban area, also known as the Melbourne City Centre.

The work that has been carried out has been, given a database of these orders, to carry out a detailed procedure from scratch following these steps: a detailed pre-processing of the data, a graphic descriptive analysis and an optimization of the journeys of transport of the orders considering the restaurants as start nodes and the geographical position of the client’s houses like destination nodes. In order to solve different scenarios according to the order, some subdivisions have been recalled in the sample stratifying by type of meal, working days or holidays and weekends or days during the week.

The optimization problem consist of minimizing these distances; it has been done through a linear optimization program with restrictions following the concept of minimum spanning tree, that is, link all the nodes of a network directly or indirectly with the minimum length of the bonding arcs. through a binary matrix where the variables have been the arches between nodes and always fulfilling the requirements of passing exactly once through each address and returning to the starting restaurant at the end of the route.

Keywords: Linear optimization with restrictions, Melbourne, Minimization of distances, Arches and nodes, Graphic descriptive analysis, Pre-processing, Minimum spanning tree.

Classificació AMS (MSC 2010):

62-07	Data análisis
62-09	Graphical Methods
05C12	Distance in graphs
74Pxx	Optimization

ÍNDEX

Capítol I: INTRODUCCIÓ

En aquest apartat es descriuen els conceptes inicials del projecte, el contingut de la base de dades escollida i els fonaments dels problemes de geolocalització entre d'altres.

1. Idees Inicials

El principal objectiu del treball va consistir en optimitzar trajectes de menjar a domicili perquè actualment és un afer que afecta de manera consistent a escala mundial, a més s'ha incrementat arran del Covid i és un afer que es pot observar constantment.

Sobretot a Barcelona han aparegut i crescut grans empreses com Glovo, Deliveroo, Just Eat i Getir que es dediquen al transport de menjar a domicilis. La millora de les tecnologies i el fet que la humanitat hagi pres un distanciament social a causa d'aquest virus ha fet augmentar en gran manera aquesta pràctica de demanar menjar des de casa.

Per això es va considerar si es podria fer un treball de fi de grau d'alguna d'aquestes empreses. Però sabent que aquestes dades són privades i que no es podrien aconseguir, va ser quan es va començar a buscar a la web de bases obertes anomenada Kaggle.

Després d'alguns dies de recerca es va trobar el que es buscava, una font de dades que tractava aquest mateix concepte i complia les exigències. En aquell punt va ser quan s'inicia l'estudi d'aquesta cadena australiana de restaurants.

2. Què se'n sap de Melbourne i el centre de la ciutat?

Melbourne és una de les ciutats més importants d'Austràlia. Aquesta illa del continent oceànic està situada a l'hemisferi sud, a prop de Japó, Nova Zelanda i totes les illes de Nova Guinea. Melbourne és una de les ciutats més grans situada dins l'estat de Victoria i compta amb una superfície metropolitana total de 9993 km^2 (una mica més petita que Nova York).

Però la cadena de restaurants no treballa per tota la ciutat sinó només per la regió central coneguda com a *Central Bussines District* (CBD) i per tant ens centrarem en aquesta zona, que compta amb una superfície de 1,79 km^2. Aquesta part de la ciutat, també anomenada "*the City*", és la més antiga i va ser construïda l'any 1837. Hi resideixen 47.000 habitants, compta amb uns quants gratacels i edificis d'arquitectura Contemporania i Victoriana[1].

Ara que es té una idea sobre la regió d'estudi es pot passar a parlar del contingut de la base de dades.

[1] Tota la informació que fa referencia a les característiques de la ciutat de Melbourne s'ha extret de la Wikipedia

3. La base de dades de Melbourne CBD

Tot i que dins la web de Kaggle es van trobar fins a sis taules relacionades amb el tema, es van escollir només dues (les altres quatre havien estat tractades i l'objectiu d'aquest TFG era realitzar un estudi d'una base de dades des de zero).

Per tant, la base de dades està composta per dues taules. Una primera anomenada "branches" que dona informació sobre les tres branques on es preparen els menjars d'aquesta cadena de restaurants i una segona taula anomenada "comandes" on hi ha tota la informació sobre cada una de les comandes que fa el client.

3.1 Taula "branches"

Està formada per les següents variables:

- **Branch_code**: Codi únic de cada branca
- **Branch_name**: Nom complet de cada branca
- **Branch_lat**: Latitud de la branca en qüestió
- **Branch_lon**: Longitud de la branca en qüestió

Aquesta primera taula serveix per geolocalitzar els restaurants i identificar-los.

3.2 Taula "comandes"

Comprèn les següents variables:

- **Order_id**: Identificador de la comanda
- **Date**: Dia de la comanda
- **Time**: Hora exacta de la creació de la comanda
- **Order_type**: Tipus de menjar que es demana. Hi ha tres tipus: esmorzar, dinar i sopar
- **Branch_code**: Codi de la branca. Serveix per unir aquesta taula amb l'anterior
- **Order_items**: Cadena de caràcters que mostren els plats de la comanda
- **Order_price**: Preu de cada comanda en dòlars australians
- **Customer_lat**: Latitud de l'habitatge del client
- **Customer_lon**: Longitud de l'habitatge del client
- **CustomerHasloyalty**.: Variable lògica que indica si un client té una targeta de fidelitat
- **Distance_to_customer_KM**: Distància entre el client i una de les branques del restaurant
- **Delivery_fee**: Preu de l'enviament, de la tramesa

Sobretot es treballarà amb aquesta taula per fer tots els càlculs per les optimitzacions.

4. Conceptes de geolocalització

A continuació s'expliquen els termes "longitud" i "latitud".

4.1 Longitud

La longitud geogràfica del planeta Terra representa l'angle imaginari que determina la distància entre un punt i el meridià de Greenwich o meridià 0. Aquest meridià divideix el món en dues regions, l'occidental (Oest) i l'oriental (Est).

La longitud es pot mesurar de diferents formes:

- Entre 0° i 360°, augmentant cap a l'Est del meridià 0°
- Entre 0° i 180° indicant a quin hemisferi ens referim (W o E) pertany
- Entre -180° i 180°, de 0° a 180° Est i de -180° a 0° Oest

Les dades de Melbourne s'han mesurat de la tercera manera així que es pot dir que la **longitud té un domini de [-180,180]** i perquè quedi clar a l'hora de visualitzar gràfics la **longitud representa l'eix de les *x***.

4.2 Latitud

Per altra banda, la latitud representa l'angle que determina la distància entre un punt qualsevol i l'equador (la línia imaginaria horitzontal que divideix el món en dos hemisferis Nord i Sud). Es pot representar de dues formes:

- Entre 0° i 90° escrivint N o S després dels graus
- Entre -90° i 90°, sent els graus negatius el sud i els positius el nord

Per tant, es pot acceptar que **el domini de la latitud és [-90,90]** i al contrari de la longitud, **la latitud representa l'eix de les *y*.**

Un altre aspecte a parlar sobre les posicions geogràfiques és que es poden representar amb graus, minuts i segons o simplement amb graus i nombres decimals.
Per exemple, la seqüència 70° 3' 56'' seria 70.06556°, ja que l'operació corresponent és:

70° 3' 56"= 70° + 3'/60 + 56"/3600 = 70.06556°

En aquest TFG s'utilitzarà el grau decimal com a mesura predeterminada. Per una visió més gràfica d'aquests conceptes podem visualitzar la figura 4.1.

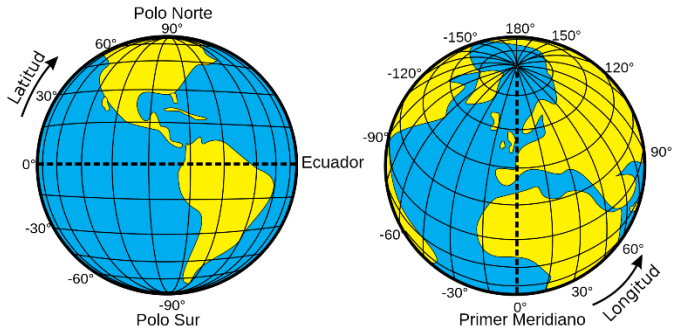


FIGURA 4.1: REPRESENTACIÓ DELS EIXOS EN EL GLOBUS TERRAQÜI

5. Tipus de problema

En el transcurs d'aquest treball es veurà que el problema tracta d'optimitzar distàncies a través d'una matriu binària, aquest problema prové de la branca de l'optimització lineal sense restriccions amb el rerefons del **problema del flux de la xarxa**.

Per entendre el problema del flux de la xarxa és essencial familiaritzar-se amb els següents conceptes:
Xarxa o Graf: R=(N,A) es defineix mitjançant dos símbols:

- Els **nodes** (conjunt N) són punts extrems d'una xarxa, també es poden anomenar vèrtex o punts extrems
- Els **arcs** (conjunt A) consisteixen en pars ordenats de nodes i representen una possible direcció que passa entre aquests nodes, aquests arcs se'ls hi associa una distància (cost, temps) representativa al problema que s'està tractant

Per tant, un graf és un conjunt de nodes i arcs que formen un flux, l'estudi d'aquest flux i la minimització de la distància és un component essencial per la majoria de problemes d'optimització.

Una de les característiques d'aquests problemes de flux és l'existència de nodes font (en aquest cas els restaurants actuarien com a tal) i nodes destí (serien els clients).

Dins dels problemes de fluxos de xarxes hi ha diferents tipus: arbre d'expansió mínima, ruta més curta, flux màxim... En aquest TFG s'aborda el primer tipus, però s'expliquen breument els tres i les diferències principals.

5.1 Arbre d'expansió mínima

Un arbre d'expansió mínima és un tipus especial d'arbre que minimitza les longituds de les vores de l'arbre. Per posar un exemple, una companyia telefònica vol cablejar a través d'una línia els barris A fins H, separats per diferents longituds distribuïts de la forma que indica la figura 5.1; en minimitzar la quantitat de cables, la companyia s'estalviarà diners.

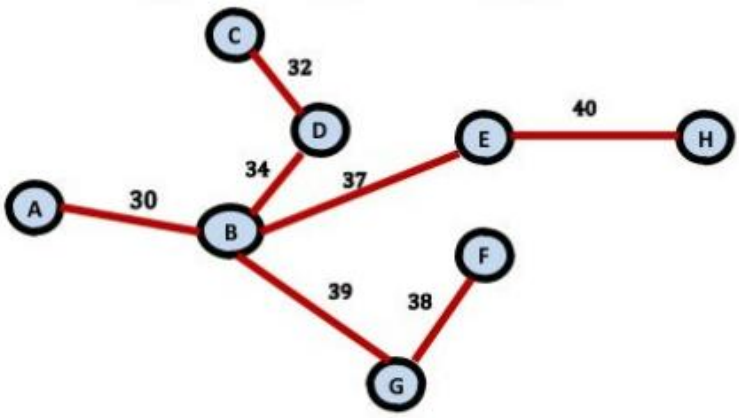


FIGURA 5.1: ARBRE D'EXPANSIÓ MÍNIMA

Un arbre té un camí que uneix dos vèrtexs qualssevol. En canvi, un arbre d'expansió té les següents propietats:

- Conté tots els vèrtexs del gràfic original
- Abarca tots els vèrtexs
- És acíclic: El graf no té cap node que torni a ell mateix

Per trobar la distància mínima hi ha diferents algoritmes com l'algoritme de Kruskal, el de Prim o el de Boruva.

5.2 Ruta més curta

El mètode de la ruta més curta permet buscar la solució a un problema d'optimització que resulti d'una combinatòria i de diferents aplicacions. L'objectiu d'aquest mètode és descobrir rutes curtes o de menor cost, depenent del cas que sigui, que va des d'un node específic fins cada un dels altres nodes de la xarxa.

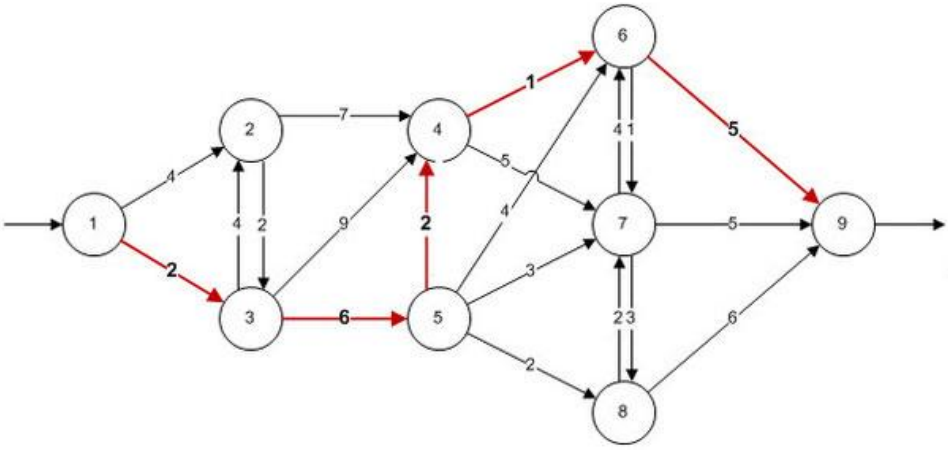


FIGURA 5.2: RUTA MÉS CURTA

Per exemplificar aquest mètode es pot visualitzar la figura 5.2. Una família vol viatjar de la ciutat 1 fins a la 9, tenint en compte que per poder fer-ho haurà de travessar carreteres amb diferents longituds que enllaçaran amb altres ciutats. La ruta marcada en vermell mostra la millor opció per estalviar-se quilometres i, per tant, realitzar la ruta més curta possible.

Per aquest mètode, l'algorisme de Dijktra és el que millor s'adequa i el més utilitzat.

5.3. *Flux màxim*

Existeixen fluxos que viatgen d'un únic lloc d'origen cap a un únic lloc de destí a través d'arcs que connecten nodes intermediaris. Aquests arcs tenen una capacitat màxima de flux i el problema tracta d'enviar des del node font fins al node destí la major quantitat possible de flux.

Alguns exemples on és important la quantitat de flux que passa a través de la xarxa són les línies d'oleoductes, transmissions de dades o xarxes elèctriques.

A diferència dels altres dos mètodes, aquest mètode té més paràmetres

- **Flux**: Circulació d'unitats homogènies d'un lloc a un altre
- **Capacitat de flux**: Capacitat d'unitats que poden entrar pel node font i sortir pel node destí
- **Capacitats residuals**: Capacitats restants una vegada el flux passa l'arc

L'algoritme per excel·lència per resoldre aquests problemes és Ford Fulkerson; un mètode que proposa camins en els quals s'augmenta el flux fins que s'arriba al flux màxim. La idea és trobar una ruta de penetració que uneixi els nodes origen i destí.

La figura 5.3 mostra un cas pràctic d'aplicació d'aquest mètode on es vol transportar el flux màxim de tones de carburant des del node 1 (mina de petroli) fins al node 5 (fàbrica de cotxes).

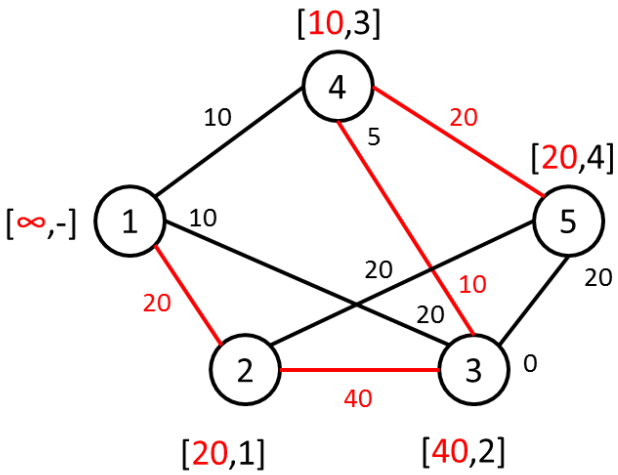


FIGURA 5.3: ALGORITME FORD FULKERSON APLICAT A UN PROBLEMA DE FLUX MÀXIM

La diferència més important entre el l'arbre d'expansió mínima i el flux màxim i la ruta més curta és que pel primer mètode s'imposa la condició que s'ha de passar per tots els nodes. En

canvi pels altres dos mètodes no és necessari. A més, el mètode del flux màxim té en compte la capacitat de cada arc i node, en canvi, ni l'arbre d'expansió ni la ruta més curta no ho tenen present. Per últim, l'arbre d'expansió mínima té la propietat de ser acíclic, és a dir, el mètode no té cap node que torni a ell mateix.

A continuació es veu el següent exemple.

5.4 Exemple: El problema del viatjant (TSP)

Un viatjant es planteja recórrer diferents ciutats dels Estats Units durant un estiu, però ho vol fer de la següent manera: No vol passar dues vegades per la mateixa ciutat, sortirà de la seva ciutat natal i finalitzarà el viatge en aquesta mateixa ciutat i per últim vol estalviar-se els màxims quilòmetres possibles. Per tant, quin trajecte hauria de seguir?

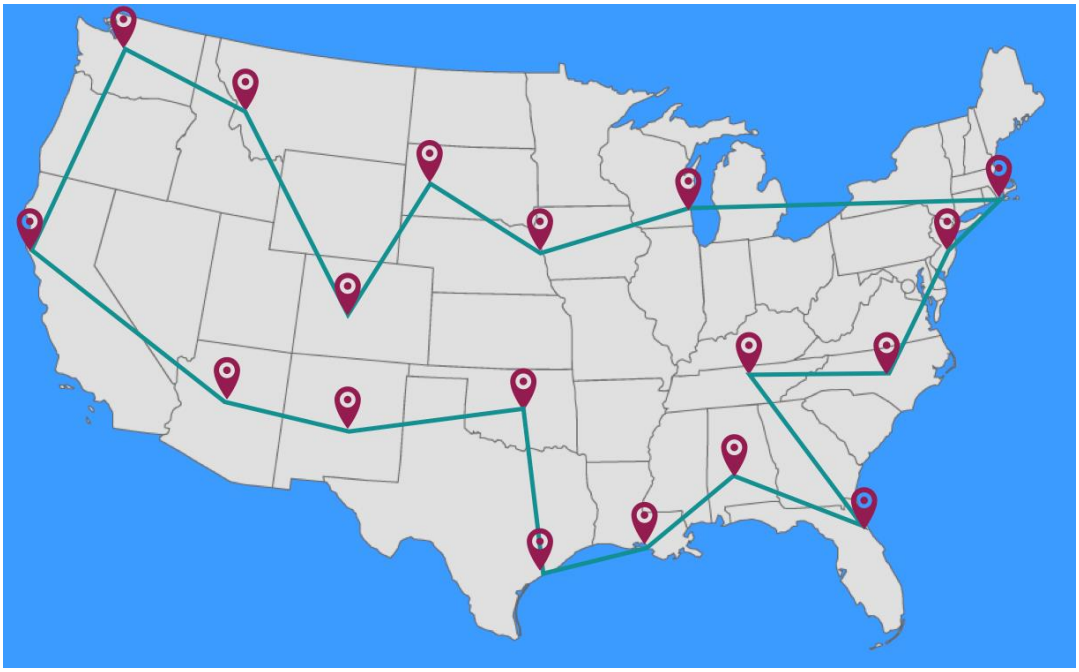


FIGURA 5.4: RUTA DEL VIATJANT PER ESTATS UNITS SEGUINT UN PROBLEMA D'OPTIMITZACIÓ BINÀRIA

Com es pot veure en la figura 5.1, consisteix a trobar la ruta més curta entre un conjunt de N llocs S de forma que cada lloc es visiti exactament una vegada menys l'inicial amb el qual la ruta no es pot creuar.

El problema plantejat en aquest TFG és semblant al de l'exemple, però amb la diferència que les variables no són els nodes que representen els llocs sinó els arcs entre dos nodes. A més, els requisits esmenats no s'introdueixen en la funció objectiu sinó que són restriccions. Les característiques d'aquesta optimització es veuran més endavant.

6. Visualització del problema

Ara que s'ha après el tipus de problema que es tracta, es pot visualitzar, a través de la figura 6.1, el problema que es vol optimitzar.

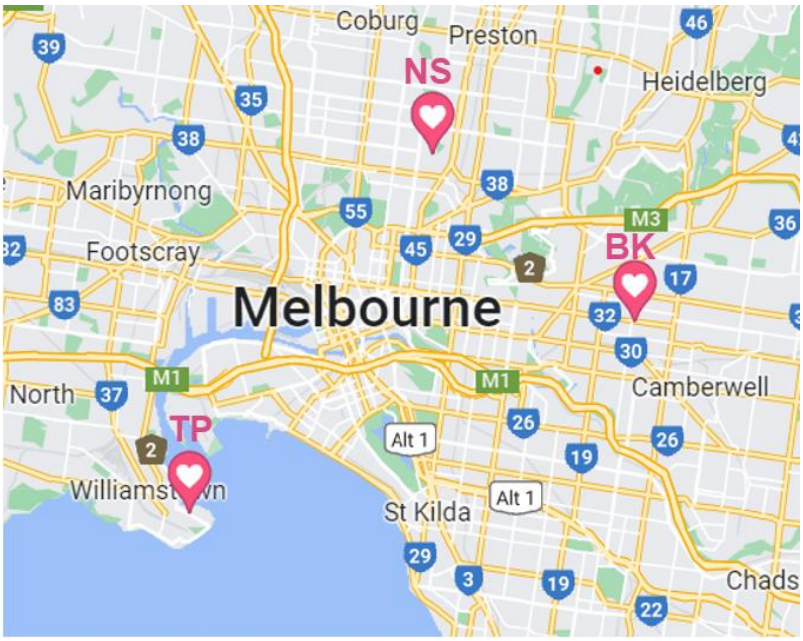


FIGURA 6.1: MAPA DE LA CIUTAT DE MELBOURNE I LOCALITZACIÓ DELS 3 RESTAURANTS

Aquests tres punts representen els restaurants d'aquesta cadena, es vol estudiar els trajectes tenint en compte que un treballador pot dur a terme comandes de diferents restaurants, ja sigui Thompson (TP), Bakers (BK) com Nickolson (NS).

Com es pot apreciar, els restaurants estan als afores de Melbourne CBD.
En primer lloc, l'establiment Thompson està tocant el port, al costat de la badia Port Phillip. Està situat al sud-oest del centre i és el més allunyat dels tres restaurants.

En segon lloc, el restaurant Nickolson que està situat al nord de Melbourne CBD és el que està més a prop de la regió d'estudi. Al costat seu està el canal Merri Creek, el parc Balfe i el seu nom fa referència a *Nickolson Street*, carrer on està situat.

L'últim restaurant està situat a l'est del centre, justament al carrer *Barker's Road* i a prop dels jardins centrals. Aquesta carretera és una de les més importants que creua el centre de Melbourne, com si es tractés de la Gran Via per Barcelona.

7. Assumpcions del problema

Per tal d'establir una idea clara del que es vol estudiar, s'han redactat unes assumpcions que indiquen els fonaments de com serà el problema a optimitzar.

- S'Assumeix que si dos clients viuen al mateix bloc de pisos, actuaran com un únic client que ha realitzat dues comandes, ja que la base de dades no diferencia en altura, només en latitud i longitud

- Se suposa que l'encarregat/encarregada de fer els trajectes té sempre les mateixes condicions per cada una de les comandes, no canvia si realitza comandes de Bakers que si les agafa pel restaurant Nickolson. Tampoc canvien les condicions meteorològiques que podrien afectar el trajecte ni les condicions de tràfic
- La ruta entre dos clients i/o client i restaurant és la mateixa en ambdós sentits. Això significa que la distància des del punt A fins al B és la mateixa que des del punt B fins a l'A.

8. Objectius

També s'han marcat els objectius que es volen aconseguir durant el transcurs d'aquest treball, són els següents:

- Tractar una base de dades des de zero, realitzant els canvis pertinents perquè sigui apte per l'estudi
- Processar aquesta base de dades de la forma més paramètrica possible, optimitzant el codi utilitzat
- Analitzar la base de forma visual a través de gràfics geogràfics que mostrin la latitud i longitud de tots i cada un dels clients, estudiant-los per diferents factors
- Modelitzar diferents escenaris per disminuir la mida de la mostra
- Optimitzar els trajectes de les comandes seguint la idea de flux de la xarxa que permetin crear la ruta òptima
- Minimitzar les distàncies d'escenaris de diferents tipus de mida i de característiques a través de la matriu binària

Capítol II: METODOLOGÍA

Un cop acabada la introducció és hora de parlar sobre la construcció global d'aquest treball i detallar quin tipus d'optimització s'ha portat a cap.

1. Softwares i procediment

Per dur a terme aquest treball s'han utilitzat dos *softwares*. El principal ha sigut l'eina estadística **R** i en concret l'entorn ***RStudio***. El fet que sigui el llenguatge més emprat en la carrera i és amb el que més còmode es sentia l'autor ha decantat aquesta tria. R permet realitzar gràfics de diverses maneres, funcions de tota mena i és útil a l'hora de tractar i modificar estructures de dades.

Per altra banda, el *software* per dur a terme les optimitzacions ha sigut **SAS**. Al ser de pagament s'ha hagut de treballar amb ***SAS on Demand***, la versió online per estudiants que et permet dur a terme tota mena de models i optimitzacions. SAS és una eina potent d'anàlisi i optimització usada mundialment que permet portar a cap operacions amb dades de gran mida i modelitzar diferents situacions.

Com s'han barrejat exactament aquests softwares?

Per facilitar la comprensió de l'estructura global del TFG s'ha creat el següent diagrama (figura 1.1) que explica el procediment dut a terme, els *scripts* emprats, les dues taules originals i els *softwares* utilitzats en cada moment.

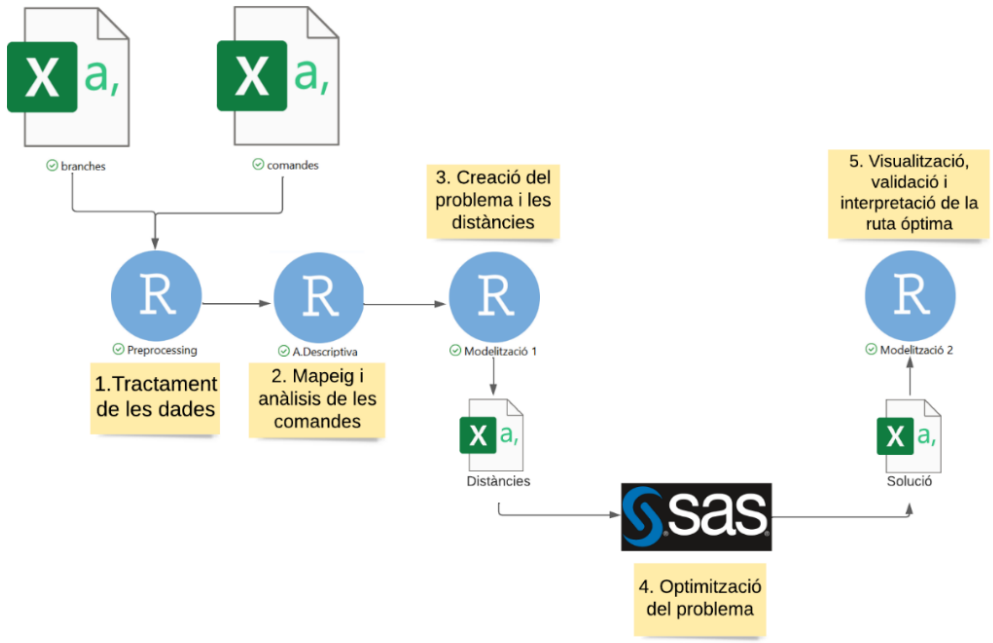


FIGURA 1.1: DIAGRAMA DEL PROCEDIMENT GLOBAL DEL TFG

1. **Tractament de les dades**: S'introdueixen les taules "branches" i "comandes" a R i s'inicia el preproccesament de la base de dades per eliminar errors, variables innecessàries, missings i outliers
2. **Mapeig i anàlisi de les comandes**: Descriptiva numèrica inicial i gràfica de les comandes en funció de diferents variables
3. **Creació del problema i les distàncies**: Reducció de la mostra a través de la estratificació segons diversos factors i creació de les distàncies entre clients i restaurants d'un escenari en concret
4. **Optimització del problema**: Una vegada construïdes les distàncies s'exporten en *csv* i s'importen a SAS on es realitza la minimització que indica la ruta òptima
5. **Visualització, validació i interpretació de la ruta**: S'exporta la solució en *csv* i s'importa a R novament per visualitzar i validar la solució per poder interpretar-la

2. Optimització binària

Aquesta part del treball tracta de trobar un model d'optimització que permeti crear la ruta entre els restaurants i els clients.

Després d'investigar sobre possibles mètodes es va decidir que es portaria a terme construint una matriu quadrada binària on cada valor d'aquesta estigués associat al vèrtex entre dos nodes i que prengués el valor 1 si la ruta òptima passa per aquell vèrtex i 0 en cas contrari.

Aquesta matriu binària dependrà de les distàncies calculades entre vèrtex que també es guarden formant una matriu de mateixa dimensió.

Per exemplificar-ho, es comença per la següent matriu de distàncies:

Distancies	BK	ORDC01406	ORDZ10125	ORDR12733
BK	99999	1254.34	2893.56	2399.03
ORDC01406	1254.34	99999	6302.89	7426.94
ORDZ10125	2893.56	6302.89	99999	1543.56
ORDR12733	2399.03	7426.94	1543.56	99999

TAULA 2.1: EXEMPLE D'UNA MATRIU DE DISTÀNCIES (M)

Com es veu en la taula 2.1, es tracta del restaurant Bakers i tres clients. Es poden veure diferents detalls que poden cridar l'atenció. Quina distància s'ha calculat? Per què la diagonal pren el valor 99999 i no 0 com seria lògic? Quina gràcia té que mantindré tota la matriu si amb la part superior de la diagonal hi hauria prou?

2.1 Distàncies

Existeixen diferents maneres de calcular la distància entre un punt A i un punt B. La distància euclidiana, la distància de Manhattan... Tot i que en el conjunt de dades es té una variable anomenada ***distanceKM*** l'objectiu d'aquest TFG és dur a terme un treball des de zero així que com que també es tenen la latitud i longitud dels clients es va decidir ignorar-la i crear la distància (en metres) través d'una funció pròpia anomenada **Calc_dist**[2].

La funció auxiliar anomenada **distGeo** calcula la distància euclidiana a partir de la latitud i longitud de dos punts en concret. Per això es va decidir que la distància que s'utilitzaria fos l'**euclidiana[3].**

2.2 Termes en l'optimització

Matriu binària

Dins de SAS s'introdueix la matriu de les distàncies i les variables del model són el conjunt de X on X és la matriu binària. Tornant al cas anterior, es mostra la taula 2.2.1:

X	BK	ORDC01406	ORDZ10125	ORDR12733
BK	*X[1,1]*	*X[1,2]*	*X[1,3]*	*X[1,4]*
ORDC01406	*X[2,1]*	*X[2,2]*	*X[2,3]*	*X[2,4]*
ORDZ10125	*X[3,1]*	*X[3,2]*	*X[3,3]*	*X[3,4]*
ORDR12733	*X[4,1]*	*X[4,2]*	*X[4,3]*	*X[4,4]*

[2] El codi de la funció que crea les distàncies euclidianes es troba en l'annex 5

[3] La distancia euclidiana entre dos punts és: $d_E(P_1P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

TAULA 2.2.1: EXEMPLE D'UNA MATRIU BINÀRIA

On *X[2,1]* és la distància entre el client **ORDC01406** i el restaurant **Bakers** i serà 1 si la ruta passa per aquest vèrtex i 0 en cas que no ho faci.

Funció objectiu

Per tant, tenint en compte la matriu binària **X** i la matriu **distancies** es pot formular la funció objectiu.

$$min\ z\ = \sum_{i=1}^{N}\sum_{j=1}^{N} X_{i,j} * distancies_{i,j}$$

$$N\ \in \{1,2,\dots n\}$$

$$n = nº\ de\ clients + nº\ de\ restaurants\ del\ problema$$

Restriccions

També és necessari implementar restriccions per tal que es compleixi els requisits anomenats de passar exactament una vegada per cada un dels clients. $N \in \{1,2\dots n\}$ on *n* és el nombre de nodes.

Constriccions perquè la ruta no passi dues vegades pel mateix client:

$$ClientCol_i: \sum_{j=1}^{N} X_{i,j} = 1$$

$$ClientRow_j: \sum_{i=1}^{N} X_{i,j} = 1$$

Constricció teòrica que anul·la els bucles[4] dins la ruta:

$$Bucle\{a_1 \in N, a_2 \in N, a_3 \in N, \dots, a_n \in N\}: \quad X_{a_1a_2} + X_{a_2a_3} + \dots + X_{a_{n-1}a_n} = n$$

$$on\ a_1 \neq\ a_2 \neq \dots \neq\ a_n$$

En un principi, també es tenia la restricció que anul·lava la distància entre un client i ell mateix:

$$NoDiag: \sum_{i=1}^{N} X_{i,i} = 0$$

[4] En el capítol de resultats es parlarà àmpliament del problema de bucles, com s'ha solucionat de manera pràctica i quines restriccions s'han afegit en cada escenari.

Però es va decidir que no era necessària si s'intercanviava abans de passar-ho al model, el valor 0 de la distància entre un mateix per 99999 (un valor que de segur el model optimitzat no escull per passar-hi la ruta) i així estalviar memòria del software de SAS.

És vital comentar que aquest mètode de matriu binària pot ser útil en el cas que, a diferència d'aquest TFG, la distància entre A i B no sigui la mateixa que entre B i A. Hi ha problemes més semblants a la realitat on les rutes de transport són totalment diferents d'anada que de tornada, ja que no sempre existeixen carrers de doble sentit.

Una vegada s'ha vist quin procediment i quin tipus d'optimització es durà a terme, es comença per la part inicial, el tractament de les dades.

Capítol III: PREPROCESSAMENT

El tractament o preprocessament de dades és essencial per iniciar qualsevol treball. És tracta d'un seguit de tècniques per millorar la base de dades, fer-la apta per la modelització i identificar i arreglar els possibles erros que s'han produït dins de la base.

En aquest treball se li ha donat especial importància aquest tractament i a l'optimització de codi i per això s'ha volgut explicar pas a pas.

1. Variables en detall

La taula 1.1 fa referència a la taula "branches" que es compren de 4 variables i 3 registres.

NOM	*TIPUS*	*DESCRIPCIÓ*	*EXEMPLE*
branch_code	Character	Codi únic de cada branca	NS
branch_name	Character	Nom complert de cada branca	Nickolson
branch_lat	Numeric	Latitud de la branca en qüestió en grau decimal	-37.7730
branch_lon	Numeric	Longitud de la branca en qüestió en grau decimal	144.9836

TAULA 1.1: CONTINGUT DE LA TAULA "BRANCHES"

La taula 1.2 en canvi, fa referència a la taula "comandes". Té unes dimensions de 12 variables i 500 observacions.

NOM	*TIPUS*	*DESCRIPCIÓ*	*EXEMPLE*
order_id	Character	ID de la comanda	ORDC01406
date	Character	Dia de la comanda (YYYY-MM-dd)	2018-08-07
time	Character	Hora exacta de la creació de la comanda (hh:mm:ss)	15:16:03
order_type	Character	Tipus de menjar que es demana ("Breakfast","Lunch" i "dinner")	Lunch
branch_code	Character	Codi de la branca(NS,BK,TP)	NS
order_items	Character	Cadena de caracters que mostren els plats de la comanda	[('Fries', 6), ('Salad', 4)]
order_price	Numeric	Preu de cada comanda en dolars australians	140.80
customer_lat	Numeric	Latitud del domicili del client	-37.81243
customer_lon	Numeric	Longitud del domicili del client	144.9978
customerhasloyalty.	Integer	Variable que indica si un client té una tarjeta de fidelitat(0,1)	1
distance_to_customer_km	Numeric	Distància entre el client i la branca del restaurant en qüestió	8.335
delivery_fee	Numeric	Preu de l'enviament	13.68

TAULA 1.2: CONTINGUT DE LA TAULA "COMANDES"

1.1 Comentaris sobre les variables

La variable ***time*** és l'hora exacta de la creació de la comanda, és a dir, quan la comanda surt de la cuina d'un dels restaurants i està llesta per ser enviada al client.

Pel que respecta a la variable ***order_type*** hi ha tres tipus: "*Breakfast*" que seria l'esmorzar, "*Lunch*" el dinar i "*Dinner*" el sopar.
S'ha de tindre en compte que els horaris d'alimentació espanyols són molt diferents de la resta i es veuran resultats com sopar a les 7 o dinar a les 12 per exemple.
També s'ha de vigilar amb la traducció de *dinner* que és sopar i no dinar com es podria arribar a pensar.

- "*Breakfast*": Horari de 8:00:00 a 12:00:00
- "*Lunch*": Horari de 12:00:01 a 16:00:00
- "*Dinner*": Horari de 16:00:01 a 20:00:00

El menjar que s'ofereix en aquests restaurants és molt variat, ja que cuinen pels tres àpats del dia. Dins la variable ***order_items*** es poden trobar plats com: ous fregits, magdalenes, cereals, cafè, salmó, amanida, hamburgueses, patates fregides, pasta, pollastre i més.
L'estructura que segueix aquesta cadena de caracters es basa en primer el nom del plat i després la quantitat. Per exemple la comanda *[('Fries', 6), ('Salad', 4)]* conté 6 plats de patates i 4 amanides.

És molt important per evitar confusions parlar sobre l'economia d'Austràlia i la diferència amb Espanya perquè més endavant es veurà que en la variable ***order_price*** els valors dels preus de les comandes són molt elevades, una mitjana de 500 dòlars australians que serien uns 375 dòlars americans i uns 300 euros.

El cost de la vida australiana és molt més car que l'espanyola, però els sous també són més alts. Actualment, el salari mínim espanyol és menor als 1000 euros. En canvi, Australià té un salari mínim interprofessional de 3300 dòlars australians que serien una mica més de 2000 euros. La diferencia també es veu reflectida en el salari mitja anual espanyol que arriba als 24000 euros/any a diferencia dels 67200 dòlars australians/any. Per això els preus de les comandes d'aquesta base de dades són tan elevats.

Si un client té targeta de fidelitat, en la variable ***costumerHasloyalty.*** hi apareixerà un 1, se li aplicarà un descompte del 50% en les despeses de la tramesa.

2. Canvis en els formats

Després d'un breu anàlisi visual i tenint en compte la classe de cada variable s'han vist alguns errors en el format. Es fan els següents canvis:

a. Canvi del format de la variable ***date*** de *Character* a [*Data format* "yyyy-mm-dd"]
b. Canvi del format de la variable ***time*** de Character a [*Data format* "hh:mm:ss"]
c. Canvi de *Character* a factor en la variable ***order_type***.

d. Canvi del nom de la variable ***distance_to_customer_KM*** a ***distanceKM***
e. Eliminació del punt final de la variable ***costumerHasloyalty.*** que passa a ser ***costumerHasloyalty***
f. Canvi de dòlars australians a dòlars americans en les dues variables monetàries
g. Canvi de la variable de ***costumerHasloyalty*** Integer a Factor afegint l'etiqueta *Yes* = 1 i *NO* = 0
h. Canviar en les dues taules el format de la variable ***branch_code*** de Character a Factor i posar especial atenció a la variable de la taula "comandes" i corregir els errors.

Tots aquests errors estan explicats detalladament a l'annex amb el codi corresponent, només es comentaran alguns detalls d'aquests canvis:

- S'han utilitzat funcions de format com *as.factor, as.date, as.numeric* per la majoria de casos.
- S'ha escollit una relació del 0.75 entre el dòlar americà i l'australià
- Un problema que sorgeix en la variable ***branch_code*** és que hi ha hagut errors en la tipificació dels codis, ja que hi ha casos on el codi està en minúscula quan hauria d'estar en majúscules. Es pot veure en la taula 2.1:

bk	BK	ns	NS	tp	TP
11	147	12	160	9	161

TAULA 2.1: TIPUS DE CARÀCTERS DE LA VARIABLE "BRANCH_CODE"

Per això s'ha creat una funció que ha agrupat els valors que pertanyen al mateix restaurant i s'ha corregit l'error.

BK	NS	TP
158	172	170

TAULA 2.2: TIPUS DE CARÀCTERS DE LA VARIABLE "BRANCH_CODE" DESPRES DE LA MODIFICACIÓ

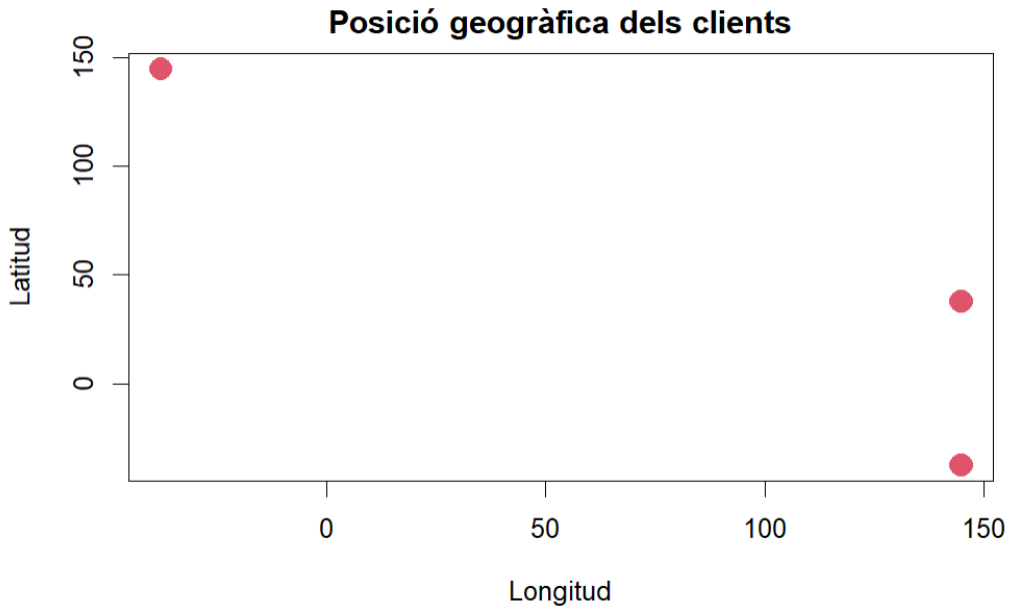
S'observa el canvi en la taula 2.2. Es realitzen quasi les mateixes comandes per cada branca.

3. Estudi de les latituds i longituds

Les coordenades centrals de geolocalització de Melbourne són **37.81753° N,144.96715°E**. Per tant, s'ha d'estudiar que les coordenades dels clients estiguin a prop d'aquestes i en cas contrari intentar identificar i corregir l'error. Es recorda que en tot moment s'ha de tenir present que…

$$Longitud = x \quad Domini: [-180,180]$$
$$Longitud = y \quad Domini: [-90,90]$$

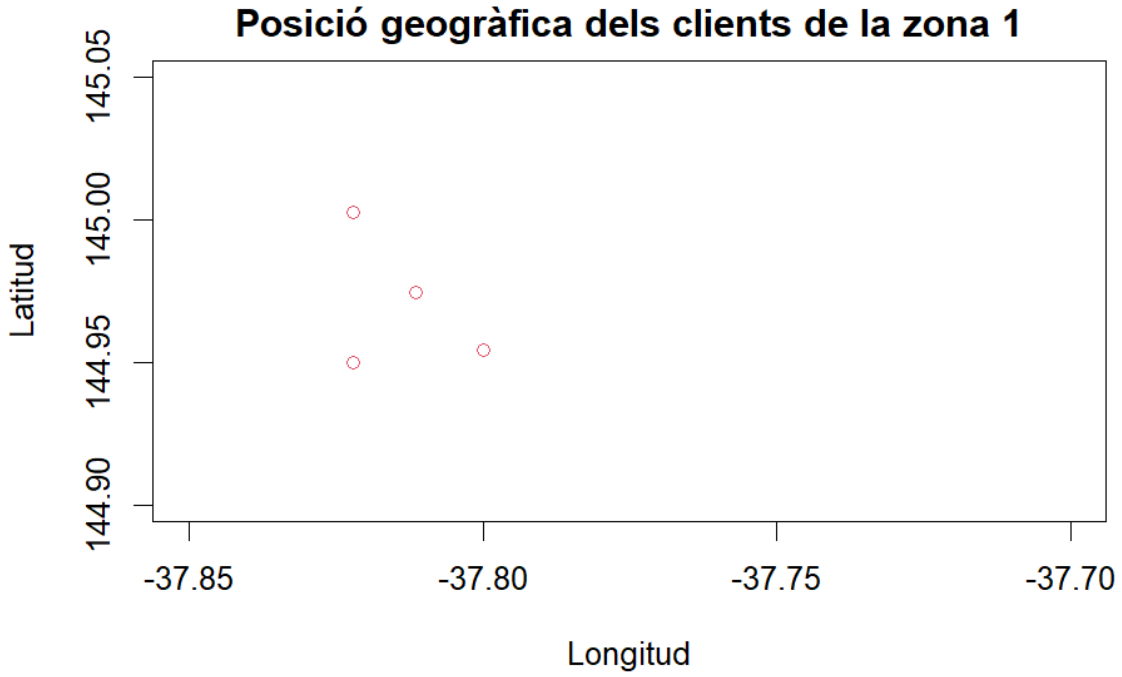
Es mira de manera gràfica quines zones geogràfiques abasten els clients.



GRÀFIC 3.1: POSICIÓ GEOGRÀFICA DELS CLIENTS ABANS DE CAP MODIFICACIÓ

Es pot apreciar a través del gràfic 3.1 que hi ha tres zones de possibles clients.

La primera zona, situada a la part superior esquerra de la gràfica, es regeix aproximadament dintre de les coordenades -37 longitud, 145 latitud. Aquesta zona no existeix en el planeta Terra, ja que se sap que la latitud no pot ser superior a 90.
Es realitza una gràfica per saber exactament quants *outliers* hi ha exactament dins d'aquesta zona.



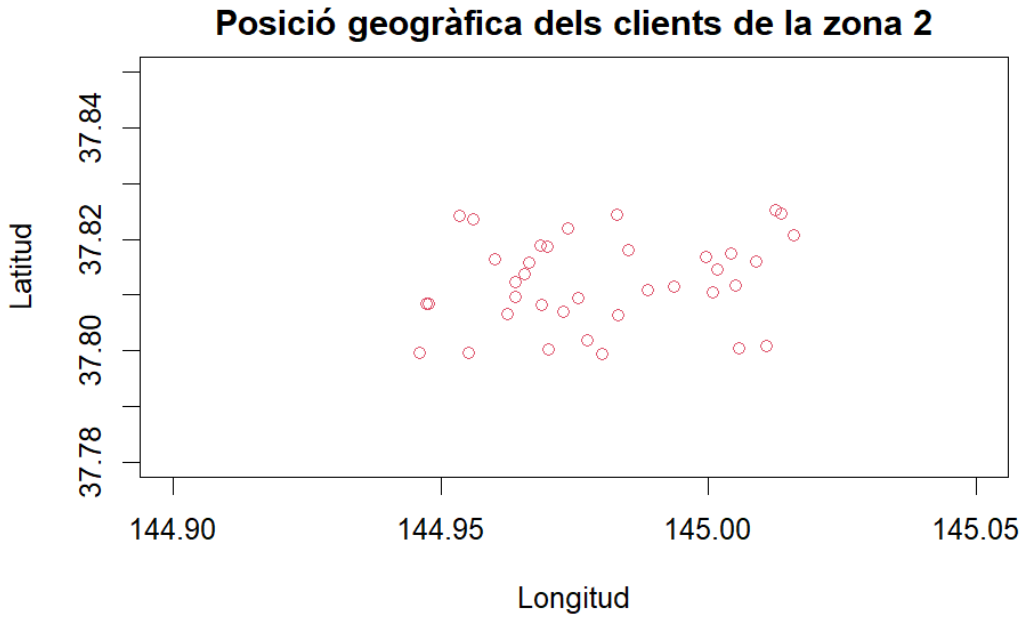
GRÀFIC 3.2: ESTUDI GEOGRÀFIC DE LA ZONA 1

Es poden visualitzar quatre *outliers* en el gràfic 3.2. En un principi, es considera la teoria que són clients que molt possiblement tenen les dades de longitud i latitud intercanviades i per això es deu l'error.

Una possible solució és identificar aquests 4 clients i intercanviar les dues columnes per tal que siguin posicions geogràfiques reals.
Una altra solució més conservadora seria eliminar aquestes observacions rebutjant la primera hipòtesi, però aquest treball es decanta per triar la primera opció perquè es creu lògica la possibilitat de variables intercanviades.

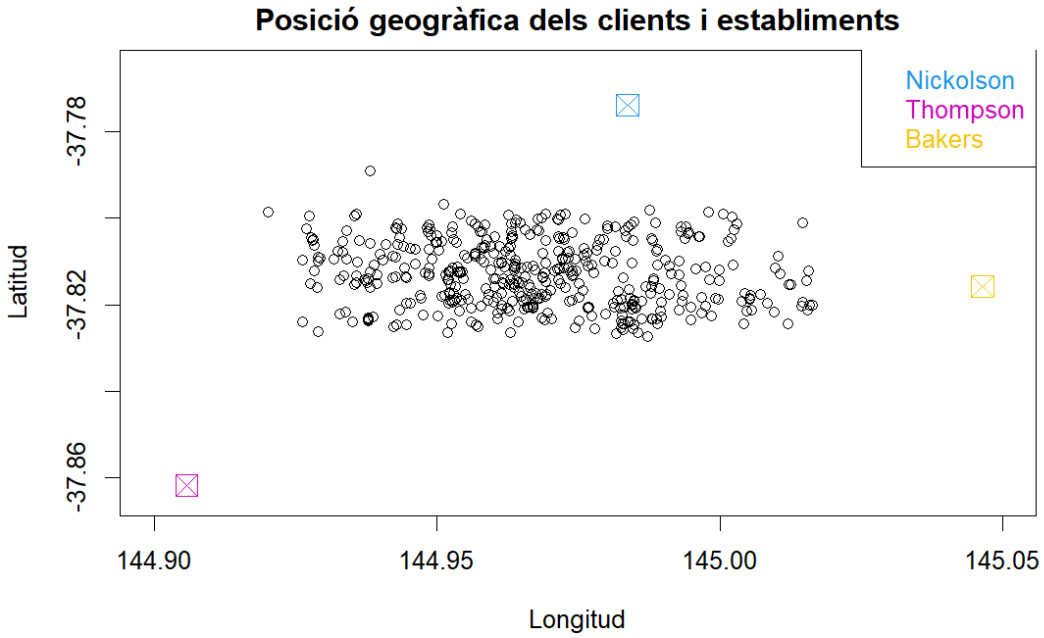
La segona zona, situada a la part central dreta, té coordenades aproximades de longitud 145 i latitud 37. No té gaire sentit que hi hagi clients en aquella regió perquè aquella zona és oceànica, a prop de les costes del Japó.
Per tant, apareix una altra teoria plausible, en aquest cas es creu que hi ha hagut un problema amb el signe de la latitud (probablement error humà haver-se oblidat el signe negatiu a l'introduir les dades).

Altrament, es podria considerar triar l'opció conservadora i eliminar aquests *outliers* però es creu fermament que l'error s'ha causat per aquest error en el signe.
També es procedeix a crear la gràfica per saber el nombre de punts estranys en aquesta zona.



GRÀFIC 3.3: ESTUDI GEOGRÀFIC DE LA ZONA 2

Com es pot observar en el gràfic 3.2, a diferència de la primera zona, aquest error s'ha propagat en nombroses comandes a clients, en total 37.
Es soluciona l'error identificant els clients d'aquesta zona i canviant el signe en la latitud.
Ara per acabar de confirmar la correcció dels errors es "mapegen" les posicions dels clients conjuntament amb les posicions dels tres establiments (veure gràfic 3.4).



GRÀFIC 3.4: MAPATGE DELS CLIENTS I ELS RESTAURANTS

Les posicions dels clients són força homogènies, formant un núvol de punts rectangular que es concentra en la part central i cèntric respecte els establiments. El restaurant Nickolson és el més proper als domicilis dels clients seguit de Bakers i finalment Thompson.

Es considera finalitzat l'estudi de les posicions geogràfiques i ara s'estudien els *missings*.

4. Estudi dels missings

Els *missings* o valors nuls (també anomenats *NA's*) són espais buits dins d'una base de dades on s'ha perdut informació d'una observació per diverses raons.

El procediment és establir quants i quins *missings* apareixen en aquesta base de dades i intentar estimar-los i/o eliminar-los en el pitjor dels casos.
Al realitzar una identificació de NA's a través de la funció *is.na()* es veu com només la variable ***date*** té valors perduts. Però sorgeix el problema que justament és una variable temporal que mostra el dia, mes i any de la comanda i normalment aquests valors perduts són difícils de tractar. L'opció més indicada és eliminar-los de la base de dades, eliminant així els 22 missings i deixant la base de dades amb 478 observacions.

5. Correcció de la variable order_type segons l'hora

Observant la base de dades s'ha vist que la variable ***order_type*** no funciona del tot bé, ja que no respecta l'horari que indica la variable ***time***.
Per exemple hi ha casos on s'estableix una etiqueta de "*dinner*" quan la hora de comanda és a les 11 del matí (4a observació de la taula 5.1).

time <chr>	order_type <chr>
15:16:03	Lunch
08:20:16	Breakfast
14:05:04	Lunch
11:43:05	Dinner

TAULA 5.1: EXEMPLES DE LES VARIABLES TIME I ORDER_TYPE

Es recorda que l'horari establert és el següent:

- "Breakfast": Horari de 8:00:00 a 12:00:00
- "Lunch": Horari de 12:00:01 a 16:00:00
- "Dinner": Horari de 16:00:01 a 20:00:00

Una vegada es sap l'horari s'ha de corregir la variable seguint els passos d'identificar els errors, substituir per les etiquetes correctes i comprovar que s'ha modificat d'acord amb el ordre establert.

Primer es mira si hi ha comandes en horaris diferents (entre les 20:00:00 i les 8:00:00) però no és el cas. Després es crea una funció que avalua totes les observacions i estudia si es compleixen les condicions de les etiquetes de la variable **order_type** i en cas contrari es canvia l'etiqueta per la correcta. Una vegada aplicada la funció s'ha vist com hi havia 37 casos on l'etiqueta era errònia i s'ha pogut corregir.

6. Comprovació de les comandes dins l'any 2018

S'ha d'assegurar el compliment que totes les comandes s'hagin realitzat l'any 2018 i eliminar aquelles observacions on no es compleixi aquesta condició en la variable ***date***.
Una vegada aplicat el filtre de l'any 2018 els resultats mostren que 15 observacions es situen fora de l'any d'estudi. Però al dur a terme l'output d'aquests anys el resultat ens pot deixar parats una mica.

```
"0010-04-20" "0004-04-20" "0005-07-20" "0011-04-20" "0006-09-20" "0001-12-20"
"0001-03-20" "0004-09-20" "0006-05-20" "0003-08-20" "0010-11-20" "0001-10-20"
"0010-08-20" "0011-05-20" "0011-07-20"
```

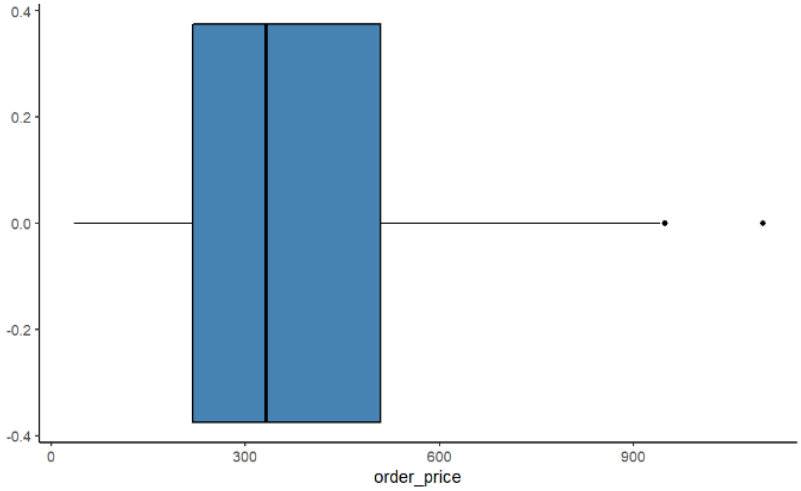
Les 15 observacions són *outliers*, errors d'introducció de dades a la base perquè és impossible que aquests anys siguin reals, ens situaria a l'època dels romans!
També podria donar-se el cas que es tractessin d'anys on els hi falta el 2 dels milers.
Sigui com sigui, cap d'aquests interessa per l'estudi i , per tant, es creu convenient que la millor opció és eliminar-los i per tant reduir la mostra fins a les 463 comandes.

7. Tractament *d'outliers* per variables numèriques

Es poden identificar *outliers* a través de la construcció de *boxplots*[5], gràfics que representen les dades numèriques a través dels seus quartils.

[5] Tots els *boxplots* de les variables numèriques es poden trobar en l'annex 2

Visualitzant el gràfic 7.1 es veu que en general no apareixen valors estranys, només hi ha un cas especial en la variable ***order_price*****,** ja que el preu d'una comanda era de 1100 dòlars.



GRÀFIC 7.1: BOXPLOT DE LA VARIABLE ORDER_PRICE

Es tracta d'un client que va demanar per sopar el dia 8 d'octubre i va encarregar una comanda gran: 10 plats de gambes (són un menjar normalment car), 10 plats de *Fish&Chips*, 9 plats de pasta i 8 de salmó (també un producte força car). Per tant, es creu que és un cas de comanda per un àpat per moltes persones i no s'ha de considerar valor atípic.

8. Eliminació de variables

En principi hi ha variables que no es donarà gens d'importància dins l'estudi. La comanda en si (saber exactament quina comanda s'ha fet de menjar) no interessa, ja que l'estudi es vol fixar en els trajectes fins al client més que saber què pot menjar-se aquest. Així que eliminar la variable ***order_items*** sembla una bona opció.

Capítol IV: ANÀLISI DESCRIPTIU GRÀFIC

Una vegada s'ha realitzat tota la part de tractament de la base de dades és hora d'analitzar d'una manera gràfica els clients i extreure informació rellevant.

Sobretot interessa dur a terme el mapatge dels clients agrupats per caps de setmana i dies entre setmana (variable que es crearà), àpats (variable ***order_type***) i vacances (també creada) entre d'altres. S'ha de tindre en compte que les dades processades s'han guardat en format *R.data* com a "*dataPRE.RData*" i "*branches.RData"*.
S'utilitzarà uns gràfics construïts a través de la llibreria ***ggplot2*** que permet treure més rendiment que graficant de forma estàndard.

1. Mesures descriptives bàsiques

Abans de començar amb els gràfics s'introdueix una breu anàlisi descriptiu (taula 1.1) que mostra mesures estadístiques com la mitjana, la mediana i la desviació típica entre altres i que serveix per fer-se una idea de la distribució de cada variable.

VARIABLES NUMÈRIQUES

	Min	1r Q	Mediana	Mitjana	3r Q	Max	Desv.
order_price	34.80	218.40	332.20	374.60	508.60	1099.10	204.16
customer_lat	-37.83	-37.82	-37.81	-37.81	-37.81	-37.79	0.01
customer_lon	144.90	145.00	145.00	145.00	145.0	145.01	0.02
distancekm	4.08	7.72	8.77	8.68	9.81	12.89	1.63
delivery_fee	3.16	9.54	10.49	10.44	11.56	16.58	1.90

TAULA 1.1: ANÀLISI DESCRIPTIU DE VARIABLES NUMÈRIQUES

També es realitza un breu anàlisi per les variables categòriques i variables temporals, taules 1.2, 1.3 1.4 i 1.5

VARIABLES CATEGÓRIQUES I DATES

	Min	1r Q	Mediana	Mitjana	3r Q	Max
date	2018-01-04	2018-04-07	2018-07-06	2018-07-05	2018-10-04	2018-12-31

TAULA 1.2: ANÀLISI DESCRIPTIU DE LA VARIABLE TEMPORAL "DATE"

	Breakfast	Dinner	Dinner
order_type	148	151	164

TAULA 1.3: ANÀLISIS DESCRIPTIU DE LA VARIABLE CATEGÒRICA "ORDER_TYPE"

	BK	NS	TP
branch_code	149	158	156

TAULA 1.4: ANÀLISI DESCRIPTIU DE LA VARIABLE CATEGÒRICA "BRANCH_CODE"

	No	Yes
customerHasLoyalty	407	56

TAULA 1.5: ANÀLISI DESCRIPTIU DE LA VARIABLE CATEGÒRICA "CUSTOMERHASLOYALTY"

- La mitjana del preu de les comandes és de 374.60 $, s'observa que aquesta variable té una gran desviació típica (204.15)
- La distancia mitjana que recorre la comanda és de 8.68 km, tenint una dispersió de 1.63
- Tant el tipus de comanda realitzada com el restaurant que s'encarrega de la comanda són variables balancejades, cada opció compta més o menys amb el mateix nombre d'individus. És una propietat positiva de cara a l'estudi. Així i tot cal remarcar que el sopar és l'àpat que més es demana i l'esmorzar el que menys. També es pot esmenar que la branca Nickolson és la que ha portat a cap més comandes durant l'any, seguida de prop per la Thompson i finalitzant amb la de Bakers
- Per altra banda, la variable de lleialtat està molt desbalancejada perquè només un 12% dels clients tenen la targeta de lleialtat

Una vegada finalitzada l'estadística bàsica numèrica s'entra al mapatge de clients segons les diferents variables categòriques proposades.

2. Gràfic segons caps de setmana i dies entre setmana

Interessa saber si hi ha diferències entre el nombre de comandes que es realitzen entre setmana i els caps de setmana. És lògic pensar que els caps de setmana es demanin més comandes perquè en general no es treballa i es vol descansar de la rutina en general.

El problema que es planteja és que no existeix la variable binària que indiqui els caps de setmana i s'ha de crear.

A partir de la funció ***wday*** de la llibreria ***lubridate***, s'identifiquen els dies de la setmana, de l'1 al 7 tot i que no es segueix l'ordre lògic tal que dilluns pren el valor 1, el dimarts el 2 i així successivament sinó que primer s'ha de realitzar una funció que identifiqui l'ordre.

A través d'un calendari s'identifica l'ordre i es treu que el valor 1 cau en dissabte, el 2 en diumenge, el 3 en dilluns…

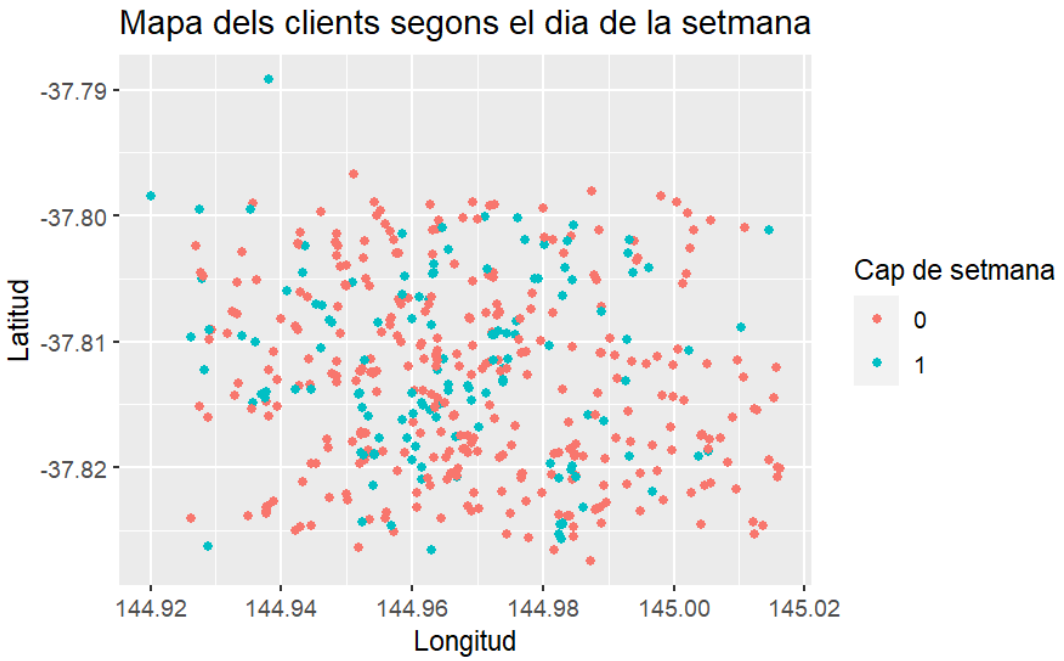
Ordre	1	2	3	4	5	6	7
Dia	dissabte	diumenge	dilluns	dimarts	dimecres	dijous	divendres
n.comandes	74	59	64	69	63	66	68

TAULA 2.1: COMPTATGE DE DIES DE LA SETMANA

El resultat de la taula dels comptatges (taula 2.1) no indica el resultat que s'esperava, ja que hi ha molta més homogeneïtat de la que en un primer moment es creia.

Per tant, hi ha 133 comandes realitzades el cap de setmana que representen el 29% de les comandes totals i 330 comandes fetes durant els dies entre setmana i són el 71% de les comandes.

A través d'una funció[6], es crea aquesta nova variable binària ***weekend*** que pren valor 0 si és entre setmana i 1 si la comanda es du a terme el cap de setmana. S'afegeix a la base de dades i es visualitza a través del gràfic 2.1.



GRÀFIC 2.1: MAPATGE DE CLIENTS SEGONS SI SÓN CAPS DE SETMANA O NO HO SÓN

La zona més central de la ciutat és la que porta a cap més comandes els caps de setmana, en canvi, la zona sud-est de Melbourne sobretot demana menjar els dies entre setmana.

3. Gràfic de clients segons dies festius o laborables

Un altre estudi interessant és veure si afecten les vacances al nombre de comandes realitzades. És raonable pensar que per vacances es vulgui cuinar menys i demanar més a domicili, però s'estudiarà per si de veritat passa.

A diferència d'Espanya, no es celebra la "Setmana Santa" sinó que existeixen altres tipus de celebracions festives. Així que es va contactar amb una família que havia estat vuit anys vivint allà per saber com estructuren les seves vacances sempre tenint en compte que estem parlant de l'any 2018 i es tracten de les vacances generals, no les de període escolar.
Per una banda, tenen el període nadalenc, que per posar-se en situació és com si fos l'agost d'Espanya. La majoria de gent fa vacances a partir de Nadal i fins a l'any nou, apart del *Christmas Day* el 25 de desembre ells celebren el *Boxing Day* l'endemà.

[6] El codi de la funció de creació de la variable ***weekend*** es troba en l'annex 3

Per altra banda, també tenen una espècie de setmana santa. Són festes de caràcter catòlic que es situen el 31 de març "*Saturday before Easter Sunday*", l'1 d'abril "*Easter Sunday*" i el 2 d'abril "*Easter Monday*". També hi ha altres dies solts que el govern de Melbourne concedeix.

Englobant totes les vacances es tenen els següents dies:

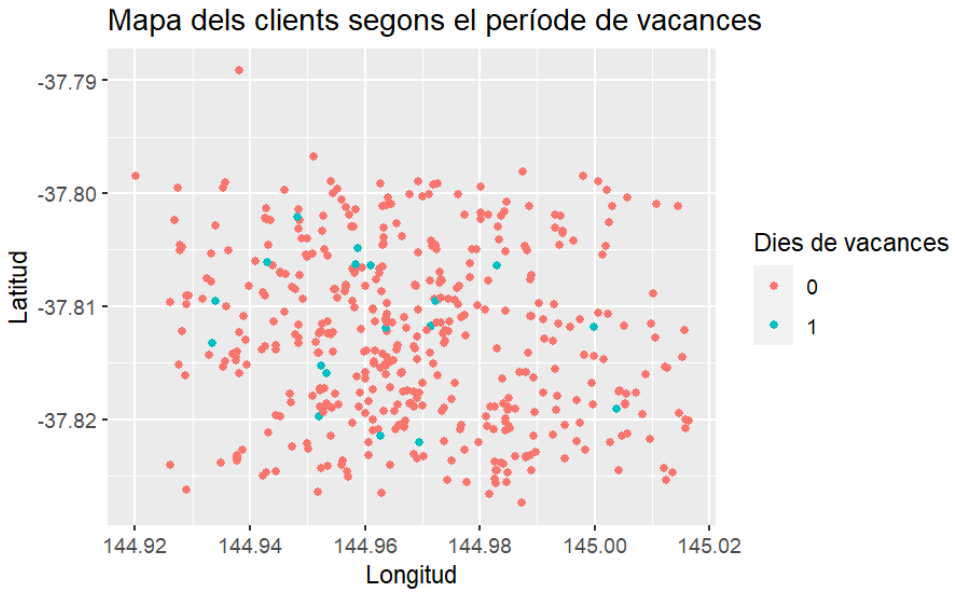
- **Període nadalenc**: Des del 24 de desembre fins a l'1 de gener
- **Els dies *Easter***: Del 31 de gener al 2 d'abril
- **Dia d'Austràlia**: 26 de gener
- **Dia del treballador**: 12 de març
- **"*Good Friday*":** 30 d'abril
- **Dia d'Anzac**: 25 d'abril
- **L'aniversari de la Reina**: 11 de juny
- **El divendres abans de la gran final AFL**: 28 de setembre
- **Dia de la Copa Melbourne**: És la carrera a cavalls celebrada el 6 de novembre

Un cop realitzada la funció que crea la variable ***holyday***[7] que estableix el valor 1 si el dia de la comanda és festiu i 0 en cas contrari, s'identifiquen 20 en període de vacances i 443 en període laboral.

L'hipòtesis inicial que suposava un augment de les comandes en període de vacances no es pot afirmar,a ja que pel que respecta aquests clients, les comandes suposen un 4.3% del total de les comandes i durant l'any hi ha 18 dies de vacances respecte a els 365 dies de l'any (el que significa un 4.9%).

Finalitzant l'apartat, com s'ha fet prèviament, s'introdueix la nova variable a la base de dades i es visualitza (gràfic 3.1).

[7] El codi de la funció de creació de la variable ***holyday*** es troba en l'annex 4

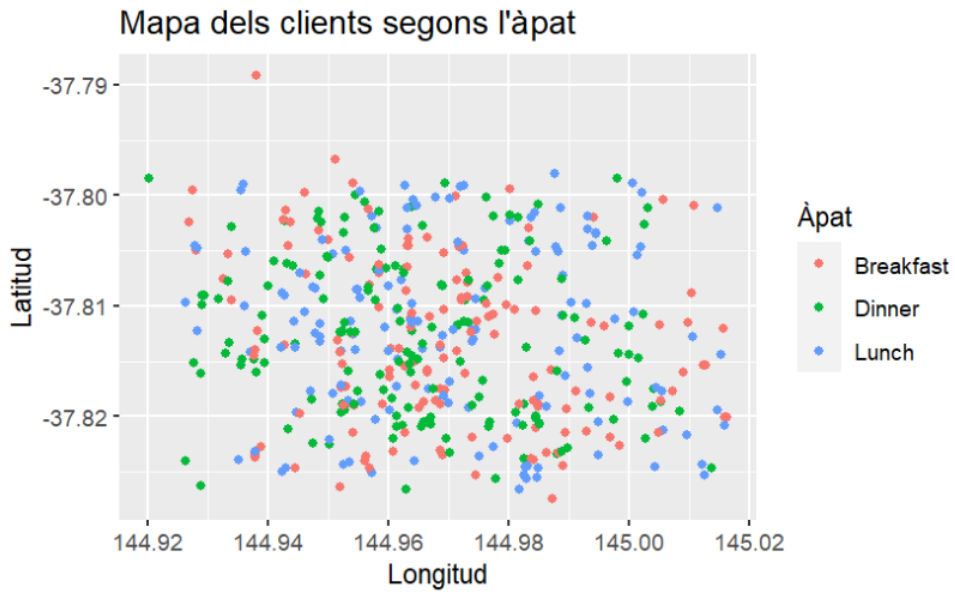


GRÀFIC 3.1: MAPATGE DELS CLIENTS SEGONS DIES LABORALS O FESTIUS

Les comandes realitzades en època de vacances no reflecteixen cap patró significatiu.

4. Gràfic de clients segons l'àpat

És un dels gràfics que pren més importància de cara a l'estudi, ja que pot donar una idea de quines zones de Melbourne l'empresa hi té més activitat al llarg del dia.



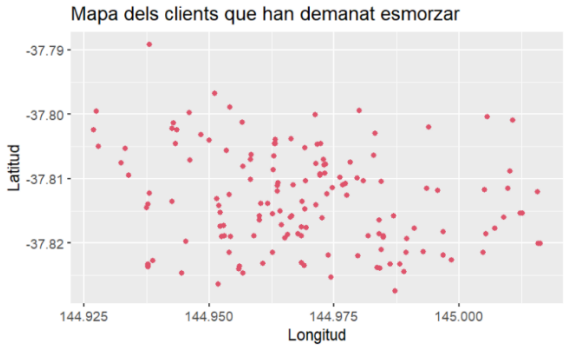
GRÀFIC 4.2: MAPATGE DE LES COMANDES SEGONS L'ÀPAT

Es pot veure en el gràfic 4.1 que hi ha bastanta equidispersió segons el tipus d'àpat. No hi ha cap zona de la ciutat on es demani més un àpat que un altre.

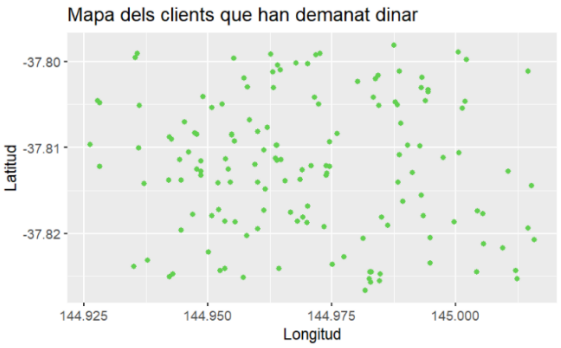
A més, el que es fa és estratificar la base de dades per aquesta variable i continuar el treball amb aquesta subdivisió. La taula **comandes** es dividirà en **comandes_break, comandes_lun** i **comandes_din**. Cadascuna té 13 variables que són les 11 originals i les

dues noves afegides. També és una manera de reduir el nombre de clients a fi de poder crear un model de cara a l'optimització dels trajectes.

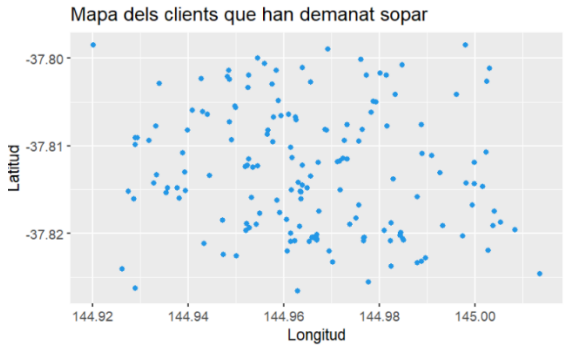
Les tres gràfiques estratificades són les següents:



GRÀFIC 4.3: MAPATGE DE LES COMANDES D'ESMORZAR



GRÀFIC 4.4: MAPATGE DE LES COMANDES DE DINAR



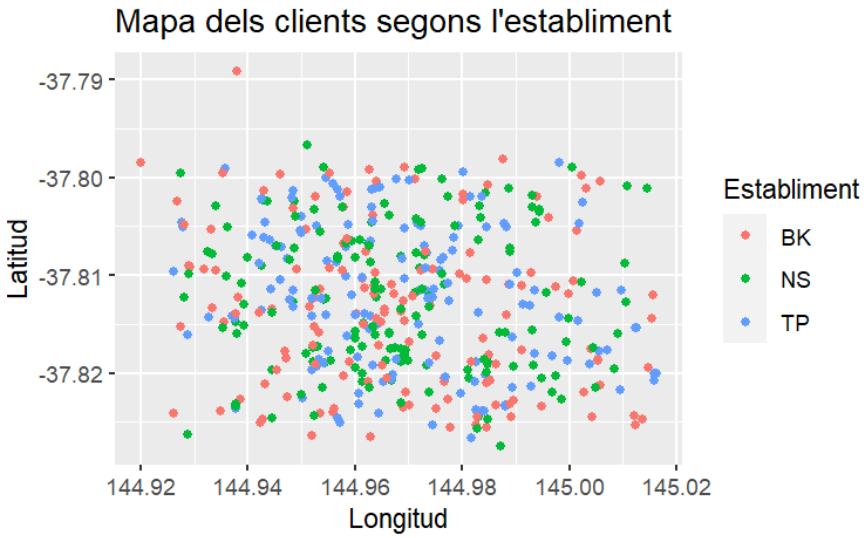
GRÀFIC 5.4: MAPATGE DE LES COMANDES DE SOPAR

Fent un anàlisi dels gràfics 4.2, 4.3 i 4.4 es pot dir que l'esmorzar es demana sobretot al centre i al sud de Melbourne CBD, però la regió més nòrdica no en demana. El dinar engloba totes les regions, fins i tot els voltants del centre de Melburne.
El sopar segueix la mateixa distribució que el dinar però més concentrat en una zona una mica més cèntrica i no tant pels voltants.

5. Gràfic de clients segons l'establiment de la comanda

És rellevant saber de quin establiment prové la comanda, no es tracta d'un problema on un repartidor d'un sol establiment reparteix només la comanda i torna sinó que pot anar variant d'establiment i així realitzar diverses comandes dels 3 restaurants diferents.

Si fos el primer cas, es tractaria de 3 problemes d'optimització separats amb un node inicial únic i el gràfic mostraria que els clients més a prop de cada un dels establiments rebria la comanda d'aquell mateix.



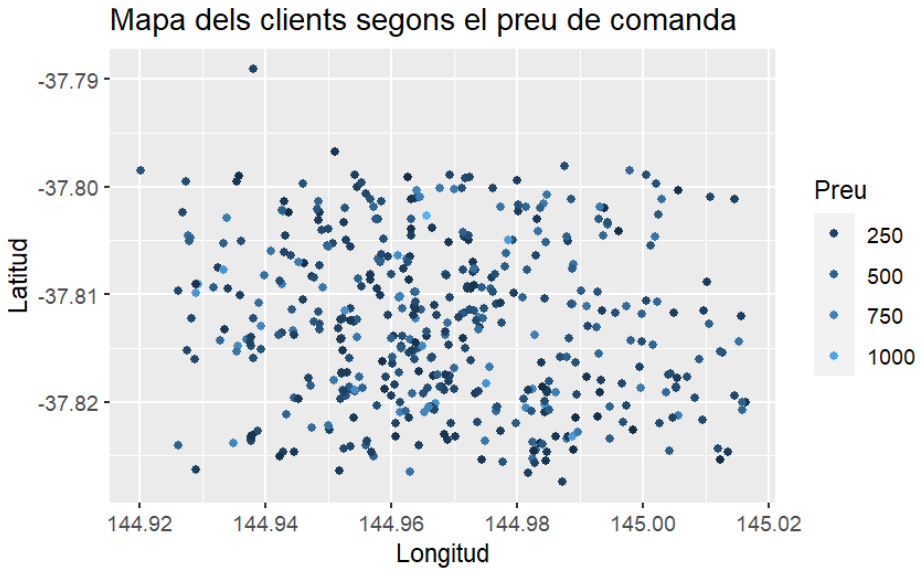
GRÀFIC 5.1: MAPATGE DELS CLIENTS SEGONS L'ESTABLIMENT DE LA COMANDA

El gràfic 5.1 mostra una equidispersió bastant notable. A tots els diferents llocs de la ciutat hi ha comandes provinents dels 3 establiments.

6. Gràfic de preu de comanda per client

Per finalitzar la part descriptiva gràfica es vol estudiar si hi ha relació entre la zona on viu el client i el preu de la comanda. En cas que es trobés un patró, es podria diferenciar zones de Melbourne CBD amb un nivell socioeconòmic més elevat que altres.

En tractar-se d'una variable numèrica el gràfic diferenciarà amb colors quatre nivells: el preu de la comanda va d'1 a 250 $, de 250 a 500 $, de 500 a 750 $ i de 750 a 1000 $.



GRÀFIC 6.1: MAPATGE DE CLIENTS SEGONS EL PREU DE LA COMANDA

A partir del gràfic 6.1 es pot dir que la part central de nord a sud hi ha més clients amb comandes més barates. Però aquest és dels únics patrons que es podria observar i tot i això no és prou convincent, en general és un gràfic que mostra pocs patrons en el preu.

Capítol V: RESULTATS

A continuació s'explicaran els quatre apartats que engloben els resultats; la divisió de la mostra, la visualització i validació de quatre escenaris, les limitacions d'aquesta optimització i les possibles extensions del problema.

1. Divisió de la mostra

Dur a terme una optimització de tota la mostra no té gaire sentit. En cap cas es voldria tindre que la ruta que exerceix un transportista, barregés les comandes de l'esmorzar amb el sopar perquè llavors de segur que no s'adaptaria a un problema real.

Per tal de realitzar escenaris amb sentit s'ha decidit dividir la mostra per àpats i per dues variables més.

1.1 Set 1: Àpats i Caps de setmana/Dies entre setmana

En aquest cas es creuen les variables ***order_type i weekend***. Cal recordar que ja es tenen les subbases estratificades per tipus d'àpat.

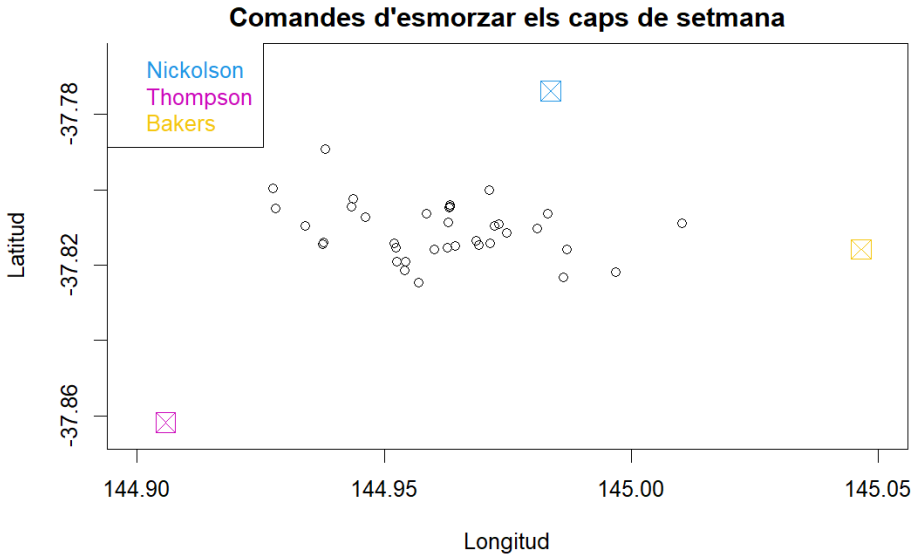
La taula 1.1.1 presenta les 6 divisions:

Nom de la taula	Descripció	Nº comandes
com_break_week	Les comandes per l'esmorzar que es realitzen entre setmana	111
com_break_end	Les comandes de l'esmorzar que es realitzen el cap de setmana	37
com_lun_week	Les comandes pel dinar que es realitzen durant la setmana	103
com_lun_end	Les comandes pel dinar que es realitzen durant el cap de setmana	48
com_din_week	Les comandes pel sopar que es realitzen durant la setmana	116
com_din_end	Les comandes pel sopar que es realitzen durant el cap de setmana	48

TAULA 1.1.1: DESCRIPCIÓ I COMPTATGE DE LES TAULES DEL SET 1

En total comprenen les 463 comandes fetes a l'any 2018. Es pot visualitzar alguna taula[8] creada com a exemple:

[8] Tots els gràfics de les taules tant del set 1 com del set 2 estan en l'annex 6



GRÀFIC 1.1.1: MAPATGE DE COMANDES D'ESMORZAR ELS CAPS DE SETMANAÇ

El gràfic 1.1.1 mostra els 37 clients que han demanat una comanda d'esmorzar en cap de setmana. Estan tots situats entre la part central i la nord-est del centre de Melbourne.

1.2 Set 2: Àpats i dies laborables/festius

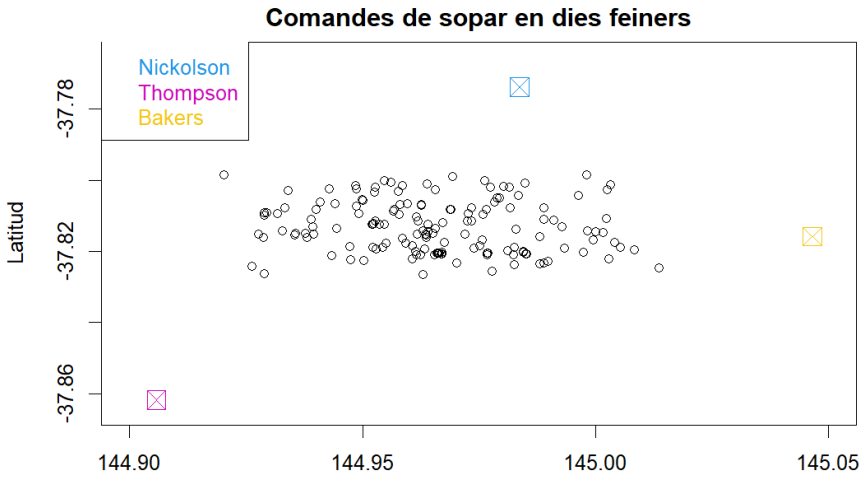
En el set 2 es creuen les variables ***order_type*** i ***holiday***. Les taules acabades en ***norm*** són les corresponents als dies laborables i les de ***holy*** els dies festius.

Les corresponents 6 divisions es mostren en la taula 1.2.1.

Nom de la taula	Descripció	Nº comandes
com_break_norm	Les comandes per l'esmorzar que es realitzen en dies laborables	140
com_break_holy	Les comandes de l'esmorzar que es realitzen en dies festius	8
com_lun_norm	Les comandes pel dinar que es realitzen en dies laborables	151
com_lun_holy	Les comandes pel dinar que es realitzen durant en dies festius	0
com_din_norm	Les comandes pel sopar que es realitzen durant dies laborables	156
com_din_holy	Les comandes pel sopar que es realitzen durant dies festius	8

TAULA 1.2.1: DESCRIPCIÓ I COMPTATGE DE LES TAULES DEL SET 2

La taula ***com_lun_holy*** està buida perquè no s'ha fet cap comanda de dinar durant els dies festius.



GRÀFIC 1.2.1: MAPATGE DE COMANDES DE SOPAR ELS DIES FEINERS

El gràfic 1.2.1 mostra com estan situats els 152 clients que han demanat sopar. En aquest cas estan tots bastant centrats i no s'observa cap patró significatiu.

2. Limitacions

Hi ha hagut dues limitacions principals relacionades amb l'optimització de la ruta; la formació de bucles i la poca memòria del *software* SAS. Això ha desencadenat que només s'hagin pogut veure escenaris amb pocs clients.

Un bucle es forma quan la ruta comença en un punt X i acaba en el mateix punt sense passar per tots els clients i restaurants.

L'optimització basada en la matriu binària necessita constriccions que restringeixin aquests bucles.
Fent memòria, a l'apartat d'optimització del capítol 2 es formula la restricció teòrica que elimina els bucles.

$$Bucle\{a_1 \in N, a_2 \in N, a_3 \in N, \dots, a_n \in N\}: \quad X_{a_1a_2} + X_{a_2a_3} + \cdots + X_{a_na_1} = n$$
$$on\ a_1 \neq\ a_2 \neq \dots \neq\ a_n$$

Aquesta restricció es pot aproximar amb la següent sèrie de restriccions que són les que s'han utilitzat al SAS:

$$ConectaTots\_1\{\ a_1\ in\ row, a_2\ in\ col\}: X_{a_1a_2} + X_{a_2a_1} \leq 1$$

$$ConectaTots\_2\{\ a_1\ in\ row, a_2\ in\ col, a_3\ in\ row\}: X_{a_1a_2} + X_{a_2a_3} + X_{a_3a_1} \leq 2$$

$$\vdots$$

$$ConectaTots\_\Omega\{\ a_1\ in\ row, a_2\ in\ col, \dots, a_\Omega\ in\ row,\}: X_{a_1a_2} + X_{a_2a_3} + \dots + X_{a_\Omega a_1} \leq \Omega$$

$$On$$

$$a_1 \neq a_2 \neq \ldots \neq a_\Omega$$
$$col, row \in N$$
$$N = 1,2,\ldots n$$
$$n = nombre\ de\ clients + nombre\ de\ restaurants$$

$$\Omega = \begin{cases} n + 1/2 \quad si\ \ n\ és\ impar \\ n/2\ si\ és\ par \end{cases}$$

Però això a la pràctica no sempre es pot aconseguir, ja que la formulació d'aquesta sèrie de constriccions tan grans genera problemes de memòria, amb el qual el *software* atura la seva execució i l'optimització no es completa.

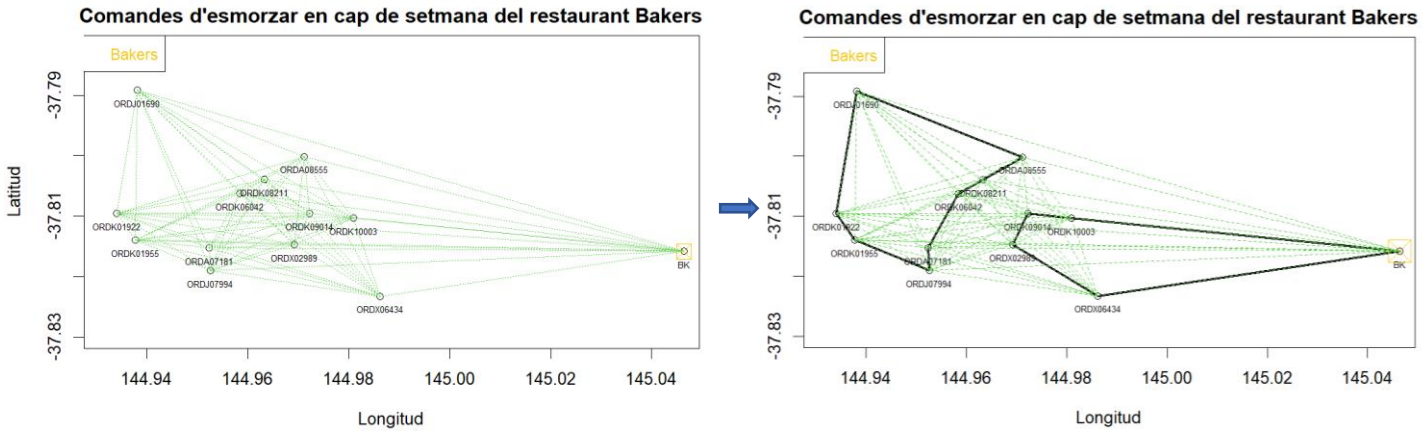
Per exemplificar-ho es pot visualitzar el següent exemple:

Aquest escenari es basa en les comandes d'esmorzar dues a terme en cap de setmana del restaurant Bakers. Són en total 12 clients els que en demanen i en el gràfic de sota es pot veure l'intent d'optimitzar la ruta. En un principi s'afegeixen les següents restriccions de bucle:

$$ConectaTots\_A\{\ i\ in\ row, j\ in\ col\}: X_{i,j} + X_{j,i} \leq 1$$

$$ConectaTots\_B\{\ i\ in\ row, j\ in\ col, k\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,i} \leq 2$$

$$ConectaTots\_C\{\ i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,i} \leq 3$$
$$\text{On } row, col \in \{1,2,\ldots 13\}$$



GRÀFICA 2.1.1: FORMACIÓ DE DOS BUCLES EN EL CAS DE COMANDES D'ESMORZAR EN CAP DE SETMANA DE BAKERS

Observant el gràfic 2.1.1, el subgràfic de l'esquerra mostra les possibles distàncies entre clients i entre el restaurant. En realitzar el procediment a través de SAS la solució resultant forma dos bucles. Un de quatre clients amb el restaurant i l'altre amb els vuit clients restants. Si se li afegeix una restricció de bucle més, el SAS para la minimització de la ruta i mostra el missatge que indica que s'ha quedat sense memòria i llavors no treu cap ruta.

Per això s'ha hagut de trobar escenaris amb pocs clients. Aquests escenaris es veuen a continuació.

3. Visualització i validació de la ruta òptima

S'han creat quatre escenaris on s'ha aplicat aquesta optimització i s'ha creat una ruta que comença en un restaurant, passa pels clients i torna al lloc d'inici.

En aquest apartat es visualitzen i validen cada un dels escenaris, especificant quines restriccions han sigut afegides per tal d'evitar la creació dels bucles.

Abans d'estudiar els escenaris, s'han de fer dos comentaris.
Una de les assumpcions del problema és que la ruta que segueix un transportista des d'un punt A fins a un B és la mateixa que des de B fins a A. Això implica que la ruta es pot realitzar en ambdós sentits (en sentit horari i antihorari), per tant, sempre hi ha dues interpretacions per cada escenari.

Per altra banda, la validació s'ha fet a través de dos mètodes; el mètode visual consisteix en la construcció del gràfic que marca la ruta seguida pel transportista i el mètode numèric és el que es pot veure en la següent funció:

```
punts<-ruta_opt[,c(1,2)]
cami<-c()
cont<-1
i<-nrow(punts)
inici<-punts$fila[i]
error<-FALSE
#es comença pel final perquè es on es situa el restaurant dins la matriu
while(cont<nrow(punts)){
      cami_i<-punts$columna[which(punts$fila==inici)]
      if(cont==1){
            cami<-c(cami,inici)
      }
      cami<-c(cami,cami_i)
      if(cont>1 && sum(duplicated(cami)>=1)){
            cont<-nrow(punts)
            print("La ruta no uneix tots els punts!")
            error<-TRUE
      }
      inici<-cami_i
       cont<-cont+1
      if(cont==nrow(punts) && error==FALSE){
            print("La ruta uneix tots els punts!")
      }
}
cami
```

FUNCIÓ 3.1: VALIDACIÓ NUMÈRICA DE LA RUTA

La funció 3.1 avalua el camí que fa la ruta i fa que aparegui un error si percep que ha passat dues vegades pel mateix client (amb aquest cas el missatge que treu és “La ruta no uneix tots els punts!”) i en cas contrari hi apareix el text “La ruta uneix tots els punts!”.

3.1 Escenari 1: Comandes del sopar en període de vacances del restaurant Thompson

El primer cas tracta de l’últim àpat del dia en període festiu. Són sis els clients que demanen sopar pel restaurant Thompson.

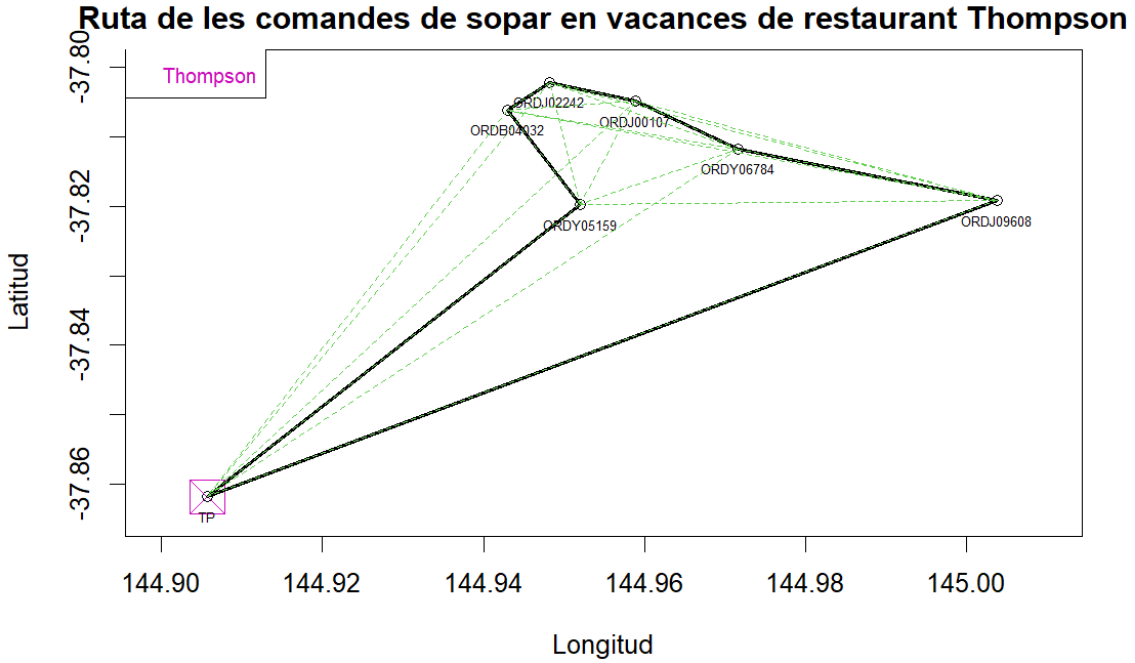
A part de les restriccions generals explicades en el capítol 2 apartat 2, s’han afegit les següents condicions per eliminar bucles.

$$ConectaTots\_A\{\, i \; in \; row, j \; in \; col\}: X_{i,j} + X_{j,i} \leq 1$$

$$ConectaTots\_B\{\, i \; in \; row, j \; in \; col, k \; in \; row\}: X_{i,j} + X_{j,k} + X_{k,i} \leq 2$$

$$\text{On } row, col \in \{1,2,\ldots 7\}$$

Una vegada s’executa el model a SAS i s’introdueix la solució a R es gràfica la ruta.



GRÀFIC 3.1.1: COMANDES DEL SOPAR EN PERÍODE DE VACANCES DEL RESTAURANT THOMPSON

La ruta òptima té un recorregut total de **23,690 km**. Visualitzant el gràfic 3.1.1 es pot confirmar que efectivament la ruta passa per tots els clients i torna al restaurant sense realitzar cap bucle.

Es pot interpretar com, en primer lloc, el transportista es dirigeix marcant una diagonal des del sur-oest fins al nord-est del centre de Melbourne fins al client més llunyà (ORDJ09608) i seguidament es dirigeix cap al centre de la ciutat marcant una semicircumferència i passant

pels cinc clients restants, sent ORDY05159 l'últim. Després de finalitzar totes les seves comandes es dirigeix de nou al restaurant.

3.2 *Escenari 2: Comandes del dinar en caps de setmana del restaurant Thompson*

En el segon escenari també hi participa el restaurant Thompson com a únic node inici, però es tracta de les comandes del dinar que s'han demanat nou clients el cap de setmana.

Les restriccions d'aquest escenari són les mateixes que l'anterior però afegint dues més.

$$ConectaTots\_A\{\, i\ in\ row, j\ in\ col\}: X_{i,j} + X_{j,i} \leq 1$$

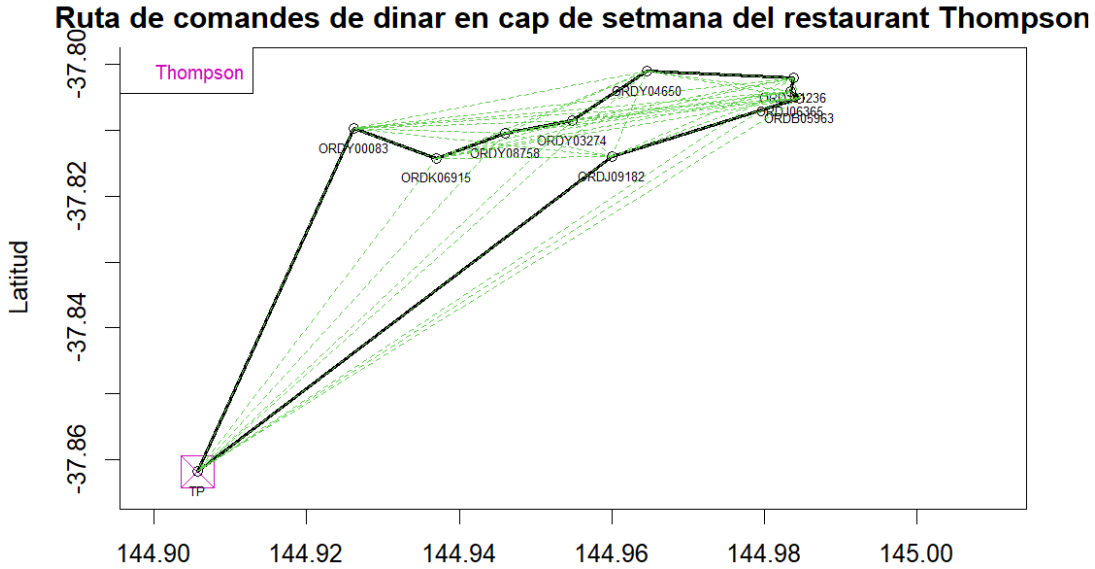
$$ConectaTots\_B\{\, i\ in\ row, j\ in\ col, k\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,i} \leq 2$$

$$ConectaTots\_C\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,i} \leq 3$$

$$ConectaTots\_D\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col, m\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,m} + X_{m,i} \leq 4$$

$$\text{On } row, col\ \in \{1,2,\ldots 10\}$$

S'executa l'optimització i es gràfica la ruta òptima.



GRÀFIC 3.2.1: COMANDES DEL DINAR

En aquest cas es fa un recorregut total de **21,643 km** des de la sortida de Thompson fins a l'arribada al mateix restaurant.

Interpretant el gràfic 3.2.1 es veu com el transportista es dirigeix marcant una diagonal des del sur oest fins a la zona central de Melbourne deixant la seva primera comanda al client ORDJ09182 i a continuació prossegueix la diagonal fins a arribar a una zona on hi ha 3 clients molt a prop l'un de l'altre. Després es va movent cap a l'oest duent a terme quatre comandes més (l'última és ORDY00083). Finalment, torna al restaurant completant la ruta.

3.3 Escenari 3: Comandes del sopar en vacances dels 3 restaurants

Aquest escenari és el més complet dels quatre, perquè realitza el transport de les comandes a través dels tres restaurants. La situació que es planteja és la de transportar el sopar a vuit clients en període festiu situats al centre de la ciutat.

En aquest cas s'introdueix un nou tipus de restricció, que elimina les situacions que el transportista va des d'un restaurant fins a un altre sense passar per cap client, ja que no tindria sentit aquest cas.

$$ConectaTots\_A\{\, i\ in\ row, j\ in\ col\}: X_{i,j} + X_{j,i} \leq 1$$

$$ConectaTots\_B\{\, i\ in\ row, j\ in\ col, k\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,i} \leq 2$$

$$ConectaTots\_C\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,i} \leq 3$$

$$ConectaTots\_D\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col, m\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,m} + X_{m,i} \leq 4$$

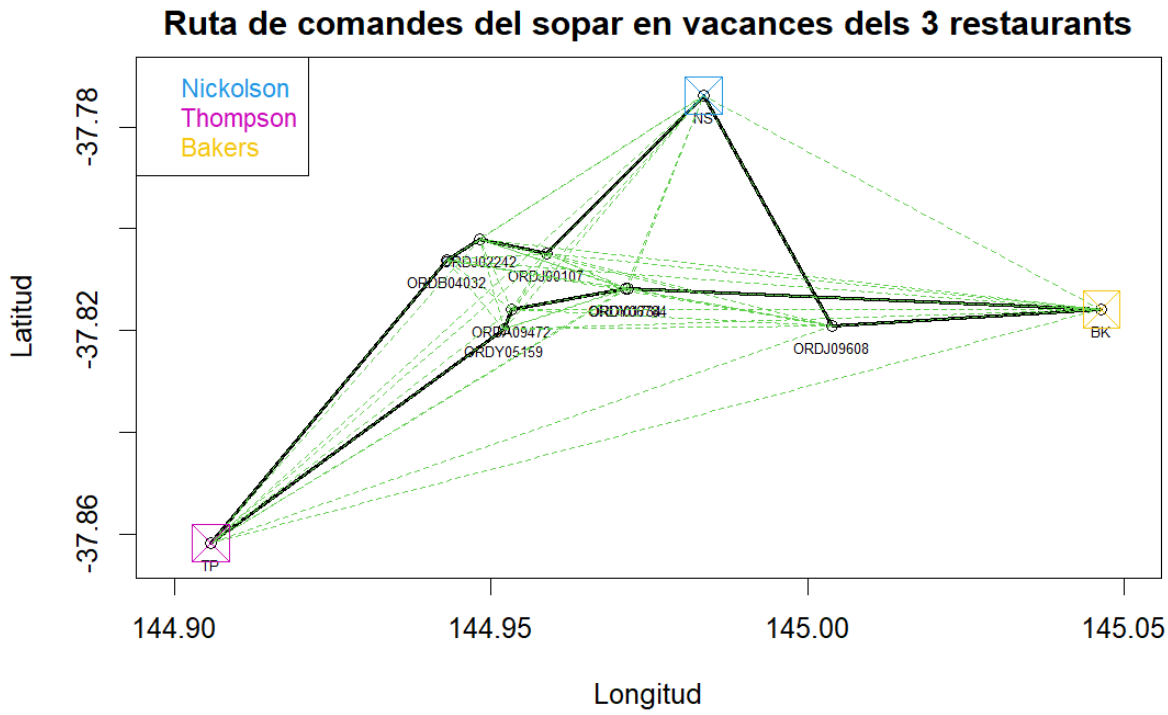
$$dosRestaurantsNo\_a\{\, i\ in\ row, j\ in\ col\}: X_{9,10} + X_{10,9} = 0$$

$$dosRestaurantsNo\_b\{\, i\ in\ row, j\ in\ col\}: X_{10,11} + X_{11,10} \leq 1$$

$$dosRestaurantsNo\_c\{\, i\ in\ row, j\ in\ col\}: X_{9,11} + X_{11,9} \leq 1$$

$$\text{On } row, col\ \in \{1,2,\ldots 11\}$$

S'executa el model amb aquestes restriccions i dins d'R es visualitza la ruta.



GRÀFIC 3.3.1: COMANDES DEL SOPAR EN PERÍODE DE VACANCES DELS TRES RESTAURANTS

En total, el recorregut és de **36,732 km** des del restaurant Nickolson[9] fins al retorn del mateix passant pels vuit clients i els altres dos restaurants.

La interpretació pot fer-se de la següent manera:

Visualitzant la ruta (vegeu el gràfic 3.3.1), s'observa com el transportista surt des de Nickolson amb tres comandes. Marcant una diagonal cap al sud oest passa per tres clients i acte seguidament es dirigeix al restaurant Thompson. En aquest restaurant recull quatre comandes i torna cap al centre on reparteix dues comandes properes entre si i dues comandes més en el mateix carrer (fins i tot podrien estar en el mateix edifici). Poc després es dirigeix al restaurant Bakers on agafa l'últim encàrrec situat una mica més a l'oest que on està (client ORDJ09608). Per últim, torna al restaurant Nickolson.

3.4 Escenari 4: Comandes d'esmorzar en període de vacances dels restaurants Bakers i Nickolson

L'últim escenari que es veurà tracta de l'únic àpat que encara no s'ha parlat, l'esmorzar. Els restaurants Bakers i Nickolson s'encarregaran de les vuit comandes fetes en període festiu.

Les restriccions d'aquest escenari són les següents:

$$ConectaTots\_A\{\, i \; in \; row, j \; in \; col\}: X_{i,j} + X_{j,i} \leq 1$$

[9] En aquest escenari la ruta pot començar tant de del restaurant Nickolson com des de Thompson com des de Bakers.

$$ConectaTots\_B\{\, i\ in\ row, j\ in\ col, k\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,i} \leq 2$$

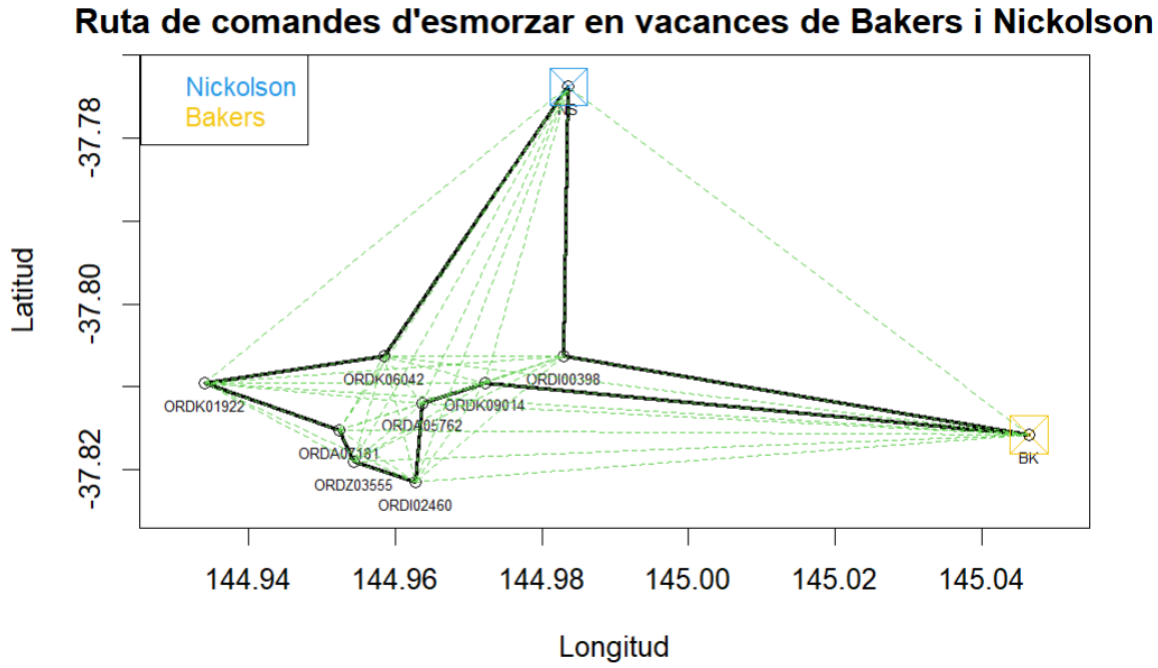
$$ConectaTots\_C\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,i} \leq 3$$

$$ConectaTots\_D\{\, i\ in\ row, j\ in\ col, k\ in\ row, l\ in\ col, m\ in\ row\}: X_{i,j} + X_{j,k} + X_{k,l} + X_{l,m} + X_{m,i} \leq 4$$

$$dosRestaurantsNo\_a\{\, i\ in\ row, j\ in\ col\}: X_{9,10} + X_{10,9} = 0$$

$$\text{On } row, col \in \{1,2,\ldots 10\}$$

Es visualitza el gràfic d'aquest model.



GRÀFIC 3.4.1: COMANDES D'ESMORZAR EN PERÍODE DE VACANCES DELS RESTAURANTS BAKERS I NICKOLSON

Aquesta ruta té una distància total de **27,138 km**.

Veient el gràfic 3.4.1 s'observa com el transportista surt des de Nickolson amb quasi totes les comandes (set de les vuit). Marcant una diagonal cap al sud oest passa per un client (ORDK06042) i després viatja més cap a l'oest fins a arribar a un client allunyat del centre de Melbourne (ORDK01922). Tot seguit, segueix adreçant-se cap al sud però amb direcció est passant per tres clients més. En aquest punt canvia la seva orientació i es dirigeix cap al nord i passa per dos clients més abans d'arribar al restaurant Bakers. D'aquest restaurant només agafa una comanda, on el client corresponent està situat al centre de la ciutat i finalment arriba al restaurant Nickolson per finalitzar el recorregut.

Capítol VI: CONCLUSIONS

Pel que respecta a les conclusions generals del treball esmenar que s'han aconseguit els objectius marcats del treball.
Primer de tot, s'ha tractat la base des de zero i processat de forma paramètrica. S'ha analitzat gràficament la base de dades per diferents factors i variables subdividint així la mostra. Com a propòsit principal del TFG, s'han optimitzat escenaris a partir del mètode d'optimització lineal amb restriccions basat en la matriu binària; tots i cada un dels escenaris mostra la ruta òptima i la longitud total del recorregut.

En segon punt important dir que el mètode d'optimització estudiat sembla útil per minimitzar tota mena de trajectes, sobretot aquells casos on hi hagi diferents recorreguts entre anada i tornada, sempre que es tingui un potent *software* que pugui modelitzar diferents situacions amb moltes variables. El plantejament d'arbre d'expansió màxima que hi té al darrere i la simplicitat de la funció objectiu facilita la comprensió i utilització del mètode.

En tercer lloc, el treball s'ha dut a terme sense tindre en compte les dates, només les hores. S'ha considerat que en un mateix dia es demanen tots els encàrrecs sense importar el dia exacte de la comanda, d'altra manera no s'hauria pogut minimitzar cap ruta perquè la majoria de vegades era una o dues comandes per dia.

Un punt negatiu que s'ha observat en el treball ha sigut que no s'ha pogut trobar una expressió general de restriccions que complís els requisits d'una ruta que passés per tots els clients i tornés al restaurant d'inici.

Com a punt futur, es podria repetir l'optimització però utilitzant algun altre software més potent com Maple, Matlab o ALGLIB i estudiar escenaris amb més clients i més restriccions.
Per altra banda, també es podrien afegir restriccions que s'adaptin a situacions reals d'aquest problema com per exemple una restricció que tingués en compte la grandària de la motxilla o bossa dels transportistes on es guarden les comandes durant la ruta (un transportista no pot portar més de cinc comandes alhora per exemple). També es podria considerar una constricció que restringís la quantitat de comandes que pot elaborar cada restaurant, el restaurant Bakers només pot crear vint àpats per emportar al dia i el restaurant Nickolson trenta. D'aquesta forma s'aconseguiria apropar-se més a la realitat.

BIBLIOGRAFIA

Articles

Aria, Massimo, and Corrado Cuccurullo. "bibliometrix: An R-tool for comprehensive science mapping analysis." *Journal of informetrics* 11.4 (2017): 959-975.

Chiu, Wei-Yu, Gary G. Yen, and Teng-Kuei Juan. "Minimum manhattan distance approach to multiple criteria decision making in multiobjective optimization problems." *IEEE Transactions on Evolutionary Computation* 20.6 (2016): 972-985.

Danielsson, Per-Erik. "Euclidean distance mapping." *Computer Graphics and image processing* 14.3 (1980): 227-248.

García, Salvador, et al. "Big data preprocessing: methods and prospects." *Big Data Analytics* 1.1 (2016): 1-22.

Held, Michael, Philip Wolfe, and Harlan P. Crowder. "Validation of subgradient optimization." *Mathematical programming* 6.1 (1974): 62-88.

Hughes, Edward P., and Trevor D. Kearney. "Optimization with the SAS® System: What It Is, What's New, and Why You Should Be Using It." (2003).

Lewis, Mark, and Fred Glover. "Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis." *Networks* 70.2 (2017): 79-97.

Pàgines web

Melbourne. (2022, 12 de junio). *Wikipedia, La enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Melbourne&oldid=144146934.

Melbourne Restaurant Food Delivery data | Kaggle
https://www.kaggle.com/datasets/jianhanma/melbourne-restaurant-food-delivery-data?select=edges.csv

Public & National Holidays in Melbourne (stepbystep.com)
https://www.stepbystep.com/public-national-holidays-in-melbourne-37710/#:~:text=Christmas%20is%20declared%20as%20a%20national%20holiday%20throughout,being%20celebrated%20in%20most%20of%20the%20commonwealth%20countries.

Árbol de expansión de peso mínimo - Algoritmo de Grafos (grapheverywhere.com)
https://www.grapheverywhere.com/arbol-de-expansion-de-peso-minimo/

Optimización binaria - Azure Quantum | Microsoft Docs
https://docs.microsoft.com/es-es/azure/quantum/optimization-binary-optimizationoft Docs

Wikipedia contributors. "Quadratic unconstrained binary optimization." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 27 Mar. 2022. Web. 22 Jun. 2022. https://en.wikipedia.org/w/index.php?title=Quadratic_unconstrained_binary_optimization&oldid=1079518221

Reading Raw Data: Reading Binary Data (sas.com):
https://support.sas.com/documentation/cdl/en/lrcon/62955/HTML/default/viewer.htm#a003209921.htm

Graphics in R with ggplot2 - Stats and R:
https://statsandr.com/blog/graphics-in-r-with-ggplot2

Detección y reemplazo de outliers con R - Adictos al trabajo:
https://www.adictosaltrabajo.com/2019/11/28/deteccion-y-reemplazo-de-outliers-con-r/

How to Treat Missing Values in Your Data - DataScienceCentral.com:
https://www.datasciencecentral.com/how-to-treat-missing-values-in-your-data-1/

Pairs with same Manhattan and Euclidean distance - GeeksforGeeks
https://statsandr.com/blog/graphics-in-r-with-ggplot2/

ANNEX

En aquest annex sobretot es vol mostrar el codi utilitzat per realitzar aquest treball, algunes funcions importants i alguns dels gràfics construïts.

Els últims cinc annexos són tots els scripts utilitzats durant tot el treball. Si es vol, també es poden consultar a través de Github per poder-los descargar i executar. Per poder-ho fer només cal accedir a https://github.com/marcelcanals/tfg.

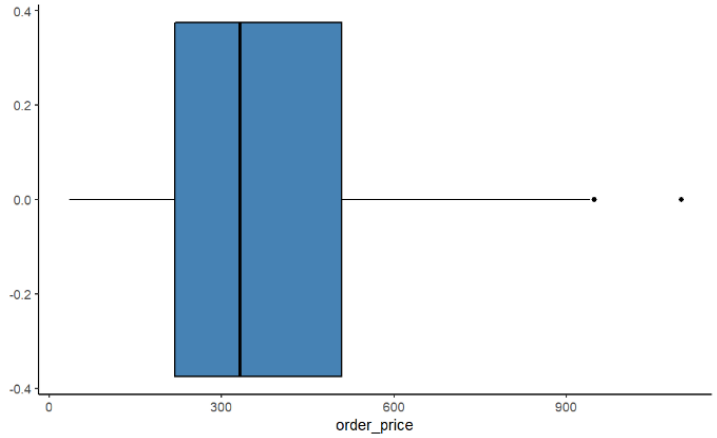
Annex 1: Taula resum: Paràmetres, conjunts i variables del model

Nom	**Figura, paràmetres, conjunts i variables**
Funció objectiu del model	$$min\ z\ = \sum_{i=1}^{N}\sum_{j=1}^{N} X_{i,j} * distancies_{i,j}$$ *Z = Longitud mínima de la ruta* $N\ \in \{1,2,\dots n\}$ *n= nº de clients + nº de restaurants del problema* *X = matriu binària*
Constriccions que restringeixen passar la ruta dues vegades pel mateix client	$$ClientCol_i: \sum_{j=1}^{N} X_{i,j} = 1$$ $$ClientRow_j: \sum_{i=1}^{N} X_{i,j} = 1$$ $N\ \in \{1,2,\dots n\}$ *n= nº de clients + nº de restaurants del problema* *X = matriu binària*
Constricció que anul·la els bucles dins la ruta	$$Bucle\{a_1 \in N, a_2 \in N, a_3 \in N, \dots, a_n \in N\}: \quad X_{a_1a_2} + X_{a_2a_3} + \dots + X_{a_{n-1}a_n} = n$$ $$on\ a_1 \neq\ a_2 \neq \dots \neq\ a_n$$ $N\ \in \{1,2,\dots n\}$ *n= nº de clients + nº de restaurants del problema* *X = matriu binària* *a = índex de cada una de les posicions de la matriu binària*

Annex 2: *Boxplots* de les variables numèriques

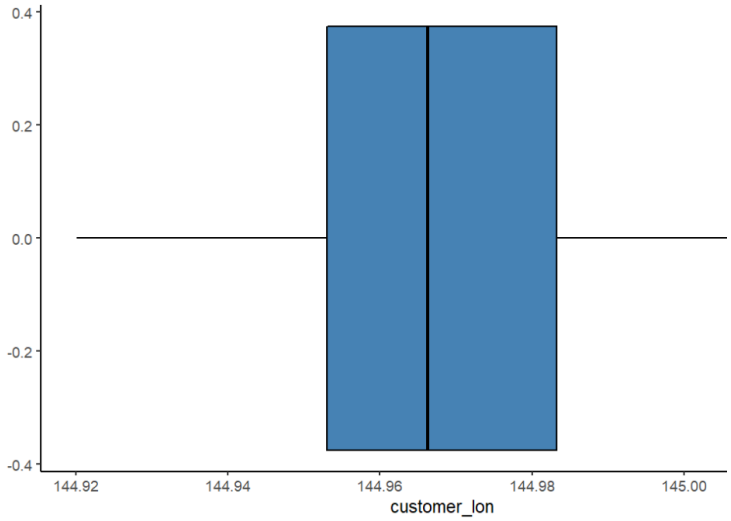
- Variable ***order_price***:

eix de les x: dòlars americans $ eix de les y: dispersió



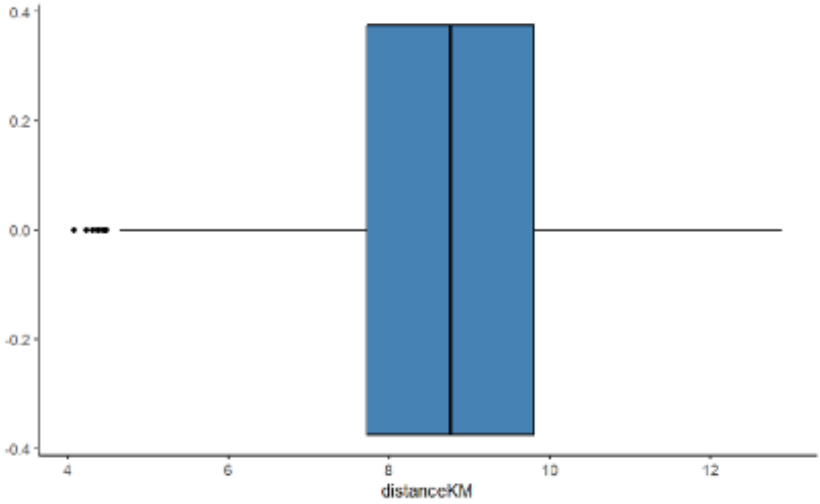
- Variable ***customer_lon***:

eix de les x: graus (º) eix de les y: dispersió



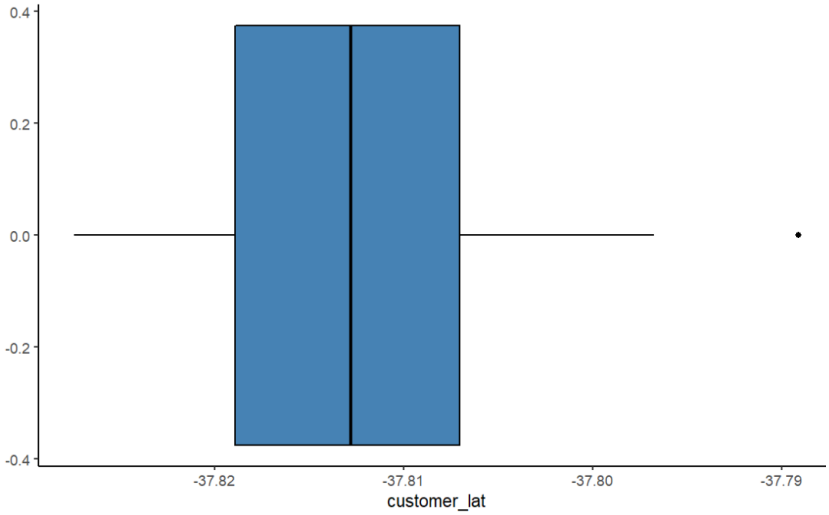
- Variable ***distanceKM***:

eix de les x: quilòmetres eix de les y: dispersió



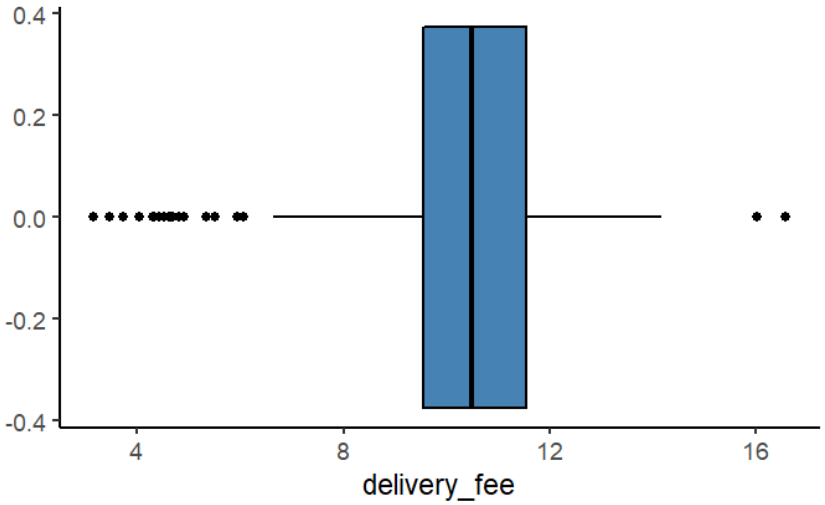
- Variable ***customer_lat***:

eix de les x: graus (º) eix de les y: dispersió



- Variable ***delivery_fee***:

eix de les x: dòlars americans $ eix de les y: dispersió



Annex 3: Funció per la creació de la variable "*weekend*"

```
library(lubridate)
index_setmana<-wday(comandes$date)
table(wday(comandes$date))
weekend<-c()
for( i in 1:length(index_setmana)){
  if(index_setmana[i]==1 || index_setmana[i]==2)
    weekend<-c(weekend,1)
  else{
    weekend<-c(weekend,0)
  }
}
table(weekend) #en efecte hem creat la variable binaria
comandes$weekend<-weekend
comandes$weekend<-as.factor(comandes$weekend)
```

Annex 4: Funció per la creació de la variable "*holyday*"

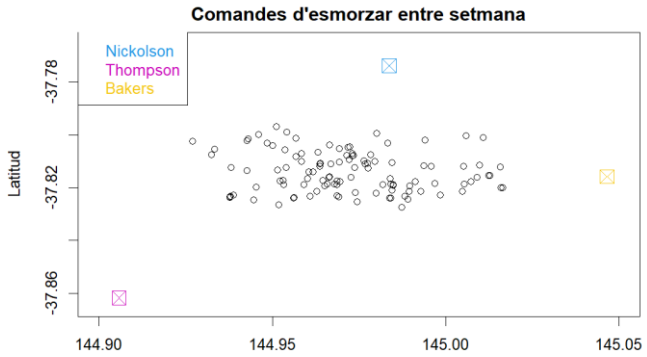
```
holy<-c("2018-01-01","2018-02-26","2018-03-12","2018-03-31","2018-04-
01","2018-04-02","2018-04-30" ,"2018-06-11","2018-09-28","2018-11-6","2018-
12-24","2018-12-25","2018-12-26","2018-12-27","2018-12-28","2018-12-
29","2018-12-30","2018-12-31")

holiday<-c()
date<-as.character.Date(comandes$date)
date[1]<-"2018-08-07" #el primer valor no el treu bé
j<-1
for(i in 1:length(comandes$date)){
  j<-1
    while(j<=length(holy)){
        if(date[i]==holy[j]){
          holiday<-c(holiday,1)
          j<-200
        }
        if(date[i]!=holy[j] && j==18){
          holiday<-c(holiday,0)
        }
      j<-j+1
    }
}
table(holiday)
comandes$holiday<-holiday
comandes$holiday<-as.factor(comandes$holiday)
```

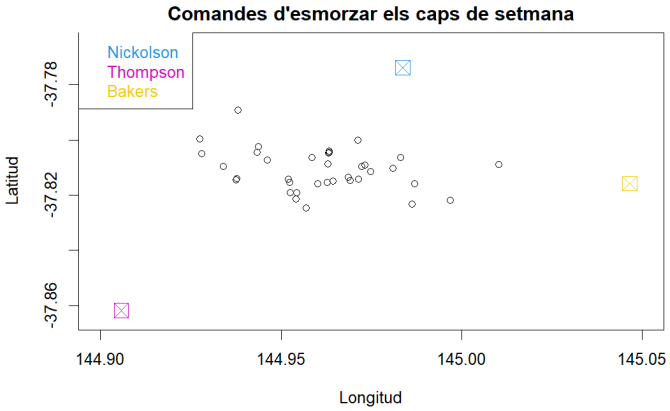
Annex 5: Funció del càlcul de distàncies entre nodes

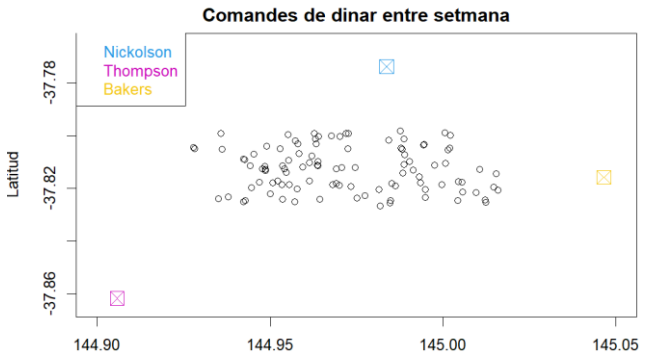
```
calc_Dist<- function(ddd){
  library(geosphere)
  attach(ddd)
  ## funció de distància
  vect<-c()
  for( i in 1:length(order_id)){
        for( j in 1:length(order_id)){
          p1_lon<-customer_lon[i]
          p1_lat<-customer_lat[i]
          p2_lon<-customer_lon[j]
          p2_lat<-customer_lat[j]

          a<-distGeo(p1=c(p1_lon,p1_lat),p2=c(p2_lon,p2_lat))
          vect<-c(vect,a)
        }
  }
return(vect)
```

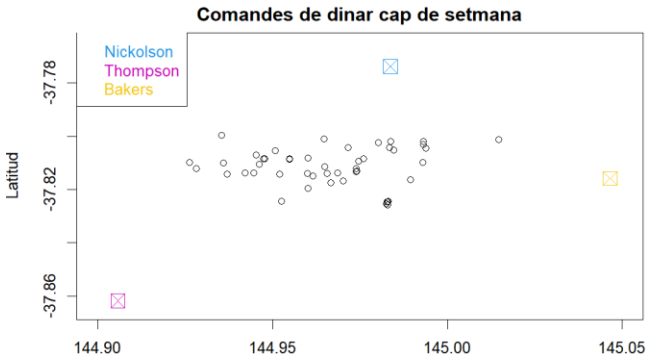
Annex 6: Gràfics dels sets 1 i 2

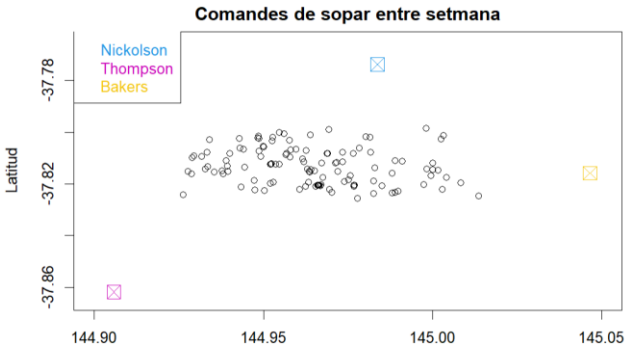
SET 1: Combinació d'Àpats vs Caps de setmana/Dies entre setmana

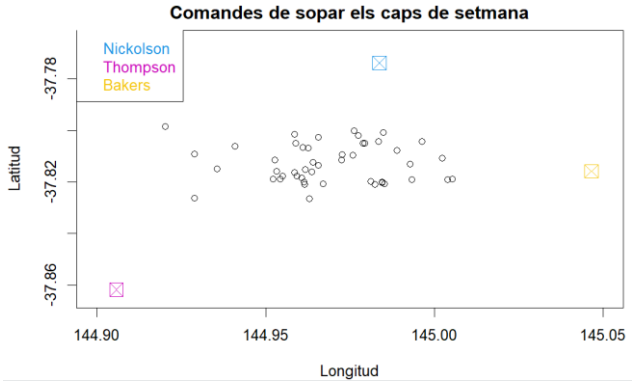




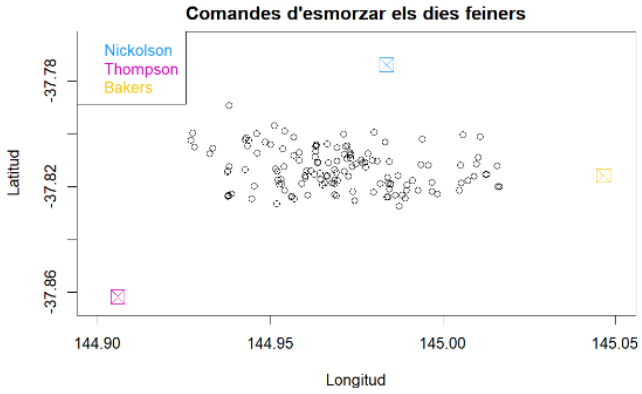


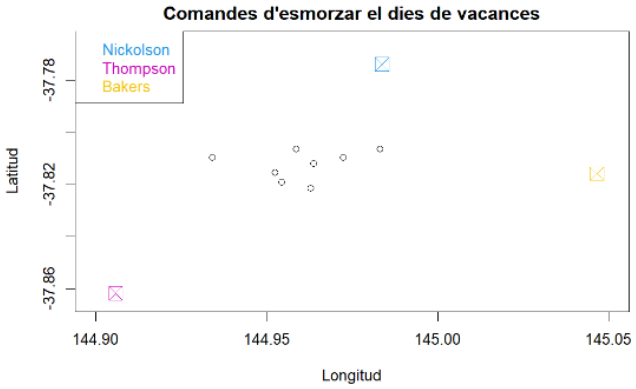


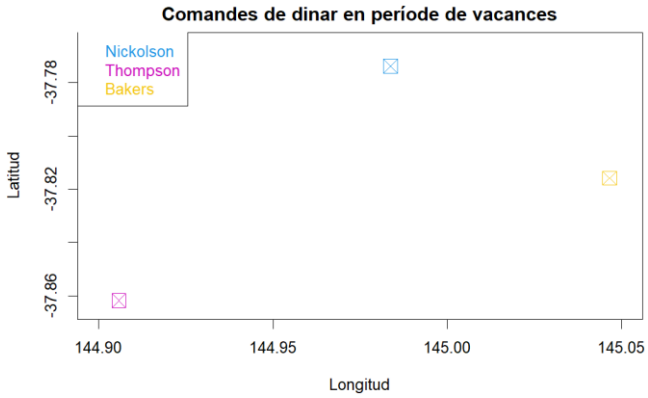


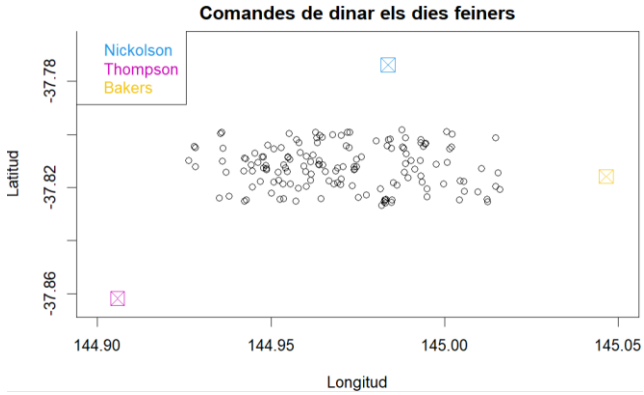


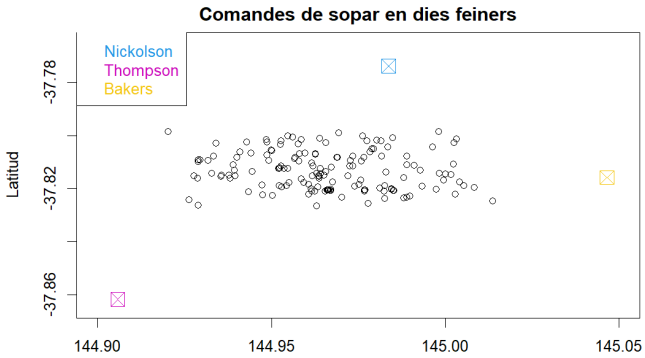
SET 2 Combinació d'Àpats vs Dies laborables/festius

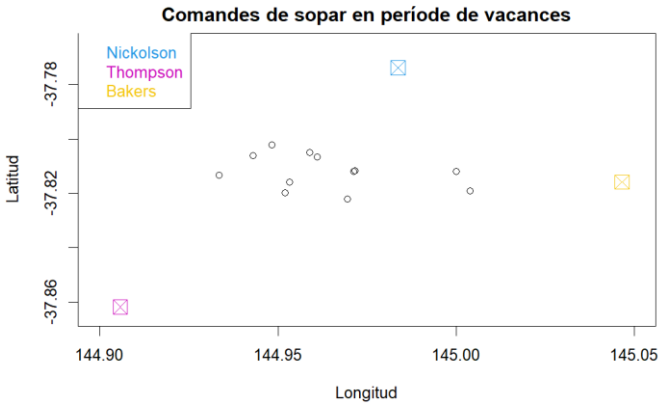












Hi ha quatre scripts escrits amb Rmd i un amb SAS. Els que pertanyen a R.Studio (Rmd) tenen el codi explícit en els requadres blaus (tots comencen per ```{r} i acaben amb ```) i els títols en gris.

Annex 7: Script del processament de les dades

```
---
title: "TFG, preprocesing de les dades"
author: "Marcel Canals"
date: "8/3/2022"
output:
  html_document:
    df_print: paged
---
# Preprocessament de la base de dades
```

La base de dades està composada per dues taules. Una primera anomenada "branches" i una segona taula amomenada "comandes" on hi ha tota la informació sobre cada una de les comandes.

Taula "branches"

Variables de la taula "branches"

- *branch_code*: [Character] Codi únic de cada branca (és la clau primaria que ens permetrà unir la aquesta taula amb la posterior){Clau Primaria}.
- *branch_name*: [Character] Nom complet de cada branca.
- *branch_lat*: [Numeric] Latitud de la branca en qüestió
- *branch_lon*: [Numeric] Longitud de la branca en qüestió

Importació de la taula "branches"

Parametrització del camí i de la base de dades:

```{r}
path0 <-"C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/"
file <- "branches.csv"
```

```{r}
branches <- read.csv(paste0(path0,file), header = TRUE)
```

Ja es té importada la primera taula. Breu anàlisis per veure com és:

```{r}
head(branches);dim(branches)
```

	branch_code <chr>	branch_name <chr>	branch_lat <dbl>	branch_lon <dbl>
1	NS	Nickolson	-37.77380	144.9836
2	TP	Thompson	-37.86183	144.9057
3	BK	Bakers	-37.81583	145.0464

Taula "comandes"

Aquesta taula en canvi, conté diverses variables importants i ens centrarem en l'estudi d'aquesta taula.

Variables de la taula "comandes"

- *order_id*: [Character] ID de la comanda {Clau Primaria}.
- *date*: [Character] Dia de la comanda.
- *time*: [Character] Hora exacta de la creació de la comanda.
- *order_type*: [Character] Tipus de menjar que es demana. Hi han tres tipus: "Breakfast" que sería l'esmorzar, "Lunch" el dinar i "Dinner" el sopar.
- *branch_code*: [Character] Codi de la branca. Variable que ens servira per unir aquesta taula amb la anterior.
- *order_items*: [Character] Cadena de caracters que mostren els plats de la comanda.
- *order_price*: [Numeric] Preu de cada comanda en dolars australians.
- *costumer_lat*: [Numeric] Latitud de la vivenda del client (coordenada Y, nort o sud)
- *costumer_lon*: [Numeric] Longitud de la vivenda del client (coordenada X, est o oest)
- *costumerHasloyalty.*: [Integer] Variable lógica que indica si un client té una targeta de fidelitat. En cas en que en tingui se li aplicarà un desconte del 50% en les despeses de la tramesa ("Gastos de envio").
- *distance_to_customer_KM*: Distància entre el client i una de les branques del restaurant.
- *delivery_fee*: Preu de l'enviament, de la tramesa

Importació de la taula "comandes"

Parametrització del camí i de la base de dades:

```{r, include = FALSE}
path <-"C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/"
file <- "dirty_data.csv"
```

```{r}
comandes <- read.csv(paste0(path,file), header = TRUE)
```

```
En aquest cas la taula "comandes" té 500 observacions i 12 variables, explicades anteriorment.
## Listat de les variables segons la seva classe
```{r}
LlistaVar <- sapply(comandes,class)
LlistaVar
```
```

```
                    order_id                        date
                 "character"                 "character"
                        time                  order_type
                 "character"                 "character"
                 branch_code                 order_items
                 "character"                 "character"
                 order_price                customer_lat
                   "numeric"                   "numeric"
                customer_lon          customerHasloyalty.
                   "numeric"                   "integer"
  distance_to_customer_KM                delivery_fee
                   "numeric"                   "numeric"
```
Es veuen alguns errors en els formats, també s'observa un punt final en la variable *customerHasloyalty.* i el nom de la variable *distance_to_customer_KM* es pot reduir considerablement.
```{r}
head(comandes)
```
```

| order_id | date | time | order_type | branch_code | order_items | order_price | customer_lat | customer_lon |
|---|---|---|---|---|---|---|---|---|
| ORDC01406 | 2018-08-07 | 15:16:03 | Lunch | NS | [('Fries', 6), ('Salad', 4)] | 140.80 | -37.81254 | 144.95412 |
| ORDZ10125 | 2018-01-12 | 08:20:16 | Breakfast | NS | [('Cereal', 8), ('Pancake', 6)] | 313.50 | -37.80931 | 144.97217 |
| ORDZ04175 | 2018-07-06 | 14:05:04 | Lunch | NS | [('Steak', 3), ('Salad', 1), ('Chicken', 6), ('Fries', 4), ('Burger', 8)] | 714.00 | -37.82048 | 144.99481 |
| ORDI03691 | 2018-04-26 | 11:43:05 | Dinner | NS | [('Pancake', 9), ('Eggs', 10), ('Cereal', 2)] | 480.25 | -37.81870 | 144.96577 |
| ORDZ04094 | 10-04-2018 | 11:12:40 | Breakfast | NS | [('Eggs', 5), ('Coffee', 3), ('Pancake', 9), ('Cereal', 7)] | 497.75 | -37.82061 | 144.96687 |
| ORDB10193 | 2018-10-15 | 17:27:53 | Dinner | TP | [('Fish&Chips', 5), ('Shrimp', 5), ('Salmon', 2), ('Pasta', 5)] | 664.50 | 37.80949 | 144.97574 |
| ORDX00958 | 2018-05-25 | 12:43:56 | Breakfast | BK | [('Salad', 2), ('Steak', 7), ('Chicken', 2)] | 413.40 | -37.79808 | 144.98754 |
| ORDI03630 | 2018-09-30 | 16:57:27 | Dinner | NS | [('Shrimp', 8), ('Fish&Chips', 6), ('Salmon', 4), ('Pasta', 7)] | 998.50 | 37.81224 | 144.96390 |
| ORDI05755 | 2018-03-07 | 10:01:41 | Breakfast | NS | [('Cereal', 1), ('Coffee', 7), ('Pancake', 3)] | 146.25 | -37.81914 | 144.98485 |
| ORDZ08573 | 2018-04-21 | 11:32:57 | Breakfast | NS | [('Eggs', 7), ('Cereal', 8), ('Pancake', 3), ('Coffee', 3)] | 417.25 | -37.81763 | 144.96941 |

| customerHasloyalty. | distance_to_customer_KM | delivery_fee |
|---|---|---|
| 1 | 8.335 | 13.700428 |
| 1 | 7.536 | 6.167473 |
| 0 | 9.860 | 15.088928 |
| 0 | 8.614 | 13.684368 |
| 0 | 8.802 | 13.760744 |
| 0 | 9.081 | 13.387529 |
| 0 | 6.412 | 12.007878 |
| 0 | 7.759 | 16.251402 |
| 0 | 8.996 | 14.114691 |
| 0 | 8.624 | 15.758749 |

### Canvis de formats
1. Canvi del format de la variable *date* de Character a [Data format "yyyy-mm-dd"]
2. Canvi del format de la variable *time* de Character a [Data format "hh:mm:ss"]
3. Canvi de Character a factor en la variable *order_type*.
4. Canvi del nom de la variable *distance_to_customer_KM* a *distanceKM*
5. Eliminació del punt final de la variable *costumerHasloyalty.* -> *costumerHasloyalty*
6. Canvi de dolars australians a dolars americans en les dues variables monetaries
7. Canvi de la variable de *costumerHasloyalty* Integer a Factor afegint la etiquieta Yes = 1 i NO = 0
8. Canviar en les dues taules el format de la variable *branch_code* de Character a Factor i posar especial atenció a la variable de la taula "comandes" i corregir els errors
## 1 i 2
```{r}
comandes$date <- as.Date(comandes$date)
#install.packages("lubridate")
library(lubridate) #llibreria que facilita els formats de dates temporals
comandes$time=as.POSIXct(comandes$time,format="%H:%M:%S")
```
```

3

```{r}
comandes$order_type<- as.factor(comandes$order_type)
```

4,5

```{r}
library(plyr)
comandes<-
rename(comandes,c(distance_to_customer_KM="distanceKM",customerHasloyalty.="customerHasloyalty"))
```

6

```{r}
comandes$order_price<-round(0.75*comandes$order_price,2)
comandes$delivery_fee<-round(0.75*comandes$delivery_fee,2)
```

7

```{r}
comandes$customerHasloyalty<-as.factor(comandes$customerHasloyalty)
levels(comandes$customerHasloyalty)<-c("No","Yes")
```

8

Un problema que sorgeix és que hi han hagut errors en la tipificació dels codis,
ja que hi ha casos on el codi està en minúscula quan hauria d'estar en majúscules.

```{r}
table(comandes$branch_code)
```

```
bk  BK  ns  NS  tp  TP
11 147  12 160   9 161
```

Per tant s'han de corregir les minúscules per majúscules i després factoritzar la variable en les dues taules.

```{r}
comandes$branch_code<-toupper(comandes$branch_code)
table(comandes$branch_code) #s'ha arreglat l'error
comandes$branch_code<-as.factor(comandes$branch_code)
branches$branch_code<-as.factor(branches$branch_code)
```

```
 BK  NS  TP
158 172 170
```

Una vegada acabada aquesta primera part es torna a fer un comptatge de les classes de les variables per veure si s'han canviat correctament

```{r}
str(comandes)
```

Estudi d'errors, missings i outliers

9. Estudi de les latituds i longituds correctes
10. Estudi dels missings en les diferents variables
11. Correcció de la variable *order_type* tenint en compte *date* (hi han casos on es dona una etiqueta de "lunch" quan la hora de comanda és a les 9 del matí)
12. Comprobacio que totes les comandes s'hagin realitzat l'any 2018
13. Tractament general d'outliers per variables numériques
14. Eliminació d'algunes variables que creiem sense importancia

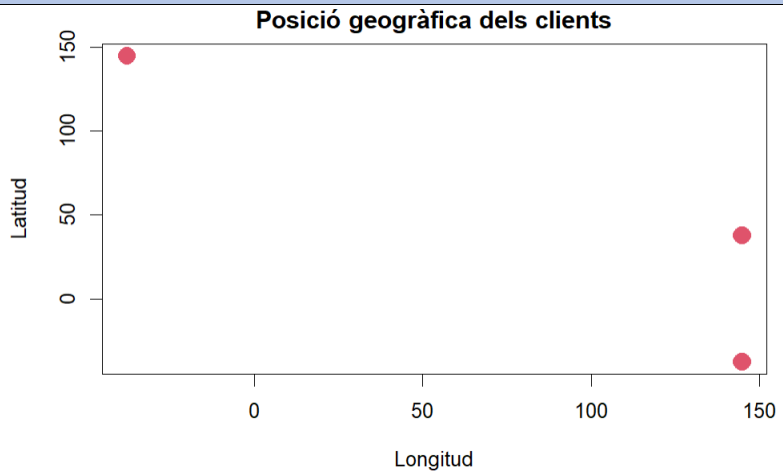
9

Les cordenades centrals de geolocalització de Melbourne són $-37,81753^{\circ} N$, $144,96715^{\circ} E$.

Per tant s'ha de comprobar que les coordenades dels clients estiguin aprop d'aquestes i en cas contrari intentar corregir l'error.
$$ Longitud=x \space\ \space\ \space\ Domini:[-180,180] \\ Latitud=y \space\ \space\ \space\ Domini:[-90,90]$$

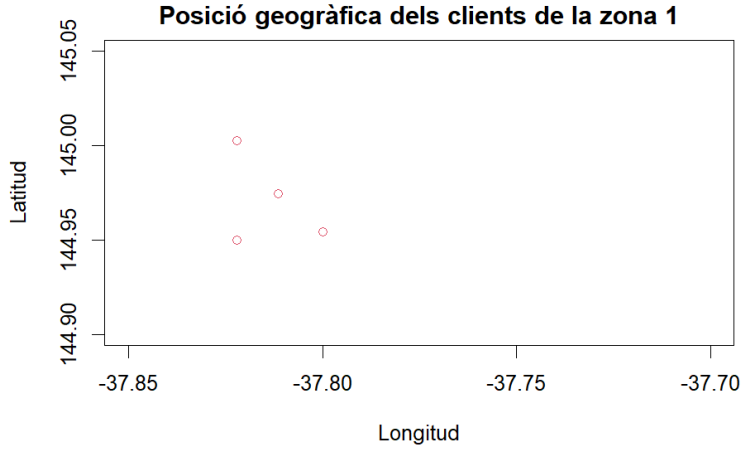
$$Longitud = x \quad Domini : [-180, 180]$$
$$Latitud = y \quad Domini : [-90, 90]$$

```{r}
plot(comandes$customer_lon,comandes$customer_lat,col=c(2),pch=c(16),cex=2,main="Posició geogràfica dels clients",xlab = "Longitud",ylab="Latitud")
```



Hi ha tres zones de possibles clients.
Es realitza la gràfica per saber exactament quants outliers hi han exactament dins d'aquesta primera zona.

```{r}
plot(comandes$customer_lon,comandes$customer_lat,col=c(2),xlim = c(-37.85,-37.70),ylim=c(144.9,145.05),main="Posició geogràfica dels clients de la zona 1",xlab = "Longitud",ylab="Latitud")
```



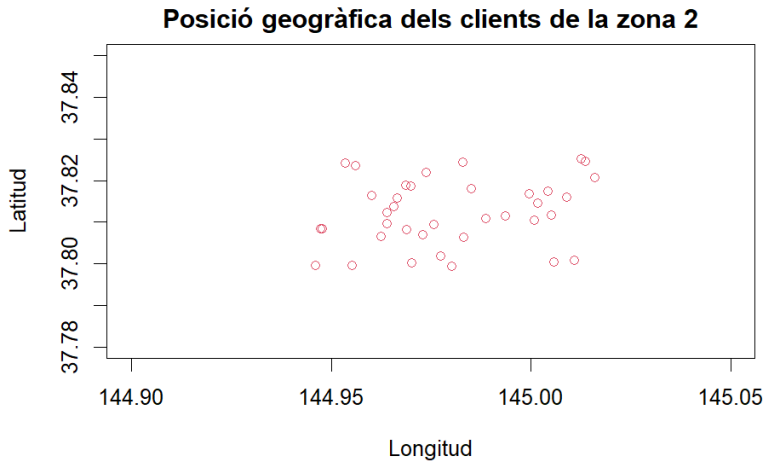
Una possible solució és identificar aquests 4 clients i intercambiar les dues columnes per tal de que siguin posicions geogràfiques reals.

```{r}
table(comandes$customer_lat>144)#efectivament son aquests 4
a<-which(comandes$customer_lat>144)#identifiquem les posicions d'aquests
a
lat_inc<-comandes$customer_lat[which(comandes$customer_lat>144)]#latituds incorrectes
lon_inc<-comandes$customer_lon[which(comandes$customer_lat>144)]#longituds incorrectes
comandes$customer_lat[a]<-lon_inc#correcio latitud
comandes$customer_lon[a]<-lat_inc#correcio longitud
```

```
table(comandes$customer_lat>144) #prova de que s'ha realitzat correctament el canvi
table(comandes$customer_lon>0) #segona prova per si acas
```

S'han arreglat els outliers de la primera zona!
Es visualitza la gràfica per saber el nombre d'outliers de la segona zona.

```{r}
plot(comandes$customer_lon,comandes$customer_lat,col=c(2),ylim =
c(37.78,37.85),xlim=c(144.9,145.05),main="Posició geogràfica dels clients de la zona 2",xlab =
"Longitud",ylab="Latitud")
```

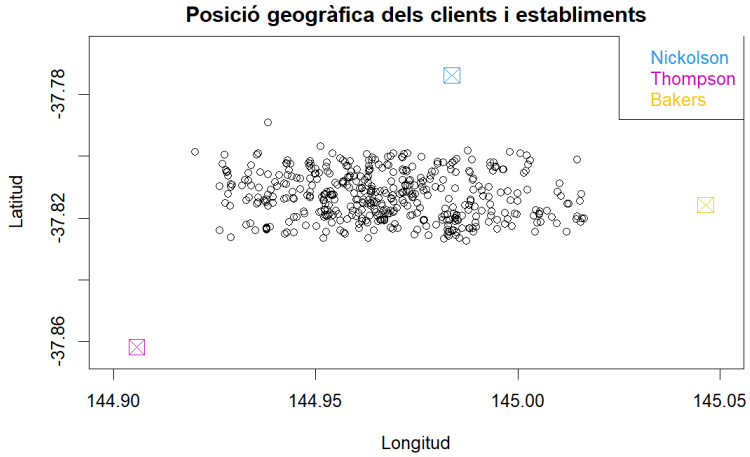


A diferència de la primera zona, aquest error s'ha propagat per nombrosos clients.
Es soluciona identificant els clients d'aquesta zona i canviant el signe.

```{r}
table(comandes$customer_lat>37)# hi han 37 clients amb l'error en la latitud
b<-which(comandes$customer_lat>37)#identifiquem les posicions d'aquests
comandes$customer_lat[b]<--comandes$customer_lat[b] #canvi de signe
table(comandes$customer_lon>0) #fem una prova per si acas
```

Ja s'han solucionat els outliers de la segona zona!
Ara per acabar de confirmar-ho es mapegen les posicions dels clients conjuntament amb les posicions dels establiments

```{r}
#plot(comandes$customer_lat,comandes$customer_lon,main="Posició geogràfica dels clients",xlab =
"Longitud",ylab="Latitud")
plot(comandes$customer_lon,comandes$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-
37.765),main="Posició geogràfica dels clients i establiments",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topright",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),
text.col = c(4,6,7))
```



Tot sembla tenir sentit i més correcte!

10.Tractament dels missings

```{r}
(ConteNA <- colSums(is.na(comandes))>0)
```

order_id	date	time	order_type
FALSE	TRUE	FALSE	FALSE
branch_code	order_items	order_price	customer_lat
FALSE	FALSE	FALSE	FALSE
customer_lon	customerHasloyalty	distanceKM	delivery_fee
FALSE	FALSE	FALSE	FALSE

```{r}
table(is.na(comandes$date)) #hi han 22 missings
which(is.na(comandes$date)==TRUE) #els hem identificat
```

Els eliminem de la base de dades

```{r}
comandes<-comandes[-which(is.na(comandes$date)==TRUE),]
```

11 Correcció de la variable *order_type*

```{r}
head(comandes[,c("time","order_type")],20)
```

	time <S3: POSIXct>	order_type <fctr>
1	2022-03-31 15:16:03	Lunch
2	2022-03-31 08:20:16	Breakfast
3	2022-03-31 14:05:04	Lunch
4	2022-03-31 11:43:05	Dinner
5	2022-03-31 11:12:40	Breakfast
6	2022-03-31 17:27:53	Dinner
7	2022-03-31 12:43:56	Breakfast
8	2022-03-31 16:57:27	Dinner
9	2022-03-31 10:01:41	Breakfast
10	2022-03-31 11:32:57	Breakfast

Primer s'ha de mirar si hi han comandes en horaris diferents (entre les 20:00:00 i les 8:00:00)

```{r,warning=FALSE}
library(hms)
time<-format(comandes$time,format="%H:%M:%S")
table(time>"20:00:00")#no hi ha cap comanda més d'hora que el horari establert
table(time<"08:00:00")#no  hi ha cap comanda més tard que el horari establert
```

#Funció que corregeix l'etiqueta de *order_type*

```{r}
comandes$time<-format(comandes$time, format = "%H:%M:%S")#tranf a character per operar
```

```
cont<-0
for( i in 1:length(comandes$time)){
  if(comandes$order_type[i]=="Breakfast"){
       if(comandes$time[i]>="08:00:00" && comandes$time[i]<="12:00:00"){
         cont<-cont
       }else if(comandes$time[i]>="12:00:01" && comandes$time[i]<="16:00:00"){
         cont<-cont +1
         comandes$order_type[i]<-"Lunch"
       }else{
         cont<-cont +1
         comandes$order_type[i]<-"Dinner"
       }
  }else if(comandes$order_type[i]=="Lunch"){
       if(comandes$time[i]>="08:00:00" && comandes$time[i]<="12:00:00"){
         cont<-cont +1
         comandes$order_type[i]<-"Breakfast"
       }else if(comandes$time[i]>="12:00:01" && comandes$time[i]<="16:00:00"){
         cont<-cont
       }else{
         cont<-cont +1
         comandes$order_type[i]<-"Dinner"
       }
  }else{
      if(comandes$time[i]>="08:00:00" && comandes$time[i]<="12:00:00"){
       cont<-cont +1
       comandes$order_type[i]<-"Breakfast"
      }else if(comandes$time[i]>="12:00:01" && comandes$time[i]<="16:00:00"){
       cont<-cont +1
       comandes$order_type[i]<-"Lunch"
      }else{
       cont<-cont
      }
  }
}
cont
head(comandes[,c("time","order_type")],20)
```

Hi havien 37 casos on l'etiqueta de l'ordre de comanda estava malament. Amb la vista anterior es pot veure que no hi ha cap error.

12. Comprobacio que totes les comandes s'hagin realitzat l'any 2018

```{r}
table(comandes$date < "2018-01-01");table(comandes$date > "2018-12-31")
```

Aquest codi ens informa de que hi ha 15 observacions on la data és inferior a l'any 2018.
Anem a aprofunditzar més per veure quines dates són exactament.

```{r}
#which(comandes$date < "2018-01-01") ens dona le posicions
comandes$date[which(comandes$date < "2018-01-01")]
```

```
"0010-04-20" "0004-04-20" "0005-07-20" "0011-04-20" "0006-09-20" "0001-12-20"
"0001-03-20" "0004-09-20" "0006-05-20" "0003-08-20" "0010-11-20" "0001-10-20"
"0010-08-20" "0011-05-20" "0011-07-20"
```

La millor opció és eliminar-los.

```{r}
comandes<-comandes[-which(comandes$date < "2018-01-01"),] #eliminem tots 15
```

13. Tractament general d'outliers per variables numériques

BOXPLOTS PER LES VARIABLES NUMÉRIQUES

```{r,out.width="50%"}
library(ggplot2)
```

```
ListaVar <- sapply(comandes,class)
VarInt <- which(ListaVar == "integer")
VarCat <- which(ListaVar == "character")
VarNum <- which(ListaVar == "numeric")
#Boxplots
loopNum1 <- function(names) {
  n.use<-sym(names)
  plotbox <- ggplot(comandes,aes(x=!! n.use)) + geom_boxplot(col='black', fill='steelblue') +
theme_classic()
  return(plotbox)
}
lapply(c(names(VarNum),names(VarInt)),loopNum1)
```

```{r}
comandes[which(comandes$order_price >1000),]
```

order_items <chr>	order_price <dbl>
[('Shrimp', 10), ('Fish&Chips', 10), ('Pasta', 9), ('Salmon', 8)]	1099.12

14. Eliminació d'algunes variables que creiem sense importancia

S'elimina per decisió final la variable *order_items*

```{r}
comandes<-comandes[,!colnames(comandes)=="order_items"]
```

Es guarda la base de dades ja preprocessada

La base de dades tractada té 463 observacions i 11 variables.

El següent pas es iniciar un breu anàlisi descriptiu i per tant es mantenen aquestes dades en format R.data per eficiència d'espai i s'utilitzen més endavant.

```{r}
save(comandes, file = paste0(path0,"dataPRE.RData"))
save(branches, file = paste0(path0,"branches.RData"))
```

Annex 8: Script de l'anàlisi descriptiu gràfic

---
title: "Descriptiva Gràfica Melbourne"
author: "Marcel Canals"
date: "31/3/2022"
output: html_document

Importació de les dades ja processades en un format R.data

```{r}
path<-"C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/"
load(paste0(path,"dataPRE.RData"))
load(paste0(path,"branches.RData"))
```

Estadístics descriptius basics

```{r}
summary(comandes)
```

S'utilitzen gràfics construits a través de la llibreria *ggplot2* que permet treure més rendiment i una millora en la visualització.

Gràfic de clients per caps de setmana i dies entre setmana

A partir de la funció *wday* de la llibreria *lubridate* s'identifiquen els dies de la setmana del 1 al 7 tot i que no segueixen l'ordre lógic de 1-dilluns, 2-dimarts, 3-dimecres...
S'ha de trobar l'ordre i crear la variable binaria.

```{r}
library(lubridate)
index_setmana<-wday(comandes$date)
table(wday(comandes$date))
```

Mirant un calendari del 2018 es troba l'ordre:
- *1*: dissabte
- *2*: diumenge
- *3*: dilluns
- *4*: dimarts
- *5*: dimecres
- *6*: dijous
- *7*: divendres

Es crea la nova variable binaria *weekend* que pren valor 0 si és entre setmana i 1 si la comanda es realitza el cap de setmana.

```{r}
weekend<-c()
for( i in 1:length(index_setmana)){
  if(index_setmana[i]==1 || index_setmana[i]==2)
    weekend<-c(weekend,1)
  else{
    weekend<-c(weekend,0)
  }
}
table(weekend) #en efecte hem creat la variable binaria
```

S'afegeix a la base de dades i es mapeja.

```{r}
library(ggplot2)
comandes$weekend<-weekend
comandes$weekend<-as.factor(comandes$weekend)
ggplot(comandes,aes(customer_lon,customer_lat,color=weekend))+
  geom_point()+ guides(colour = guide_legend(title = "Cap de setmana"))+ggtitle("Mapa dels clients segons el dia de la setmana")+xlab("Longitud")+ylab("Latitud")
```

Gràfic de clients per vacances

Engloban totes les vacances de Melbourne estudiades es té que:
- *Periode nadalenc*: Desde el 24 de desembre fins l'1 de gener (Any nou)
- *Els dies Easter*: Del 31 de gener al 2 d'abril
- *Dia d'Australia*: 26 de gener
- *Dia del treballador*: 12 de març
- *"Good Friday"*: 30 d'abril
- *Dia d'Anzac*: 25 d'abril
- *L'aniversari de la Reina*: 11 de juny
- *El divendres abans de la gran final AFL*: 28 de setembre
- *Dia de la Copa Melbourne*: Es la carrera a cavalls celebrada el 6 de novembre

Es crea una variable binaria anomenada *holiday* que imputa el valor 1 si el dia de la comanda és festiu i 0 en cas contrari.
Per tant, primer es defineix un vector amb les dates dels festius per avaluar per cada comanda la seva data corresponent.

```{r}
```

```
holy<-c("2018-01-01","2018-02-26","2018-03-12","2018-03-31"  ,"2018-04-01"   ,"2018-04-02"
          ,"2018-04-30"   ,"2018-06-11"   ,"2018-09-28"   ,"2018-11-6","2018-12-24"        ,"2018-12-25"
          ,"2018-12-26"   ,"2018-12-27"   ,"2018-12-28"," 2018-12-29"     ,"2018-12-30","2018-12-31")
```

La funció de les vacances:

```
```{r}
holiday<-c()
date<- as.character.Date(comandes$date)
date[1]<-"2018-08-07" #no se perquè el primer valor no mel treu bé
j<-1
for(i in 1:length(comandes$date)){
  j<-1
    while(j<=length(holy)){
       if(date[i]==holy[j]){
         holiday<-c(holiday,1)
         j<-200
       }
       if(date[i]!=holy[j] && j==18){
         holiday<-c(holiday,0)
       }
      j<-j+1
    }
}
table(holiday)
```
```

Ara es construeix el gràfic dels clients agrupant per comandes en període festiu i no festiu.

```
```{r}
comandes$holiday<-holiday
comandes$holiday<-as.factor(comandes$holiday)
ggplot(comandes,aes(customer_lon,customer_lat,color=holiday))+
  geom_point()+ guides(colour = guide_legend(title = "Dies de vacances"))+ggtitle("Mapa dels clients
segons el període de vacances")+xlab("Longitud")+ylab("Latitud")
```
```

Gràfic de clients per apats

```
```{r}
ggplot(comandes,aes(customer_lon,customer_lat,color=order_type))+
  geom_point()+ guides(colour = guide_legend(title = "Àpat"))+ggtitle("Mapa dels clients segons
l'àpat")+xlab("Longitud")+ylab("Latitud")
```
```

Es subdivideix la base de dades per tal d'estratificar per àpats.

```
{r}
comandes_break<-comandes[which(comandes$order_type=="Breakfast"),]
comandes_din<-comandes[which(comandes$order_type=="Dinner"),]
comandes_lun<-comandes[which(comandes$order_type=="Lunch"),]
data.frame(
breakfast=length(comandes_break$order_id),lunch=length(comandes_lun$order_id),dinner=length(co
mandes_din$order_id))
```
```

Es construeixen els 3 grafics per separat.

```
```{r,out.width="50%"}
ggplot(comandes_break,aes(customer_lon,customer_lat))+
  geom_point(color=2)+ guides(colour = guide_legend(title = "Àpat"))+ggtitle("Mapa dels clients que
han demanat esmorzar")+xlab("Longitud")+ylab("Latitud")
ggplot(comandes_lun,aes(customer_lon,customer_lat))+
  geom_point(color=3)+ guides(colour = guide_legend(title = "Àpat"))+ggtitle("Mapa dels clients que
han demanat dinar")+xlab("Longitud")+ylab("Latitud")
```

```
ggplot(comandes_din,aes(customer_lon,customer_lat))+
  geom_point(color=4)+ guides(colour = guide_legend(title = "Àpat"))+ggtitle("Mapa dels clients que han demanat sopar")+xlab("Longitud")+ylab("Latitud")
```

Gràfic de clients per establiment de comanda

```{r}
ggplot(comandes,aes(customer_lon,customer_lat,color=branch_code))+
  geom_point()+ guides(colour = guide_legend(title = "Establiment"))+ggtitle("Mapa dels clients segons l'establiment")+xlab("Longitud")+ylab("Latitud")
```

Gràfic de valor de comanda per client

```{r}
ggplot(comandes,aes(customer_lon,customer_lat,color=order_price))+
  geom_point()+ guides(colour = guide_legend(title = "Preu"))+ggtitle("Mapa dels clients segons el preu de comanda")+xlab("Longitud")+ylab("Latitud")
```

Guardem la base de dades amb els canvis realitzats

S'ha finalitzat la part descriptiva del treball, creant dues noves variables; *weekend* (binaria amb 1 = és cap de setmana i 0 = cas contrari) i *holiday* (binaria amb 1 = és període de vacances i 0 = cas contrari).
S'exporten les dades en format R.data, la base de dades *comandes* i les 3 subbases.

```{r}
save(comandes, file = paste0(path,"data_post_descriptiva.RData"))
save(comandes_break, file = paste0(path,"comandes_break.RData"))
save(comandes_lun, file = paste0(path,"comandes_lun.RData"))
save(comandes_din, file = paste0(path,"comandes_din.RData"))
```

Annex 9: Script de la modelització 1: Creació del problema

---
title: "Modelització1"
author: "Marcel Canals"
date: "4/4/2022"
output: html_document

Importació de les dades

```{r}
path<-"C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/"
load(paste0(path,"data_post_descriptiva.RData"))
load(paste0(path,"branches.RData"))
load(paste0(path,"comandes_break.RData"))
load(paste0(path,"comandes_lun.RData"))
load(paste0(path,"comandes_din.RData"))
```

Set de combinació 1: Àpats i Caps de Setmana/Dies entre setmana

Es creen i grafiquen 6 situacions:

- **com_break_week**: Les comandes per l'esmorzar que es realitzen durant la setmana
- **com_break_end**: Les comandes per l'esmorzar que es realitzen durant el cap de setmana
- **com_lun_week**: Les comandes pel dinar que es realitzen durant la setmana
- **com_lun_end**: Les comandes pel dinar que es realitzen durant el cap de setmana
- **com_din_week**: Les comandes pel sopar que es realitzen durant la setmana
- **com_din_end**: Les comandes pel sopar que es realitzen durant el cap de setmana

Creació de les combinacions

Al ja tenir una predisposició amb els 3 subgrups creats en la part descriptiva serà més fàcil formar les combinacions. Es comença amb l'esmorzar durant la setmana.

1.1 Esmorzar:Entre setmana

Primer es crea la combinació:

```{r}
Combreak_week<-comandes_break[which(comandes_break$weekend==0),]
```

I després es grafica:

```{r,out.width="50%"}
plot(com_break_week$customer_lon,com_break_week$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes d'esmorzar entre setmana",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

1.2 Esmorzar:Cap de setmana

Primer es crea la combinació:

```{r}
com_break_end<-comandes_break[which(comandes_break$weekend==1),]
```

I després es grafica:

```{r,out.width="50%"}
plot(com_break_end$customer_lon,com_break_end$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes d'esmorzar el cap de setmana",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

1.3 Dinar:Entre setmana

Posar especial atenció entre dinar("lunch") i sopar("dinner")

```{r}
com_lun_week<-comandes_lun[which(comandes_lun$weekend==0),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_lun_week$customer_lon,com_lun_week$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de dinar entre setmana",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

1.4 Dinar:Cap de setmana

Posar especial atenció entre dinar("lunch") i sopar("dinner")

```{r}
com_lun_end<-comandes_lun[which(comandes_lun$weekend==1),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_lun_end$customer_lon,com_lun_end$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de dinar entre setmana",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

1.5 Sopar:Entre setmana

```{r}
com_din_week<-comandes_din[which(comandes_din$weekend==0),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_din_week$customer_lon,com_din_week$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de sopar entre setmana",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

1.6 Sopar:Cap de setmana

```{r}
com_din_end<-comandes_din[which(comandes_din$weekend==1),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_din_end$customer_lon,com_din_end$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de sopar a Melbourne",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

Set de combinació 2: Àpats i Vacances/No vacances

La segona combinació que farem serà estudiar la variable order_type però en aquest cas realitzant subgrups per la variable *holyday*, que mesura els dies festius.

- **com_break_norm**: Les comandes per l'esmorzar que es realitzen durant els dies feiners
- **com_break_holy**: Les comandes per l'esmorzar que es realitzen durant les vacances
- **com_lun_norm**: Les comandes pel dinar que es realitzen durant els dies feiners
- **com_lun_holy**: Les comandes pel dinar que es realitzen durant les vacances
- **com_din_norm**: Les comandes pel sopar que es realitzen durant els dies feiners
- **com_din_holy**: Les comandes pel sopar que es realitzen durant les vacances

Primer es crea la combinació:

```{r}
com_break_norm<-comandes_break[which(comandes_break$holiday==0),]
```

I després es grafica:

```{r,out.width="50%"}
plot(com_break_norm$customer_lon,com_break_norm$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes d'esmorzar els dies feiners",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

2.2 Esmorzar:Vacances

```{r}
com_break_holy<-comandes_break[which(comandes_break$holiday==1),]
```

Grafiquem:

```{r,out.width="50%"}
plot(com_break_holy$customer_lon,com_break_holy$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes d'esmorzar el dies de vacances",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

2.3 Dinar:Dies feiners

```{r}
com_lun_norm<-comandes_lun[which(comandes_lun$holiday==0),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_lun_norm$customer_lon,com_lun_norm$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de dinar els dies feiners",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

```

#### 2.4 Dinar:Vacances

```{r}
com_lun_holy<-comandes_lun[which(comandes_lun$holiday==1),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_lun_holy$customer_lon,com_lun_holy$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de dinar en període de vacances",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

2.5 Sopar:Dies feiners

```{r}
com_din_norm<-comandes_din[which(comandes_din$holiday==0),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_din_norm$customer_lon,com_din_norm$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de sopar en dies feiners",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

2.6 Sopar:Vacances

```{r}
com_din_holy<-comandes_din[which(comandes_din$holiday==1),]
```

Grafiquem la combinació:

```{r,out.width="50%"}
plot(com_din_holy$customer_lon,com_din_holy$customer_lat,xlim = c(144.9,145.05),ylim=c(-37.865,-37.765),main="Comandes de sopar en període de vacances",xlab = "Longitud",ylab="Latitud")
points(branches$branch_lon,branches$branch_lat,col=c(4,6,7),pch=7,cex=2)
legend("topleft",c(branches$branch_name[1],branches$branch_name[2],branches$branch_name[3]),text.col = c(4,6,7))
```

Creació de les distàncies entre nodes

Funció que calcula les distàncies

```{r}
calc_Dist<- function(ddd){
  library(geosphere)
  attach(ddd)
```
```

```
  ## función de distància
  vect<-c()
  for( i in 1:length(order_id)){
        for( j in 1:length(order_id)){
          p1_lon<-customer_lon[i]
          p1_lat<-customer_lat[i]
          p2_lon<-customer_lon[j]
          p2_lat<-customer_lat[j]

          a<-distGeo(p1=c(p1_lon,p1_lat),p2=c(p2_lon,p2_lat))
          vect<-c(vect,a)
        }
  }
return(vect)
}
```

Definició del problema

En aquest apartat es defineixen les característiques del problema que es vol optimitzar (quins restaurants hi formen part, quin àpat, si és en període festiu o no, si és en cap de semana…). Per tant els espais amb punts (…..) són els llocs on s'ha d'omplir segons el problema que es vulgui tractar

```
```{r,warning=FALSE}
#selecionem model i restaurants
rest<-c(…….) #per exemple c("BK","TP","NS")
#calcul de la mostra
conc<-c()
a<-0
for( i in 1:length(rest)){
#es defineixen les propietats, per exemple "com_din_holy" (sopar en vacances)
  a<-which(……...$branch_code==rest[i])
  conc<-c(conc,a)
}
dd<-…….[conc,]
#Pel calcul de les distancies nomes necesitem l'identificador, la latitud i longitud
ddd<-dd[,c(1,7,8)]
#afegim com si fos un punt la localització del restaurant
for ( i in 1:length(rest)){
ddd[length(ddd$order_id)+1,c(1:dim(ddd)[2])]<-
c(rest[i],branches$branch_lat[which(branches$branch_code==rest[i])],branches$branch_lon[which(bra
nches$branch_code==rest[i])])
ddd$customer_lat<-as.numeric(ddd$customer_lat)
ddd$customer_lon<-as.numeric(ddd$customer_lon)
}
```
```

Es crida la funció *calc_Dist* per calcular les distancies d'aquest problema

```
```{r,warning=FALSE}
#La funció nomes pot tornar el vector de distancies i s'ha de crear en data.frame fora.
dist<-calc_Dist(ddd)
dist<-matrix(dist,nrow = nrow(ddd))
diag(dist)<-99999 #per tal d'estalviarnos una restricció
dist<-as.data.frame(dist)
```
```

grafiquem les possibles distàncies

```
```{r,warning=FALSE}
#es grafiquen els domicilis dels clients
plot(ddd$customer_lon,ddd$customer_lat,main="Comandes de sopar en vacances dels 3
restaurants",xlab = "Longitud", ylab="Latitud")
#es grafica la localització del restaurant o restaurants que participen en la comanda
nom_res<-c()
```
```

```
for(i in 1:length(rest)){
points(branches$branch_lon[[which(branches$branch_code==rest[i])]],branches$branch_lat[[which(br
anches$branch_code==rest[i])]],col=(5),pch=9 ,cex=2)
a<-branches$branch_name[branches$branch_code==rest[i]]
nom_res<-c(nom_res,a)
}
legend("topleft",nom_res,text.col =(5),cex = 0.6)
text(ddd$customer_lon,ddd$customer_lat,labels = order_id,cex=0.5,pos = 1)
#es grafica les distancies entre clients
#primer de tot es crea aquesta funció per saber els index que sintrodueixen a la funció segments
x<-c()
y<-c()
for( i in 1:(nrow(ddd)-1)){
  x_i<-rep(i,(nrow(ddd)-i))
  x<-c(x,x_i)
  y_i<-(i+1):nrow(ddd)
  y<-c(y,y_i)
}
segments(x0=ddd$customer_lon[x],y0=ddd$customer_lat[x],x1=ddd$customer_lon[y],y1=ddd$custom
er_lat[y],lty = 2,col = 3)
```

De cara a optimitzar aquest problema en concret a través de SAS s'han d'introduir unes noves variables que ajuden a la importació de les dades a aquest nou software.

Tractament

```{r}
clients<-c(1:nrow(dist))
constant<-rep(1,nrow(dist))
mod_p2<-cbind(constant,clients,dist)
```

Exportem en format csv el model per implementar-lo al SAS

```{r,warning=FALSE}
library(readr)
setwd("C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante
Melbourne/bases de datos/base de modelos")
write_csv(mod_p2,file ="mod.csv",append = FALSE)
```

Cada vegada que es vulgui exportar en csv les dades, s'haura d'anar a la carpeta en qüestió i eliminar les dades anteriors.

Es guarda la base de dades del problema en concret que s'està tractant

```{r}
save(ddd, file = paste0(path,"datosProblemaOpt.RData"))
save(rest, file = paste0(path,"RestProblema.RData"))
```

Annex 10: Script de SAS: Optimització de la ruta

Aquest és l'script base que realitza l'optimització, depenen del problema s'introdueixen més o menys restriccions.

```
/* importació de les dades*/

%web_drop_table(mod);
```

```
FILENAME REFFILE '/home/u47389835/TFG/mod.csv';

PROC IMPORT DATAFILE=REFFILE
            DBMS=CSV
            OUT=mod;
            GETNAMES=YES;
RUN;
proc print data= mod;
proc contents data=mod;run;
%web_open_table(mod);

/* optimització del problema */
proc optmodel;
            /* declaració dels index sets */
            set row;
            set col;

            /* declaració dels paràmetres */
            number demanda{col};
            number distancies{row, col};
            number oferta{row};

            /* importació de valors dels índex*/
            read data mod into col=[clients] demanda[clients]=col("constant");
            read data mod into row=[clients] oferta[clients]=col("constant");
            /* importació de les distàncies */
            read data mod into row=[clients]
             {d in col} <distancies[clients,d]=col("V"||d)>;

             /*matriu binaria que indicara la ruta òptima*/
            var X{row, col} BINARY;

            /* minimització de la suma de distancies*/
            min z = sum{i in row, j in col} X[i,j] * distancies[i,j];

            /*restriccions*/
            /* La ruta no pasa dues vegades pel mateix client*/
            con ClientCol{j in col}: sum{i in row} X[i,j] = 1;
            con ClientRow{i in row}: sum{j in col} X[i,j] = 1;

            /*Resolució del problema i visualització del resultat*/
            SOLVE;
            print z;
            *print distancies*
            print X;

            /* Es guarden els resultats*/
              create data solucio
      from [fila columna]={row,col} ruta=X;
quit;
proc print data=solucio;

/* S'exporta la solució*/
proc export data=solucio dbms=csv outfile='/home/u47389835/TFG/solucio.csv';
```

Annex 11:Script de modelització 2:Visualització i validació

---

title: "Modelització 2. Visualització de la solució"
author: "Marcel Canals"
date: "15/5/2022"
output: html_document

Importació de la ruta desde SAS (SAS -> csv -> R)

```{r}
ruta <- read.csv("C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/base de modelos/soluciones(rutas) de SAS/solució.csv")
```

Importació de les dades del problema en concret

```{r}
path<-"C:/Users/marce/OneDrive/Escriptori/UNI/C4/s2/TFG/Idea  Delivery restaurante Melbourne/bases de datos/"
load(paste0(path,"data_post_descriptiva.RData"))
load(paste0(path,"branches.RData"))
load(paste0(path,"datosProblemaOpt.RData"))
load(paste0(path,"RestProblema.RData"))
```

La idea es convertir la informació de la ruta per tal de poder construir gràficament segments que permetin veure la solució.

Per això es crea una funció que identifica els punt inici i final de cada segment

```{r}
ruta_opt<-ruta[which(ruta$ruta==1),]
lon_puntA<-c();lat_puntA<-c();lon_puntB<-c();lat_puntB<-c()
for( i in 1:nrow(ruta_opt)){
  lon_puntA<-c(lon_puntA,ddd$customer_lon[ruta_opt$fila[i]])
  lat_puntA<-c(lat_puntA,ddd$customer_lat[ruta_opt$fila[i]])
  lon_puntB<-c(lon_puntB,ddd$customer_lon[ruta_opt$columna[i]])
  lat_puntB<-c(lat_puntB,ddd$customer_lat[ruta_opt$columna[i]])
}
ruta_opt<-cbind(ruta_opt,lon_puntA,lat_puntA,lon_puntB,lat_puntB)
```

###Grafic de la ruta óptima

```{r}
#es grafiquen els domicilis dels clients
plot(ddd$customer_lon,ddd$customer_lat,main="Ruta de comandes de sopar en vacances dels 3 restaurants", xlab = "Longitud",ylab="Latitud")
#es grafica la localització dels restaurants
nom_res<-c()
 for(i in 1:length(rest)){
points(branches$branch_lon[[which(branches$branch_code==rest[i])]],branches$branch_lat[[which(branches$branch_code==rest[i])]],col=(6),pch=7 ,cex=3)
 a<-branches$branch_name[branches$branch_code==rest[i]]
 nom_res<-c(nom_res,a)
 }
 legend("topleft",nom_res,text.col =(6),cex = 0.8)
#s'afegeixen els identificadors de clients i restaurants
text(ddd$customer_lon,ddd$customer_lat,labels = order_id,cex=0.5,pos = 1)
#es grafica les distancies entre clients
segments(x0=ruta_opt$lon_puntA,y0=ruta_opt$lat_puntA,x1=ruta_opt$lon_puntB,y1=ruta_opt$lat_puntB,lwd = 2)
# Es crea aquesta funció per saber els index que sintrodueixen a la funció(segments)
x<-c()
```

```
y<-c()
for( i in 1:(nrow(ddd)-1)){
  x_i<-rep(i,(nrow(ddd)-i))
  x<-c(x,x_i)
  y_i<-(i+1):nrow(ddd)
  y<-c(y,y_i)
}
segments(x0=ddd$customer_lon[x],y0=ddd$customer_lat[x],x1=ddd$customer_lon[y],y1=ddd$custom
er_lat[y],lty = 2,col = 3,lwd = 0.1)
```

Funció que comprova si tots els punts estan units a través de ruta_opt

```
```{r}
punts<-ruta_opt[,c(1,2)]
cami<-c()
cont<-1
i<-nrow(punts)
inici<-punts$fila[i]
error<-FALSE
 #es comença pel final perque es on hi ha el restaurant
while(cont<nrow(punts)){
  cami_i<-punts$columna[which(punts$fila==inici)]
  if(cont==1){
    cami<-c(cami,inici)
  }
  cami<-c(cami,cami_i)
  if(cont>1 && sum(duplicated(cami)>=1)){
    cont<-nrow(punts)
    print("La ruta no uneix tots els punts!")
    error<-TRUE
  }
  inici<-cami_i
  cont<-cont+1
  if(cont==nrow(punts) && error==FALSE){
    print("La ruta uneix tots els punts")
  }
}
cami
```
```

Procediment per executar cada problema d'optimització

1. S'estableixen les condicions del problema (quin àpat serà la comanda, si dependrà de si és un dia festiu o feiner, si és en cap de setmana o no i quins restaurants hi formaran part de la ruta)
en l'script de Modelització 1

2. Es crida la funció **calc_Dist** per calcular les distancies d'aquest problema
en l'script de Modelització 1

3. Es grafiquen totes les distàncies possibles del problema
en l'script de Modelització 1

4. S'afegeixen a les dades del problema les variables necesaries per implementar l'optimització al SAS
en l'script de Modelització 1

5. S'importen les dades cap a SAS (es guarden les distancies del problema en csv amb el nom **mod.csv** i també es guarden les dades del problema en **r.data**)
en l'script de Modelització 1

6. Dins del SAS on Demand s'elimina l'anterior **mod** i es carrega el nou amb les noves distancies. També s'elimina la anterior solució anomenada **solucio.csv** i es fa run del script **ModeloBase**
en SAS on demand

7. Una vegada executat el model, s'haurà creat un fitxer **solucio.csv** que conté la ruta optima. Es descarga desde SAS i s'introdueix a la carpeta **soluciones(rutas)** previament s'haura canviat el nom de **solucio** a **solucio_x** segons quants problemes s'hagin fet anteriorment.
en SAS on demand

8. Es carreguen les dades **solució_x** i **r.data** i es grafica la ruta comparant-la amb les distancies possibles
en l'script de Modelització 2