Práctica

Presentación

Esta práctica cubre los todos los módulos del programa de la asignatura.

Competencias

Las competencias que se trabajan en esta prueba son:

- Uso y aplicación de las TIC en el ámbito académico y profesional.

- Capacidad para innovar y generar nuevas ideas.

- Capacidad para evaluar soluciones tecnológicas y elaborar propuestas de proyectos teniendo en cuenta los recursos, las alternativas disponibles y las condiciones de mercado.

- Conocer las tecnologías de comunicaciones actuales y emergentes, así como saberlas aplicar convenientemente para diseñar y desarrollar soluciones basadas en sistemas y tecnologías de la información.

- Aplicación de las técnicas específicas de ingeniería del software en las diferentes etapas del ciclo de vida de un proyecto.

- Capacidad para aplicar las técnicas específicas de tratamiento, almacenamiento y administración de datos.

- Capacidad para proponer y evaluar diferentes alternativas tecnológicas para resolver un problema concreto.

Objetivos

La correcta asimilación de todos los aspectos trabajados durante el semestre

En esta Práctica abordamos un caso real de minería de datos dónde tenemos que poner en juego todos los conceptos trabajados.

Hay que trabajar todo el ciclo de vida del proyecto. Desde el objetivo del proyecto hasta la implementación del conocimiento encontrado pasando por la preparación, limpieza de los datos, conocimiento de los datos, generación del modelo, interpretación y evaluación.

Descripción de la Práctica a realizar

La prueba está estructurada en un total de un ejercicio práctico.

Recursos

**Básicos**

Material docente proporcionado por la UOC.

**Complementarios**

En el wiki del aula:

* Los descritos para las PECs anteriores
* Fichero gourmetdb.zip

Criterios de valoración

**Ejercicios prácticos**

Para la práctica es necesario documentar en cada apartado del ejercicio práctico qué se ha hecho, cuál era el objetivo y cómo se ha hecho.

Formato y fecha de entrega

El formato de entrega es: usernameestudiante-practica.zip

+ Detalle composición del paquete..............

--Indice\_documentos\_practica.txt

+ Respuestas................................................

--- username\_estudiante\_MD\_Practica.pdf/odt/docx

+ Scripts R con funciones de las diferentes secciones de la práctica.

--- md\_prac-n.R

--- md\_prac-n+1.R

+ Fichero de datos generados a partir de los originales con transcendencia.

--- md\_prac-n.+1.csv

La fecha de entrega es: 08/01/2018

Es necesaria entregarla en el buzón de entregas del aula

Enunciado

Introducción

Queremos plantear como podría ser un hipotético caso real de proyecto de minería de datos. Todo parte de una presentación del cliente que expone un problema de negocio difuso y muy genérico que se deberá reconducir como proyecto de minería de datos. Habéis trabajado un ciclo de vida de proyecto que ahora tiene que ser el eje de las respuestas.

Transformar los objetivos de negocio en objetivos de minería de datos.

Preparar una estrategia: **Hay que aplicar al menos dos tipos de algoritmos de MD vistos en el semestre. Por lo tanto, como mínimo tendrá dos objetivos de MD.**

Analizar bien los datos, entender el modelo que representan los datos.

Preparar los datos, limpieza, creación del modelo, evaluación de la calidad, interpretación y propuesta de que se puede hacer con el conocimiento extraído.

Orientar la práctica como podría ser la presentación de resultados que le haríais al cliente. Por tanto lenguaje claro y ordenado.

Los datos: Están en ficheros csv. Hay un script de SQL que os puede ayudar a entender la tipología de los datos y sus relaciones. Si no queda claro preguntar al Foro.

Para preparar los datos creatividad y libertad absoluta: R, Excel, Acces, Mysql ... Lo que queráis. Trabajar en las herramientas que seáis más eficientes y productivos. Si os perdéis u os quedéis parados en algún punto no dudéis en contactar conmigo.

Presentación del cliente

Hola, me llamo Juan Schlemmer. Soy CEO a la cadena de tiendas presenciales Gourmet. Nos dedicamos al negocio de la alimentación y bebida de calidad. Tenemos presencia en tiendas en Europa y Estados Unidos.

El negocio no nos va mal, pero me gustaría poder aumentar la facturación de la empresa. No tengo claro si aumentando las ventas o reduciendo gastos ... El otro día jugando al golf con un compañero que se dedica al negocio de las comunicaciones en comentó que había puesto en marcha un proyecto de ... Big Data o Minería de Datos o no sé qué del conocimiento ... no lo tengo claro ... Me dijo que le aconsejaron acciones concretas a partir de datos recogidos de su negocio y que está contento ... La cuestión es que os enviarán unos datos extraídos de nuestro ERP para que me hagáis esto de la minería. Por favor si tenéis dudas con los datos preguntar a nuestros técnicos. ¿Cuándo podré tener un primer informe? ¿Dentro de un mes? Perfecto ...

|  |
| --- |
| **Nota:** Propiedad intelectual A menudo es inevitable, al producir una obra multimedia, hacer uso de recursos creados por terceras personas. Es por lo tanto comprensible hacerlo en el marco de una práctica de los estudios de Informática, Multimedia y Telecomunicación de la UOC, siempre y cuando esto se documente claramente y no suponga plagio en la práctica.   Por lo tanto, al presentar una práctica que haga uso de recursos ajenos, se debe presentar junto con ella un documento en qué se detallen todos ellos, especificando el nombre de cada recurso, su autor, el lugar dónde se obtuvo y su estatus legal: si la obra está protegida por el copyright o se acoge a alguna otra licencia de uso (Creative Commons, licencia GNU, GPL ...).  El estudiante deberá asegurarse de que la licencia no impide específicamente su uso en el marco de la práctica. En caso de no encontrar la información correspondiente tendrá que asumir que la obra está protegida por copyright.  Deberéis, además, adjuntar los ficheros originales cuando las obras utilizadas sean digitales, y su código fuente si corresponde. |

Una de las primeras coses que tenemos que hacer y es claramente indispensable, es ver que datos tenemos, como están relacionados entre ellos, las cabeceras y con el enunciado que tenemos pensar bien claro que es lo que queremos hacer.

Contamos con 14 ficheros, uno de ellos con información referente a la base de datos y como está organizada y 13 ficheros .cvs(.csv) con los datos, sin cabeceras.

Vamos a describir la información que vemos en base de datos:

CREATE TABLE GOURMET.TIENDA (

NOMBRE CHAR(100) NOT NULL ,

DIRECCION CHAR(100) NOT NULL ,

SUPERFICIE DECIMAL(5,2) NOT NULL ,

FORMATO\_TIENDA CHAR(100) NOT NULL ,

PAIS CHAR(100) NOT NULL ,

TIPO\_ZONA CHAR(100) NOT NULL )

IN USERSPACE1 ;

Fichero tienda.cvs

Cabeceras: Nombre, Direccion, Superficie, Formato\_Tienda, Pais, Tipo\_Zona. Siendo el identificador el nombre y no puede existir ningún registro a nulo.

Se utilizando ésta información le ponemos la cabecera:

> tienda <- read.csv('tienda.cvs', encoding = 'UTF-8' , header = FALSE)

> cabeceraTienda <- c('Nombre', 'Direccion', 'Superficie', 'Formato\_Tienda', 'Pais', 'Tipo\_Zona')

> colnames(tienda) <- cabeceraTienda

> head(tienda)

Nombre Direccion Superficie Formato\_Tienda Pais Tipo\_Zona

1 Florencia Venize, 56 275 Tienda de barrio Italia XRTS

2 Fort Lauderdale 23, Vyn Ness Street 135 Centro comercial Estados Unidos BRTX

3 Liverpool 23, Kingstown Road 80 Tienda de barrio Reino Unido BRXS

4 Londres I 78, Westland Street 114 Galería Reino Unido BRTX

5 Londres II 198, SouthCastle Street 195 Centro comercial Reino Unido BRXX

6 Madrid Paseo de los Pajaritos, 158 150 Galería España BXTX

Nombre: Nombre de la tienda, generalmente la ciudad seguido del numero

Direccion: dirección del local

Superficie: superficie del local

Formato\_Tienda: información sobre el tipo de tienda

País

Tipo\_Zona:

CREATE TABLE GOURMET.PAIS (

NOMBREPAIS CHAR(100) NOT NULL ,

EXTENSION DECIMAL(10,0) NOT NULL ,

POBLACION DECIMAL(10,0) NOT NULL ,

NOMBREREGION CHAR(100) NOT NULL )

IN USERSPACE1 ;

Esta tabla se encuentra relacionada con la anterior (aunque no se ve rastro de foregin key) donde nomberpais debería de ser foregin key en tienda.

Fichero: país.cvs

Cabecera: NombrePais, Extension, Poblacion, NombreRegion

PK: NombrePaís

NombrePais: nombre del país

Extension: extensión del país (área)

Poblacion: población del país

NombreRegion: nombre d ela región en la que se encuentra el país

Seguimos el mismo camino

> pais <- read.csv('pais.cvs', encoding = 'UTF-8' , header = FALSE

)

> cabeceraPais <- c('NombrePais', 'Extension', 'Poblacion', 'NombreRegion')

> colnames(pais) <- cabeceraPais

> head(pais)

NombrePais Extension Poblacion NombreRegion

1 Estados Unidos 9372610 266504935 Norteamérica

2 España 504750 39510740 Sur Europa

3 Francia 547030 58376462 Norte Europa

4 Alemania 356910 81549019 Norte Europa

5 Reino Unido 244820 58452516 Norte Europa

6 Holanda 37330 15533258 Norte Europa

CREATE TABLE GOURMET.REGIONGEOGRAFICA (

NOMBREREGION CHAR(100) NOT NULL ,

CONTINENTE CHAR(100) NOT NULL )

IN USERSPACE1 ;

Relacionada con el anterior por nombre región

Fichero: regiongeográfica.cvs

Cabecera: NombreRegion, Continente

NombreRegion: nombre de la región

Continente: continente donde se encuentra la región especificada

> region <- read.csv('regiongeografica.cvs', encoding = 'UTF-8' , header = FALSE)

> View(region)

> cabeceraRegion <- c('NombreRegion', 'Continente')

> colnames(region) <- cabeceraRegion

> head(region)

NombreRegion Continente

1 Latinoamérica América

2 Norte Europa Europa

3 Sur Europa Europa

4 Oceanía Oceanía

CREATE TABLE GOURMET.CLIENTE (

CODCLIENTE CHAR(10) NOT NULL ,

NOMBRECLIENTE CHAR(100) ,

SEXO CHAR(25) ,

FECHANACIMIENTO DATE ,

ESTADOCIVIL CHAR(25) ,

DIRECCIÓN CHAR(100) ,

PROFESIÓN CHAR(100) ,

NUMEROHIJOS INTEGER ,

REGION CHAR(100) ,

NACIONALIDAD CHAR(100) ,

TOTALCOMPRAS INTEGER ,

PUNTOSACUMULADOS INTEGER )

IN USERSPACE1 ;

Fichero: cliente.cvs

Cabecera: CodCliente, NombreCliente, Sexo, FechaNaciemiento, EstadoCivil, Direccion, Profesion, NumeroHijos, Region, Nacionalidad, TotalCompras, PuntosAcumulados

CodCliente: código identificativo del cliente

NombreCliente: nombre del cliente

Sexo: sexo del cliente

FechaNacimiento: fecha de nacimiento del cliente

EstadoCivil: estado civil del cliente

Direccion: dirección del cliente

Profesión: profesión del cliente

NumeroHijos: número de hijos del cliente

Región: región donde vive el cliente

Nacionalidad

TotalCompras: número total de compras realizadas por el cliente

PuntosAcumulados: puntos acumulados por el cliente

En este caso ya se trata de información más útil, con información de los clientes, si queremos mejorar los ingresos, es posible que tengamos que utilizar los datos incluidos aquí.

CREATE TABLE GOURMET.PRODUCTO (

CODPRODUCTO CHAR(25) NOT NULL ,

DESCRIPCIÓN CHAR(100) NOT NULL ,

NOMBREPAIS CHAR(100) NOT NULL ,

COSTE DECIMAL(5,2) NOT NULL ,

PRECIOVENTA DECIMAL(5,2) NOT NULL ,

TIPOUNIDAD CHAR(100) NOT NULL ,

NOMBRESUBFAMILIA CHAR(100) NOT NULL ,

MARCA CHAR(100) NOT NULL ,

CODPROVEEDOR CHAR(10) NOT NULL )

IN USERSPACE1 ;

Fichero: producto.cvs

Cabeceras: CodProducto, Descripcion, NombrePais, Coste, PrecioVenta, TipoUnidad, NombreSubFamilia, Marca, CodProveedor

CodProducto: código identificativo del producto

Descripción. Descripción del producto

NombrePaís: nombre del país de origen

Coste: coste del producto para la tienda

PrecioVenta. Precio de venta del producto

TipoUnidad: descripción de con se venden las unidades del producto

NombreSubFamilia: nombre de la subfamilia a la que pertenece

Marca: marca del producto

CodProvedor: código identificativo del proveedor

Otra tabla interesante, como el objetivo es aumentar los ingresos o disminuir los gastos, podemos tener esta tabla en cuenta en caso de buscar que productos son los menos rentables y combinándolo con las compras por cuales se podría apostar más.

CREATE TABLE GOURMET.PROVEEDOR (

CODPROVEEDOR CHAR(10) NOT NULL ,

NOMBREPROVEEDOR CHAR(100) NOT NULL ,

PERSONACONTACTO CHAR(100) NOT NULL ,

DIRECCIÓN CHAR(100) NOT NULL ,

TELÉFONO CHAR(25) NOT NULL ,

PERIODOPAGO SMALLINT ,

PAGOPENDIENTE DECIMAL(5,2) ,

TIPOPROVEEDOR CHAR(100) ,

ALCANCE CHAR(25) NOT NULL )

IN USERSPACE1 ;

Fichero: proveedor.cvs

Cabeceras: CodProveedor, NombreProveedor, PersonaContacto, Direccion, Telefono, PeriodoPago, PagoPendiente, TipoProveedor, Alcance

CodProveedro: código identificativo del proveedor

NombreProveedor: nombre del proveedor

PersonaContacto: persona con la que contactar de los proveedores

Dirección

Teléfono

PeriodoPago: periodo en días en el que hay que realizar los pagos

PagoPendiente: hay pago pendiente (realmente está lleno de NA)

TipoProveedor: tipo de proveedor, sin datos

Alcance: global

CREATE TABLE GOURMET.NOMBRESUBFAMILIA (

NOMBRESUBFAMILIA CHAR(100) NOT NULL ,

DESCRIPCIÓN CHAR(100) ,

NOMBREFAMILIA CHAR(100) NOT NULL )

IN USERSPACE1 ;

Fichero: subfamilia.cvs

Cabecera: NombreSubFamilia, Descripcion, Nombrefamilia

NombreSubFamilia: nombre de la subfamilia asociada al producto

Descripción

NombreFamilia: nombre de la familia del producto (procedencia como Rioja o Pastelería)

Vemos como han empezado a encontrarse varios campos sin descripción (no llevan not null) pero está permitido, sin embargo a la hora de tratar los datos lo tendremos que tener en cuenta.

CREATE TABLE GOURMET.FAMILIA (

NOMBREFAMILIA CHAR(100) NOT NULL ,

DESCRIPCIÓN CHAR(100) ,

NOMBRESECCIÓN CHAR(100) NOT NULL )

IN USERSPACE1 ;

Fichero: familia.cvs

Cabecera: NombreFamilia, Descripción, NombreSeccion

NombreFamilia: nombre de la familia asociada

Descripción

NombreSeccion: sección de la tienda asociada al tipo de productos

CREATE TABLE GOURMET.SECCION (

NOMBRESECCIÓN CHAR(100) NOT NULL ,

DESCRIPCIÓN CHAR(100) )

IN USERSPACE1 ;

Fichero: sección.cvs

Cabecera: NombreSeccion, Descripcion

CREATE TABLE GOURMET.CABECERATICKET (

CODVENTA CHAR(25) NOT NULL ,

NOMBRETIENDA CHAR(100) NOT NULL ,

FECHA DATE NOT NULL ,

HORA SMALLINT NOT NULL ,

FORMAPAGO CHAR(100) NOT NULL ,

CODCLIENTE CHAR(25) ,

IMPORTETOTAL DECIMAL(8,4) NOT NULL ,

TOTALUNIDADES SMALLINT NOT NULL ,

PUNTOSTICKET SMALLINT NOT NULL )

IN USERSPACE1 ;

Podemos fijarnos un poco más en los valores que tenemos aquí, dado que puede ser información interesante sobre los tickets, en que tienda, cuando, pago, etc.

En este caso podemos ver que se trata solo de la primera información y que deberíamos relacionarla con el resto del ticket. En este caso podríamos hacer algo parecido a la PEC de reglas de asociación y descubrir que productos se están comprando con más frecuencia entre sí.

Fichero: cabeceraticket.cvs

Cabeceras: CodVenta, NombreTienda, Fecha, Hora, FormaPago, CodCliente, ImporteTotal, TotalUnidades, PuntosTicket

CodVenta: código identificativo de la venta realizada

NombreTienda: nombre de la tienda que realizó la venta

Fecha. Fecha de la compra

Hora: hora de la compra

FormaPago: forma en la que se realizó el pago

CodCliente: código identificativo del cliente

ImporteTotal: Importe total de la compra

TotalUnidades: número de unidades vendidas totales

PuntosTicket: puntos generados por la compra

CREATE TABLE GOURMET.LINEASTICKET (

CODLINEA SMALLINT NOT NULL ,

CODVENTA CHAR(25) NOT NULL ,

NOMBRETIENDA CHAR(100) NOT NULL ,

CODPRODUCTO CHAR(100) NOT NULL ,

CANTIDAD SMALLINT NOT NULL ,

PRECIOVENTA DECIMAL(8,4) NOT NULL ,

NOMBREPROMOCION CHAR(100) ,

CODCABECERA INTEGER NOT NULL )

IN USERSPACE1 ;

Necesitamos los datos de ambas tablas para formar el ticket entero, por lo que no son datos bastante valiosos

Fichero: lineasticket.cvs

Cabecera: CodLinea, CodVenta, NombreTienda, CodProducto, Cantidad, PrecioVenta, NombrePromocion, CodCabecera

CodLinea: número de línea en el ticket

CodVenta: código identificativo de la venta

NombreTienda: nombre de la tienda donde se vendió

CodProducto: producto que se vendió

Cantidad: cantidad del producto vendido

PrecioVenta: precio unitario de venta

NombrePromoción: promoción utilizada en la venta (puede no usarse promoción)

CodCabecera: no encuentro información

CREATE TABLE GOURMET.PROMOCION (

NOMBREPROMOCION CHAR(100) NOT NULL ,

TIPOPROMOCION CHAR(100) NOT NULL ,

COSTE SMALLINT ,

FECHAINICIO DATE NOT NULL ,

FECHAFIN DATE NOT NULL ,

CODPRODUCTO CHAR(25) NOT NULL ,

NOMBREFAMILIA CHAR(25) ,

NOMBRESECCIÓN CHAR(25) ,

NOMBRETIENDA CHAR(25) ,

NOMBREREGION CHAR(25) ,

NOMBREPAIS CHAR(25) )

IN USERSPACE1 ;

Fichero: promocion.cvs

Cabecera: NombrePromocion, TipoPromocion, Coste, FechaInicio, FechaFin, CodProducto, NombreFamilia, NombreSeccion, NombreTienda, NombreRegion, NombrePais

NombrePromoción: nombre de la promoción

TipoPromoción: tipo de la promoción

Coste: coste asociado (NAs)

FechaInicio

FechaFin

CodProducto: productos asociados a las promociones

NombreFamilia

NombreSeccion

NombreTienda

NombreRegion

NombrePais

CREATE TABLE GOURMET.PEDIDO (

CODPEDIDO CHAR(25) NOT NULL ,

NOMBRETIENDA CHAR(25) NOT NULL ,

CODPRODUCTO CHAR(25) NOT NULL ,

PRECIOCOMPRA DECIMAL(8,4) NOT NULL ,

CANTIDADSOLICITADA SMALLINT NOT NULL ,

FECHASOLICITUD DATE NOT NULL ,

CANTIDADENTREGADA SMALLINT NOT NULL ,

FECHAENTREGA DATE NOT NULL )

IN USERSPACE1 ;

Fichero: pedido.cvs

Cabecera: CodPedido, NombreTienda, CodProducto, PrecioCompra, CantidadSolicitada, FechaSolicitud, CantidadEntregada, FechaEntrega

CodPedido: código identificativo del pedido

NombreTienda: nombre de la tienda donde se realizó el pedido

CodProducto: código del producto vendido

PrecioCompra: precio de la compra del producto al proveedor

CantidadSolicitada: número de unidades solicitadas en el pedido

FechaSolicitud

CantidadEntregada

FechaEntregada

¡Ojo! En un vistazo rápido se ve que ciertos productos tienden a entregarse menos.

unique(pedido[pedido$CantidadSolicitada > pedido$CantidadEntregada, ]$CodProducto)

Con esto lo que queremos es tener información sobre los atributos de cada tabla, como crear un diccionario al que podemos volver y entender que es cada cosa. Aunque son bastante explicativos no en todos tiene el mismo sentido, por lo tanto tener esta referencia nos puede llegar a ser útil.

En este caso el cliente pretende aumentar sus ingresos o disminuir sus gastos, por ello vamos a intentar realizar un primer modelo de clustering (digo intentar dado que es un proceso iterativo). En este modelo vamos a tratar de agrupar los productos que más lucrativos son para la empresa y los que menos y si somos capaces de identificarlos por las tiendas, dado que no todos los consumidores tienen los mismos patrones en distintos países.

Vamos a empezar a visualizar los datos que nos interesan, en este caso la tabla productos nos será de gran ayuda para identificarlos en las ventas y los tickets que nos ayudan a localizar la tienda así como las ventas.

Vamos con producto, vamos aver un pequeño resumen de lo que tenemos:

> str(producto)

'data.frame': 339 obs. of 9 variables:

$ CodProducto : Factor w/ 339 levels "001CH","002CO",..: 1 2 3 4 5 6 7 8 9 10 ...

$ Descripcion : Factor w/ 248 levels "1989 Brut Champagne ",..: 64 72 78 84 112 133 170 134 185 243 ...

$ NombrePais : Factor w/ 15 levels "Alemania","Australia ",..: 13 13 13 13 13 13 13 13 13 13 ...

$ Coste : num 3.3 3.47 3.93 3.48 4.83 5.08 4.18 4.15 4.31 5.07 ...

$ PrecioVenta : num 4.69 4.98 5.59 4.98 6.84 7.13 6.23 6.23 6.23 7.48 ...

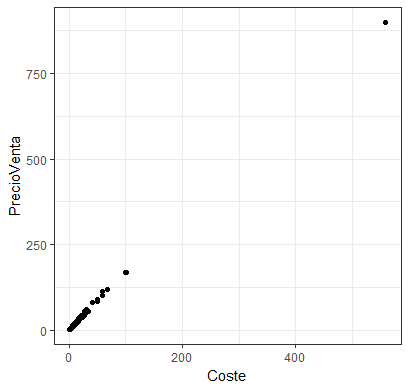
$ TipoUnidad : Factor w/ 5 levels "Botella ","Botella (1.5 l) ",..: 4 4 4 4 4 4 4 4 4 4 ...

$ NombreSubFamilia: Factor w/ 74 levels "Bizcochos ","Bollería",..: 68 68 72 68 71 71 72 68 72 73 ...

$ Marca : Factor w/ 184 levels "Allimat-Deiss ",..: 151 125 158 123 167 12 165 184 41 12 ...

$ CodProveedor : num 101 101 101 101 101 101 101 101 101 101 ...

Aquí tenemos información que vamos a utilizar, como el código, el coste y el precio de venta, por ahora el resto no nos interesa demasiado.



Vemos como por lo general la relación precio-coste es líneal sin embargo tenemos un registro que se encuentra muy por encima, vamos a verlo:

> producto[producto$PrecioVenta>300, ]

CodProducto Descripcion NombrePais Coste PrecioVenta TipoUnidad NombreSubFamilia Marca

263 264BO Bordeaux 88 Francia 558.58 899 Botella D.O. Bordeaux-Tinto Grant Estèphe

CodProveedor

263 315

Lo hemos comprobado por si tenía sentido o no.

Vamos a crear un df con los datos que vamos a necesitar de producto:

prod\_km <- producto[, c("CodProducto", "PrecioVenta", "Coste")]

De aquí podemos sacar datos más significativos, los ingresos netos del cliente por producto:

> prod\_km$Profit <- prod\_km$PrecioVenta - prod\_km$Coste

> summary(prod\_km)

CodProducto PrecioVenta Coste Profit

001CH : 1 Min. : 2.000 Min. : 1.40 Min. : 0.600

002CO : 1 1st Qu.: 6.305 1st Qu.: 4.18 1st Qu.: 2.080

003CL : 1 Median : 9.980 Median : 6.15 Median : 4.280

004DG : 1 Mean : 18.991 Mean : 11.10 Mean : 7.895

005HU : 1 3rd Qu.: 19.950 3rd Qu.: 11.10 3rd Qu.: 8.785

006MB : 1 Max. :899.000 Max. :558.58 Max. :340.420

(Other):333

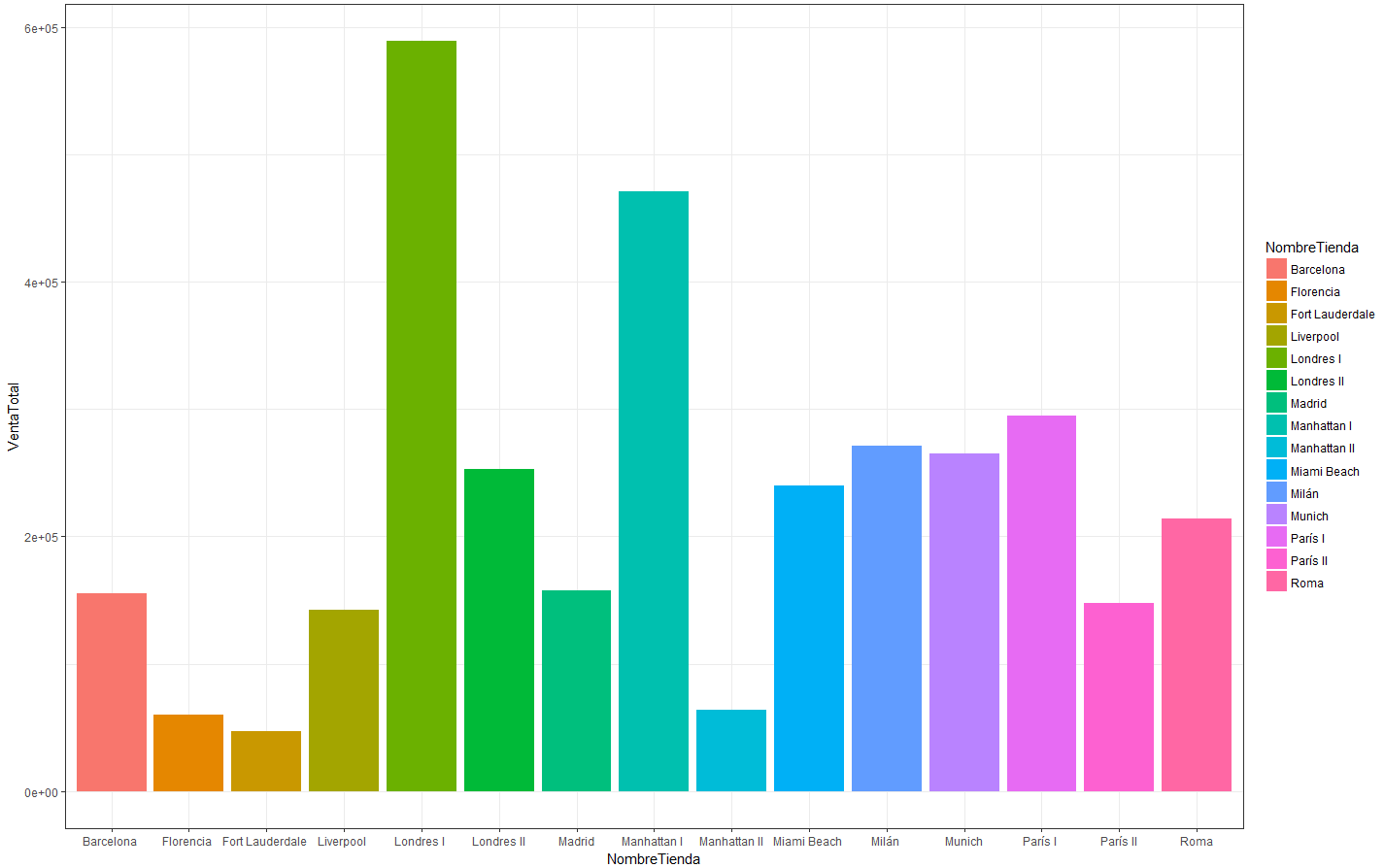
Tenemos la primera parte hecha, la segundo se trata de utilizar estos datos en las tablas de los tickets para ver cuánto se ha vendido de cada producto.

En este caso nos encontramos con un problema y es que todos los registros en la tabla de lineasticket tienen un espacio al final, por lo que no puede hacer un join bien, para ello utilizaremos gsub para eliminar este espacio:

lineas$CodProducto <- gsub(pattern = ' ', x = lineas$CodProducto, replacement = '')

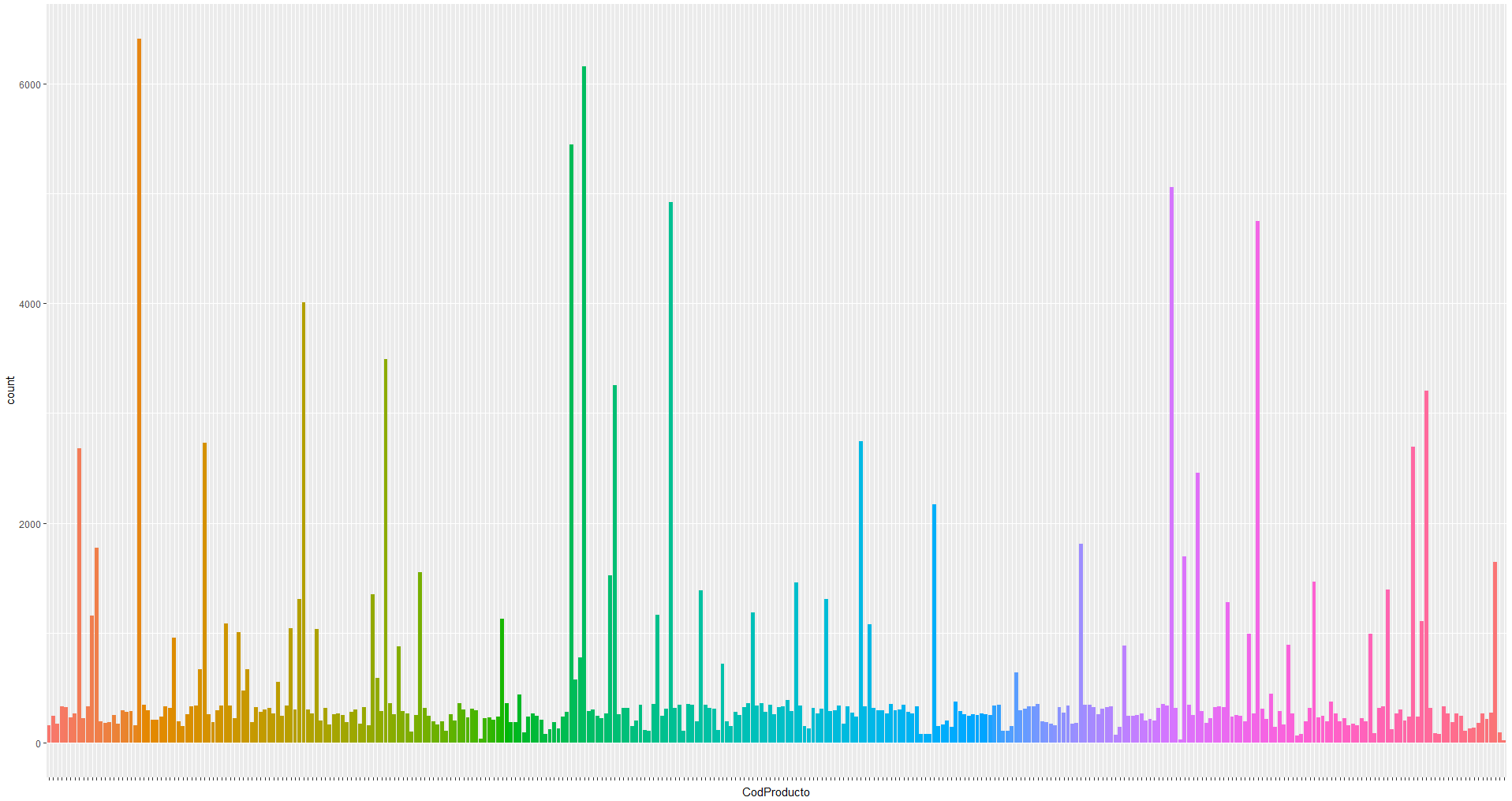
Vemos como también se repite el precio de venta, sin embargo esta redondeado, por lo que vamos a eliminar el que viene de producto. Por otro lado vamos que viene incluido unidad, para ello tenemos que multiplicarlo por el precio de venta para obtener los ingresos totales.

Veamos que tiendas son las que más ingresos han tenido:



No tiene mucho sentido en visualizar el coste, pues tal como hemos visto aumentan prácticamente de forma lineal, esto quiere decir que la gráfica será muy parecida, pero proporcional.

En el siguiente gráfico, vemos como hay productos “estrella” que claramente tienen más compras que el resto, estos son aquellos que tratamos de agrupar para potenciar que las tiendas incluyan más de estos y que por otro lado nos encuentren aquellos que nos aportan menos valor y que podrían reducir en su compra. Favoreciendo la compra de los productos más comprados y evitar comprar aquellos que apenas aportan beneficio puede ser beneficioso para el cliente.



Sin embargo, eso son las ventas de cada producto y no los ingresos, pues esto último es lo que nos interesa.

Vamos a utilizar dplyr para poder agrupar las ventas por el código del producto y así poder ver cuáles de ellos aportan más beneficios.

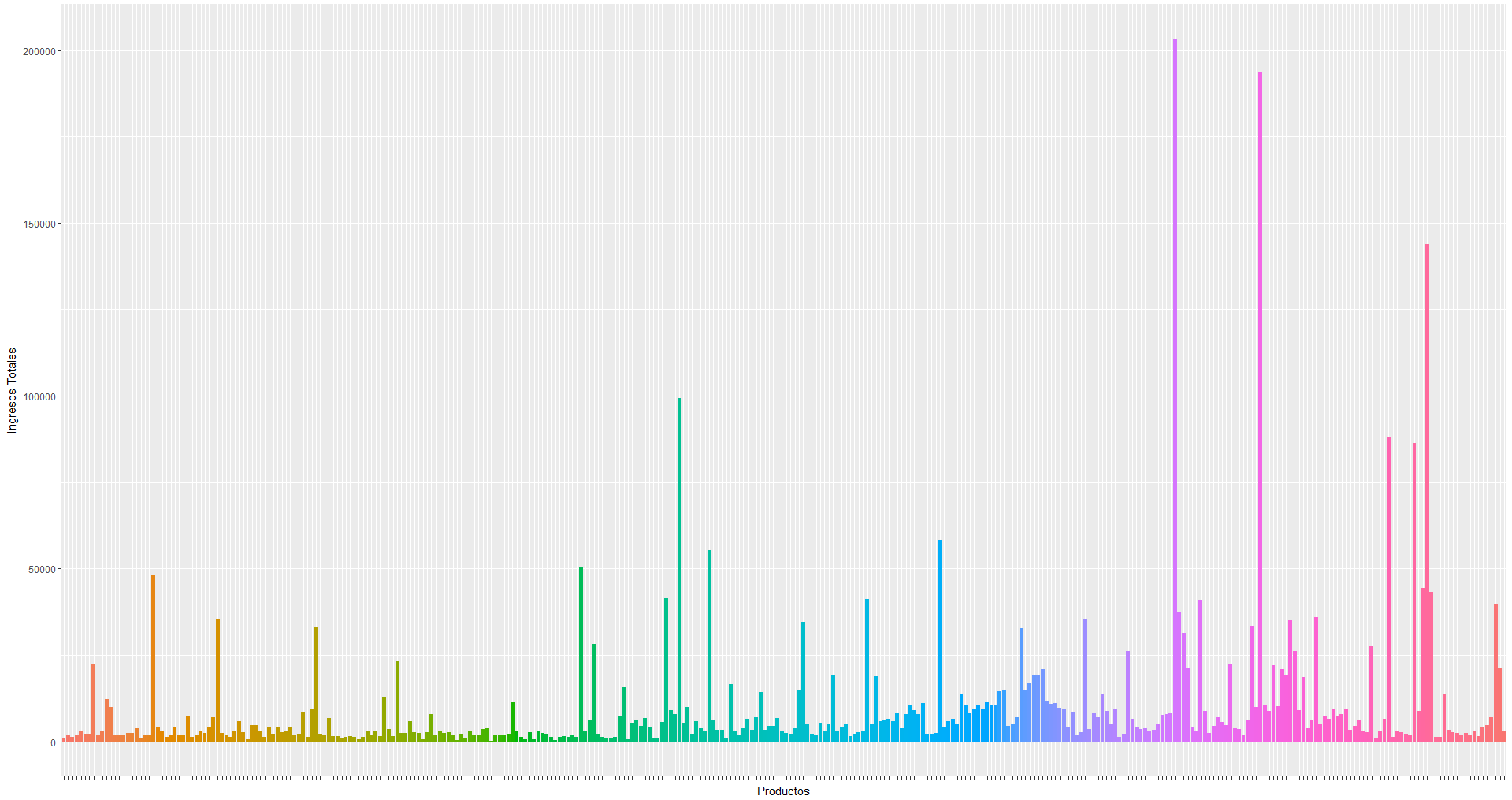
> #A partir de este punto queremos ordenar los datos y ver como se muestran, para ello usamos el paquete dplyr (podriamos usar tidyR tambien)

> library(dplyr)

> ingresoProducto <- as.data.frame(group\_by(ventasProd, CodProducto) %>%

+ summarise(sum(VentaTotal)))

Este df nos devuelve los datos que necesitamos para saber los precios de ventas totales sobre los productos:



Vemos como la cantidad no tiene por qué coincidir con los ingresos totales proporcionados por cada uno, están correlacionados, pero es posible que ciertos productos que se vendan menos cantidad proporcionen más beneficio, vender 10 botellas que aporten 300 euros por unidad sigue siendo más que vender 1000 que aporten 2 euros.

En este punto vamos a aplicar el clustering, vamos a intentar identificar los tipos de productos que tenemos, los que más ventas tienen y los que menos. Está claro que el dueño sabrá cuales son los productos que más vende, pero aquellos que se encuentren por debajo de ellos pueden no apreciarse tanto, esto es lo que queremos mostrar, productos que no son estrella sobre los que podría aumentar la oferta o al contrario productos que apenas aporten.

Para ello utilizaremos Kmeans y realizaremos varias prueas para ver que tipos de grupos nos muestra.