Descripción de los datos y variables

El archivo sales_train.csv – contiene los datos de entrenamiento (y test). Datos históricos de ventas diarias desde enero de 2013 hasta octubre de 2015.

- items.csv - Información complementaria sobre los artículos / productos.
- item_categories.csv - Información complementaria sobre las categorías de artículos.
- shops.csv- Información complementaria sobre las tiendas. Variables
- ID - una identificación que representa una tupla (tienda, artículo) dentro del conjunto de test
- shop_id – identificador único de tienda
- item_id - identificador único de producto
- item_category_id - identificador único de categoría de producto
- item_cnt_day – número de productos vendidos. Debe modelar las ventas mensuales a nivel de product y tienda.
- item_price – precio actual de un producto
- date – fecha en formato dd.mm.yyyy
- date_block_num - un número de mes consecutivo, utilizado por conveniencia.
  Enero 2013 es 0, Febrero 2013 es 1,. . . , Octubre 2015 es 33
- item_name – nombre del producto
- shop_name – nombre de la tienda
- item_category_name – nombre de la categoría de producto

1. Parte

Limpieza de datos e ingeniería de variables

a) Realice un proceso de limpieza de variables. Identifique casos anómalos e indique de forma explícita el tratamiento de estas observaciones.

Solución:

Supuesto:
Esta parte del ejercicio se realizó con toda la data, y debido a la cantidad de cambios a la evaluación, para este ejercicio mantuvimos el primer análisis.

Lo que primero realizamos fueron unos histogramas del comportamiento de los datos por día de la semana, mes y año.

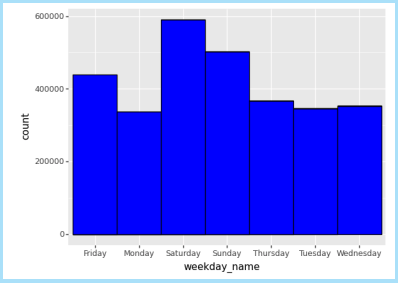

Figura 1: Histograma día de la semana

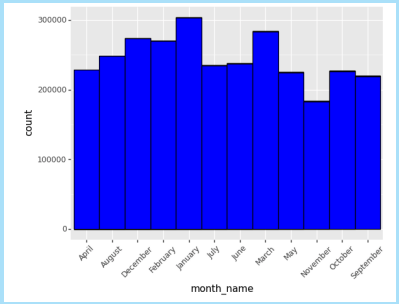

Figura 2: Histograma mes

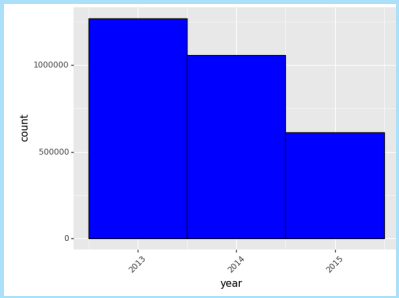

Figura 3: Histograma años

Figura 4: Histogramas

De estos podemos observar que los días con más ventas son los sábados y domingos, que el mes con más ventas es enero y de la figura 12 se desprende que las ventas están en descenso, cabe destacar que hay que tenemos datos para todos los meses del 2013 y 2014, pero para el año 2015 el último registro es el 31 de Octubre.

A continuación hicimos un gráfico de puntos para determinar visualmente si es que existen Outliers.

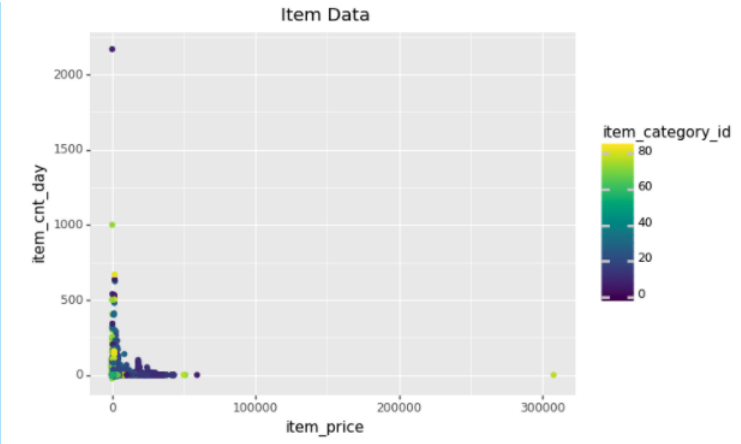

Figura 5: Dispersión de Puntos

En este gráfico, encontramos tres tipos de datos con los que trabajamos, y son los siguientes:

1.- Existe un punto que parece tener el precio negativo, para confirmarlo, realizamos una descripción de la variable, la cual nos entregó que tenemos precio mínimo de -1. Este análisis no nos indica si este es un registro único o existen más casos para este precio, por lo que hicimos una búsqueda de todos los productos que tienen un precio menor a cero, de lo que obtuvimos que efectivamente ocurre para solo un registro, cuyo id de producto es 2973. Para solucionar esto, lo que hicimos fue sacar el promedio del precio de este producto y este promedio asignarlo al precio de la venta con precio -1.

2.- Parece existir un punto que se aleja de los demás en cuanto al precio del item, hicimos una búsqueda de todos los precios mayores a $100,000, de lo que obtuvimos que era un único punto, luego a partir del id de aquel producto con precio mayor a $100,000 buscaremos si existe otra venta para este producto, de lo que obtuvimos que solo existió una venta.

3.- En cuanto al eje de cantidad vendida del ítem, observamos que parecen haber dos registros por sobre los 750 en cantidad. Hicimos la búsqueda y efectivamente solo existen dos productos que tuvieron más de 750 unidades vendidas en un día. Para ello, haremos un análisis para cada producto.

- Producto id 20949:
  Este tiene un máximo de 1000 ventas, un promedio de 5 y el tercer cuartil esta en 7, por lo que el dato 1000 claramente es un Outlier y lo eliminaremos.
- Producto id 11373:
  Este tiene un máximo de 2169 ventas, un promedio de 14 y el tercer cuartil esta en 8, por lo que el dato 2169 también es claramente un Outlier y lo eliminaremos.

A continuación veremos la misma figura 5 pero ahora sin los Outliers que eliminamos:

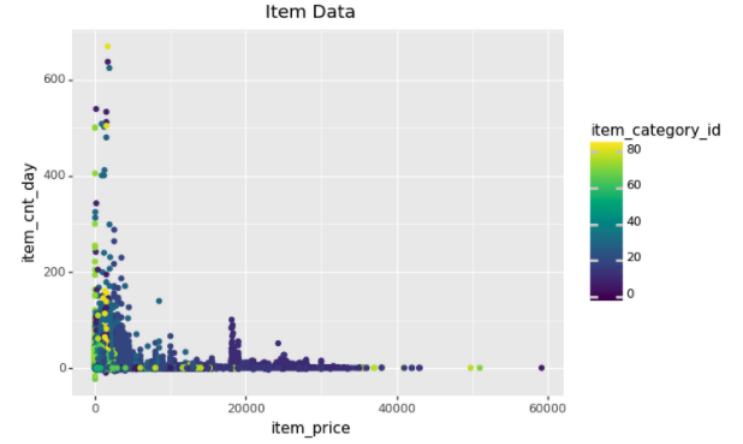

Figura 6: Dispersión de Puntos sin Outliers

Este último gráfico, fue el resultado de la primera limpieza de los datos, en el cual se puede apreciar la distribución de los datos en base a las variables de precio y demanda diaria. Y estos serán la base al análisis de valores extremos solicitado en el siguiente inciso.

b) Identifique valores extremos utilizando el método Local Outlier Factor en R. Y utilice la libreria DMwR y la función lofactor(). ¿Qué haría usted con estos datos?.

Solución:
Primero que nada, es necesario aclara que este ejercicio no se realizó con el enfoque indicado en el enunciado, por el costo computacional que este implicaba. Es por esto que la detección de los Outliers se realizó ocupando el **rango intercuartílico**.

Existen distintos tipos de valores atípicos, los leves y los extremos, ambos se rigen bajo una fórmula muy similar, pero con un pequeño matiz, el cual es, que tan lejos están estos valores antes del primer cuartil y cuan lejos están después del tercer cuartil. Luego sea:

- $Q_1 := Primer\ Cuartil$
- $Q_3 := Tercer\ Cuartil$
- $IQR := Rango\ Intercuartlico$

Un punto(p) cualquiera será un valor atípico extremo si cumple una de estas condiciones:

- $p < Q_1 - 3 * IQR$
- $p > Q_3 + 3 * IQR$

Los atributos con los que seleccionaremos los outliers, son la cantidad de ventas diarias por tienda y el precio en las tiendas. Luego de seleccionar los outliers para cada producto se eliminaran, a continuación está el proceso realizado en código, primero le quitamos los outliers en la categoría de Cantidad de ventas diarias, y luego sin datos atípicos para la cantidad de ventas diarias, se quitaran los outliers para el precio de los artículos.

Code:

```
tiendas=data.groupby('shop_id')['shop_id'].count().index
minimo=tiendas[0]

q3=data[data.shop_id == minimo]['item_cnt_day'].describe().values[6]
q1=data[data.shop_id == minimo]['item_cnt_day'].describe().values[4]
IQR=q3-q1
outlier_sup=q3+3*IQR
outlier_inf=q1-3*IQR

data_sin_Outliers=data[data.shop_id == minimo][data.item_cnt_day>=outlier_inf]
[data.item_cnt_day<=outlier_sup].copy()
tiendas=np.delete(tiendas, 0)

for i in tiendas:
    datos=data[data.shop_id == i]['item_cnt_day'].describe()

    q3=datos.values[6]
    q1=datos.values[4]
    IQR=q3-q1

    outlier_sup=q3+3*IQR
    outlier_inf=q1-3*IQR

    aux=data[data.shop_id == i][data.item_cnt_day>=outlier_inf]
    [data.item_cnt_day<=outlier_sup].copy()

    data_sin_Outliers=pd.concat([data_sin_Outliers, aux], axis=0)
```

Luego de estas lineas de código, queda almacenado en la variable data_sin_Outliers todos los productos que no tengan outliers bajo las condiciones mencionadas anteriormente en la categoría de cantidad de ventas por tienda, a continuación se presenta el código para eliminar los outliers por el atributo de precio de cada artículo:

Code:

```
tiendas=data.groupby('shop_id_categorico')['shop_id_categorico'].count().index
minimo=tiendas[0]

datos=data_sin_Outliers[data_sin_Outliers.shop_id_categorico == minimo]
['item_price'].describe()
q3=datos.values[6]
q1=datos.values[4]

IQR=q3-q1
outlier_sup=q3+3*IQR
outlier_inf=q1-3*IQR

data=data_sin_Outliers[data_sin_Outliers.shop_id_categorico == minimo]
[data_sin_Outliers.item_price>=outlier_inf]
[data_sin_Outliers.item_price<=outlier_sup].copy()
tiendas=np.delete(tiendas, 0)

for i in tiendas:
    datos=data_sin_Outliers[data_sin_Outliers.shop_id_categorico == i]
    ['item_price'].describe()

    q3=datos.values[6]
    q1=datos.values[4]
    IQR=q3-q1

    outlier_sup=q3+3*IQR
    outlier_inf=q1-3*IQR

    aux=data_sin_Outliers[data_sin_Outliers.shop_id_categorico == i]
    [data_sin_Outliers.item_price>=outlier_inf]
    [data_sin_Outliers.item_price<=outlier_sup].copy()
    data=pd.concat([data, aux], axis=0)
```

Podemos observar como cambiaron los gráficos luego de sacar los outliers. Los primeros representan la transformación de los datos respecto de la categoría de cantidad de ventas diarias. En segundo lugar se presenta la evolución respecto de los precios del item.

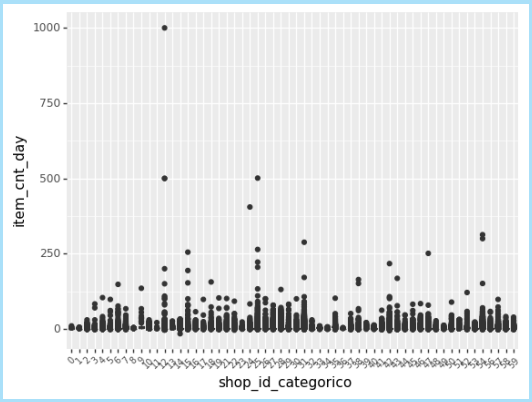

Figura 7: Boxplots por tienda y cantidad antes de filtrar outliers

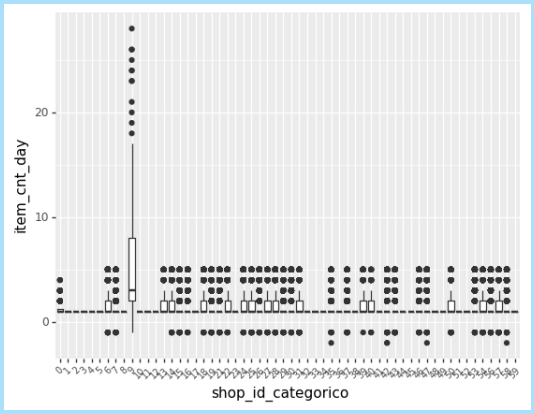

Figura 8: Boxplots por tienda y cantidad después de filtrar outliers

De estos podemos observar que, luego de la eliminación de outliers, quedaron productos con cantidades negativas, las que serán eliminadas en el siguiente proceso.

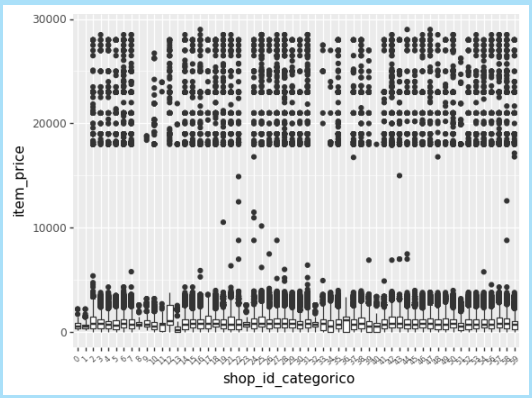

Figura 9: Boxplots por tienda y precio antes de filtrar outliers

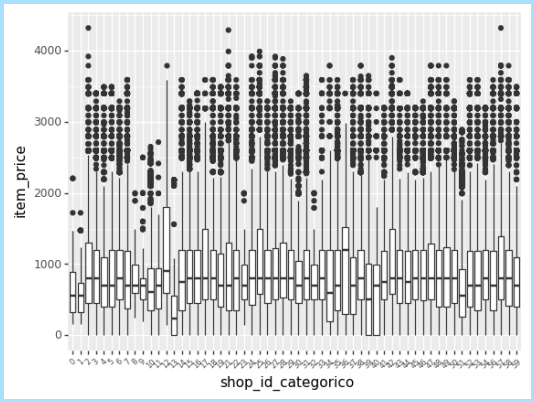

Figura 10: Boxplots por tienda y precio después de filtrar outliers

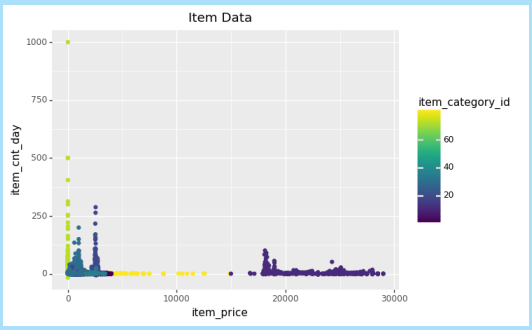

Figura 11: Dispersión puntos antes de filtrar outliers

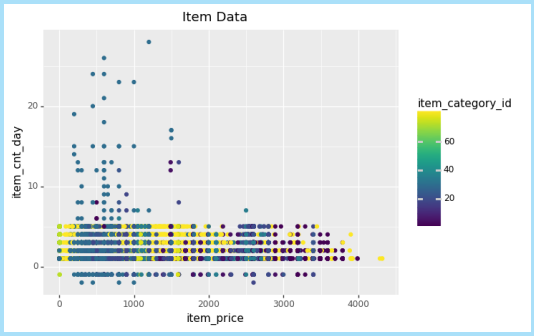

Figura 12: Dispersión puntos después de filtrar outliers

En la tabla representada a continuación, podemos observar como disminuyeron los datos después de cada proceso de eliminación de outliers.

Momento	**Registros**
Data inicial	264494
Data despues de detección Outliers en cada tienda por cantidad	239130
Data despues de detección Outliers en cada tienda por precio	233770

De lo que podemos concluir que eliminamos aproximadamente 30 mil registros.

c) Realice un análisis de correlaciones entre la cantidad de ventas diarias y el resto de las variables en el conjunto de datos.

Solución:
Para esta parte del trabajo, se requirió seleccionar un solo item, el cual fue seleccionado bajo el criterio de mayor cantidad de escenarios, de lo que resultó que el artículo con el id 20949 fue el que coincidía con aquel criterio, ya que posee una cantidad de 31340 registros, este item, pese a la limpieza realizada anteriormente, continuó siendo el item con la mayor cantidad de registro, no obteniendo una reducción. Además, este análisis de correlación va encaminado al desarrollo del modelo de regresión a realizar en la siguiente parte, por lo que se requiere estructurar la data, de tal forma que se puede analizar la correlación de las variables del modelo con respecto a la demanda del ítem seleccionado. Es por ello que, en vez de utilizar los 21807 ítems distintos, para incluirlos como columnas nuevas en el modelo, solo se utilizaran los precios de los 50 ítem mas repetidos.

(1) Se separó la data en entrenamiento y prueba.
Para ello se buscó a partir de que índice los registros tienen por fecha octubre del 2015, separando los datos de test desde el índice encontrado hasta el final de los datos y los datos de train fueron los primeros índices hasta el índice encontrado menos 1.

(2) Se seleccionaron los item_id de tal forma que se pueda ir avanzando por el vector item_N e ir incluyendo primero los ítems con mayor frecuencia en la data, donde el número de items con alta frecuencia que se agregarían está determinado por aquellos que posean más de 1.000 registros.

(3) También se filtraron los datos que posean al menos 1000 filas para así asegurar tener data suficiente para los modelos.

Code:

```
1
2  #Listado de todos los items posibles
3      item_N=data  %>%
4        group_by(item_id)  %>%
5        summarise(no_rows = length(item_id))
6      item_N=item_N[item_N$no_rows>1000,]
7      #Ordena los item de mayor frecuencia a menor frecuencia
8      item_N <- item_N[order(-item_N$no_rows),]
9      item_N<-item_N['item_id']
```

(4) Se creo un data frame con los datos del item a analizar, que en este caso es el de id 20949

Code:

```
1  I="20949"
2
3  Todo_I<-train[item_id==I ,
4                .(date=date,
5                  item_cnt_day=item_cnt_day,
6                  shop_id=shop_id,
7                  item_price_Y=item_price,
8                  item_category_id=item_category_id
9                )]
```

(5) Comienza un loop en donde se van incorporando todos los precios de los ítems seleccionados en (2)

Code:

```
1
2  #Agrega una columna al data frame con el nombre del id de los items
3
4  for (id in item_N$item_id ){  #as.character(L_item)) {
5      #id="5822"
6      if(id!=I){
7
8    x_data=train[item_id==id,
9                 .(date=date,
10                  shop_id=shop_id,
11                  item_price=item_price)]
12   # Realiza un Left merge con date y shop_id
13   Todo_I<-merge(x=Todo_I,y=x_data,by=c('date','shop_id')
14     ,all.x=TRUE)
15   Todo_I[,id]<-lapply(Todo_I$item_price, function(x)
16   ifelse(is.na(x),0, log(x) ) )
17   Todo_I$item_price=NULL
18   Todo_I$item_price.y=NULL
19   Todo_I$item_price.x=NULL
20   }#if not id =I
21   }#firts For
```

(6) Finalmente se crea el archivo csv, con el cual se trabajará para la generación de los modelos en la siguiente parte de la prueba.

Con respecto al nuevo archivo csv creado, se hace un preprocesamiento de los datos generados, en donde se aplica un reordenado de las columnas y definición del tipo de variables, para que el código interprete de mejor forma la data.

Para responder a la pregunta que se indica en esta parte, teniendo el archivo csv que utilizaremos para el resto de la prueba listo, realizamos el cálculo de las correlaciones con el método de Pearson y entregamos la información gráficamente:

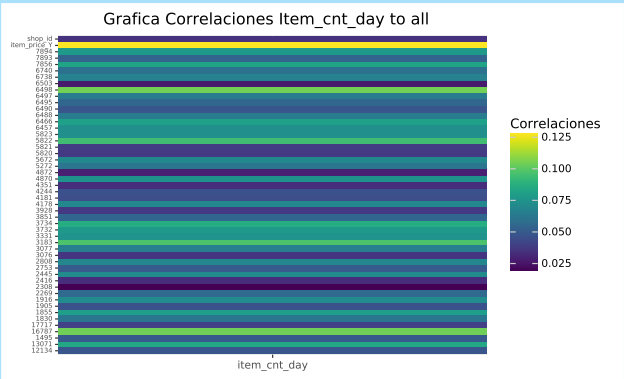

Figura 13: Gráfica Correlaciones

Del gráfico, podemos observar que item_cnt_day con la variable que más tiene correlación, es con el precio, lo que sorprende es que esta correlación es positiva, lo que puedes ser que sea producto del comportamiento del articulo en cuestión.

2. Parte

Ajuste un modelo de regresión lineal múltiple para cada producto (item_id). Considere una muestra de validación de las últimas 4 semanas.

$$\ln(\textbf{item\_cnt\_day}[shop\_id\_\textbf{i}]) = \beta_0 + \beta_1 \ln(\textbf{item\_price\_1}[shop\_id\_\textbf{i}]) + \beta_2 \ln(\textbf{item\_price\_2}[shop\_id\_\textbf{i}]) + \cdots + \beta_k \ln(\textbf{item\_price\_K}[shop\_id\_\textbf{i}]) + \alpha_1'' item\_category\_id'' + \alpha_2 shop\_id + \in$$

a) Ajuste un modelo clásico de regresión lineal múltiple. Implemente el método de selección de variables stepwise forward. Comente sus resultados en comparación con el modelo ajustado con todas las variables versus el modelo reducido vía selección stepwise forward. Evalúe la multicolinealidad del modelo obtenido vía selección stepwise forward. Calcule el MAPE (mean absolute percentage error).

Solución:

Estructuración de los datos:
Para poder desarrollar cada uno de los modelos a continuación, se realizó un ajuste a la variable shop_id, para generar variables dummy, debido que se debían realizar operaciones matriciales para generar el modelo Stepwise Forward y estas operaciones requieren argumentos numéricos, y la variable shop_id es tipo factor, lo que no permite su operación. Para generar una comparación apropiada de los modelos, se decidió utilizar los mismos datos, pese a que al realizar este ajuste se requiriera eliminar observaciones y variables (la la cantidad de variables dummy se redujo, ya que en la data de entrenamiento y la data prueba no coincidían los id de shop en 13 casos; lo que a su vez produjo una disminución de observaciones). Cabe destacar que, en un inicio se consideró utilizar solo los datos ajustados para el modelo Stepwise Forward, pero generó una gran inquietud al momento de comparar los modelos al no ocupar la misma base. Luego al utilizar los mismos datos, efectivamente si se compara de tal forma el resultado cambia, por lo que se optó por utilizar los datos ajustados para la generación de todos los modelos.

En esta parte se pide realizar un modelo de regresión lineal múltiple para cada ítem, pero debido a la magnitud de datos, como se menciono en la parte anterior, solo se realizará un modelo para el ítem con mayor frecuencia, el cual es el ítem con el id 20949, el cual posee una frecuencia de 31340, previo a la filtración de los datos. Por otra parte, se dejó fuera la variable item_category_id debido a que esta no sufría variaciones.

a) **Ajuste un modelo clásico de regresión lineal múltiple.**
Para este modelo, se usarán todas las variables.
Dentro de la construcción del modelo de regresión lineal múltiple, se utilizó cross validation para ajustar los hiperparámetros del modelo, con 10 dobladas y un tuneLength de 5.

Code:

```
### Build the full model or classic ####
set.seed(123)
lm_model <- train(
  log(item_cnt_day) ~., data = data_train, method = "lm",
  trControl = trainControl("cv", number = 10),
  tuneLength = 5
)
```

El modelo consideró 100 variables, esto tomando en cuenta que al ajustar shop_id como un factor, se considera cada tienda como una variable binaria.

Posteriormente para validar la eficiencia del modelo, se realizó una validación con la data_test, la cual como se ha explicado en el transcurso de este documento corresponde a los datos del mes de octubre del año 2015, es decir las ultimas cuatro semanas de la data.

Code:

```
1
2  ### Make predictions
3  predictions_lm_model <- lm_model  %>% predict(data_test)
4
5  predictions_lm_model<-sapply(predictions_lm_model,
6                               function(x) ifelse(x==0,
7                                                  0.000001, x ))
8
9  ### Model prediction performance
10 score_lm_model=data.frame(
11   RMSE = RMSE(predictions_lm_model, data_test$item_cnt_day),
12   Rsquare = R2(predictions_lm_model, data_test$item_cnt_day),
13   MAPE= MAPE( predictions_lm_model,sapply( data_test$item_cnt_day,
14                                    function(x) ifelse(x==0,
15                                                       0.000001, x )))
16 )
```

b) **Implemente el método de selección de variables stepwise forward.**

Para realizar la selección de variables stepwise forward, se crearon dos modelos, uno con todas las variables y uno sin ninguna, llamados comúnmente como modelo saturado y modelo nulo, respectivamente.

Esto se realizó de la siguiente forma

Code:

```
1      ## Seleccion Stepwise forward ####
2
3      null<-lm(log(item_cnt_day)~1, data=data_train)
4      full<-lm(log(item_cnt_day)~., data=data_train)
```

El paso que sigue, es el proceso en el cual se añaden las variables más representativas al modelo, lo cual se realiza por medio de la siguiente porción de código, en donde la función step, para que tenga aún un mejor desempeño de como sería al utilizar forward, se decidió utilizar both, que es una combinación de forward y backward.

Code:

```
1
2  ## Seleccion Stepwise forward ####
3  output2a<-step(null, scope = list(upper=full), data=data_train, direction="both")
4  summary(output2a)
5  model2a =  output2a$call$formula
```

c) Una vez finalizada la operación se generó el siguiente modelo:

Code:

```
1
2       log(item_cnt_day) ~ shop_id.38 + shop_id.34 + shop_id.15 +
3       X6498 + X16787 + shop_id.28 + X4178 + X3183 + shop_id.21 +
4       shop_id.57 + shop_id.22 + X3734 + X13071 + shop_id.26 + shop_id.47 +
5       shop_id.35 + shop_id.53 + X5822 + X3077 + shop_id.42 + X1855 +
6       X1830 + X6738 + X5272 + shop_id.14 + shop_id.6 + shop_id.37 +
7       X3331 + X6457 + shop_id.58 + shop_id.56 + X2808 + shop_id.39 +
8       X5823 + shop_id.16 + shop_id.18 + shop_id.7 + shop_id.46 +
9       shop_id.24 + shop_id.50 + shop_id.19 + shop_id.31 + X5672 +
10      X3851 + X1916 + X1905 + X7894 + X4870 + X6497 + X6466 + X7856 +
11      X2445 + X1495 + X7893 + X4244 + X12134 + X6488 + X2269 +
12      X4872 + X5820 + X6503 + X6740
```

Se aprecia que este método dejó fuera varias variables, entre ella la variable item_price_Y, es decir, el precio del item a predecir (producto de id 20949).

d) Evalúe la multicolinealidad del modelo obtenido vía selección stepwise forward.

Para evaluar la multicolinealidad del modelo, se calculó el número de condición (NC). Para ello se realizó la composición X'X y a partir de ella se calculó la descomposición de valores singulares de aquella matriz (svd), que descompone la matriz en 3 submatrices. En particular nos interesará la matriz "d"que corresponde a la matriz que entrega los valores únicos de X. Finalmente, se calcula el número de condición como la raiz cuadrada de la razón entre el máximo y el mínimo valor en la matriz d. A continuación se presenta el código utilizado:

Code:

```
1       # Colinealidad
2       # Para modelo con las variables seleccionadas por metodo del Stepwise Forward
3
4       X_aux_SF<-test.data[,-c(1,2,4,6,8,10,11,25,33,41,43,44,49,50,51,54,71,73,74,77,78,80,
5
6       #Descomposicion de la matriz X'X para modelo Stepwise Forward
7       XtX_SF <- t(X_aux_SF) %*% as.matrix(X_aux_SF)
8       s_SF <- svd(XtX_SF)
9
10      # Si NC >25 -> colinealidad
11      #NC para Stepwise Forward
12      NC_SF <- sqrt(max(s_SF$d)/min(s_SF$d))
```

Luego, el **número de condición (NC)**, resultó ser igual a **100.039.988**, como el resultado fue mayor a 25, se puede decir que el modelo Stepwise Forward presenta colinealidad.

e) **Calcule el MAPE (mean absolute percentage error).**

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596

Se aprecia en la tabla anterior que, tanto para el RMSE como el MAPE el modelo que tuvo un mejor rendimiento fue el modelo clásico y que para el R^2 se obtiene un mejor resultado con el modelo Stepwise Forward. Cabe destacar, que la diferencia con la que se decide cual modelo fue mejor para los 3 índices estuvo en el 4 decimal.

b) Ajuste un modelo de regresión ridge. Compare sus resultados con los obtenidos en II.a. En el ajuste considere el tunning de hiperparámetros mediante Validación cruzada. Gráfique las trazas ridge.

Solución:
Para la generación del modelo de regresión ridge, por medio de la validación cruzada se ajustaron los hiperparámetros del modelo, con 5 dobladas y para ajustar el hiperparámetro lambda se utilizó búsqueda de grilla. Se definió una amplitud de búsqueda de λ desde 0.001 a 1000 ($0{,}001 \leq \lambda \leq 1000$). Este ajuste se puede visualizar en el siguiente bloque de código.

Code:

```
### Build the model Ridge

lambda = 10^seq(-3, 3, length = 100)

set.seed(123)
ridge <- train(
  log(item_cnt_day) ~., data = data_train, method = "glmnet",
  trControl = trainControl("cv", number = 5),
  tuneGrid = expand.grid(alpha = 0, lambda = lambda)
)
```

Este modelo utilizó todas las variables provistas.

A continuación se presenta el gráfico las trazas ridge.

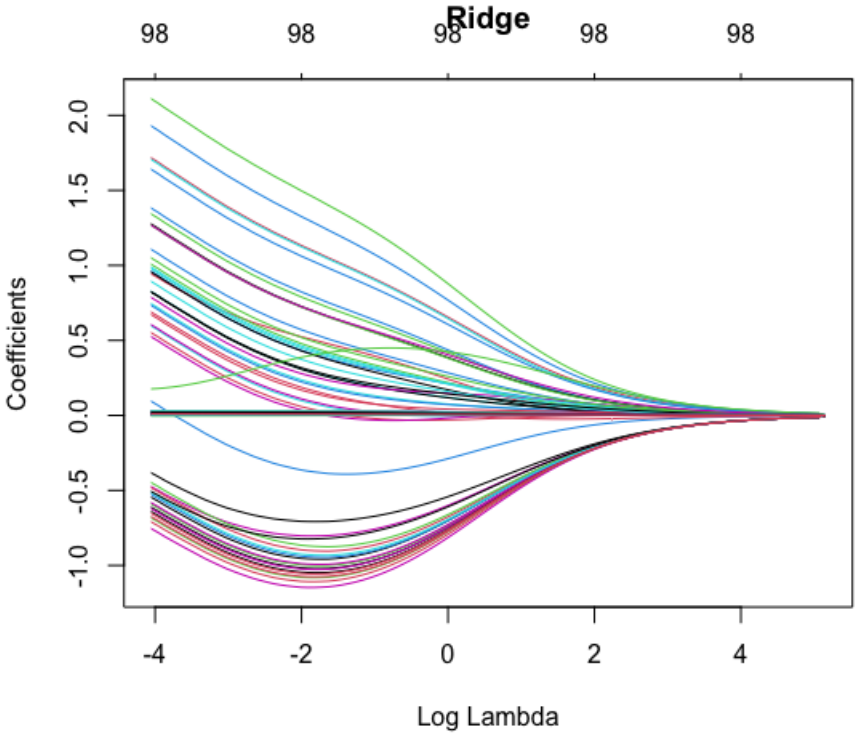

Figura 14: Gráfico de Trazas Ridge

De la gráfica anterior se aprecia que a medida que va aumentando el valor de lambda (log de lambda) los coeficientes comienzan a regularizarse. Se puede notar que a partir de un lambda de aproximadamente 1 los coeficientes comienzan estabilizarse de forma más notoria y ya a partir de un lambda de 3 la mayoría de los coeficiente se han estabilizado.

En la siguiente tabla se presentan distintas métricas del desempeño de cada uno de los modelos realizados hasta el momento.

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596
Ridge	1.809513	0.238284	0.5924713

En este caso, el único valor que mejoró al realizar el modelo ridge fue el MAPE.

c) Ajuste un modelo de regresión lasso. Compare sus resultados con los obtenidos en II.a y II.b. En el ajuste considere el tunning de hiperparámetros mediante Validación cruzada. Gráfique las trazas lasso.

Solución:

El modelo de regresión Lasso se ajustó de la misma manera que en el modelo anterior (Parte 2 b), con la diferencia de que ahora alpha es 1. Este ajuste se puede visualizar en el siguiente bloque de código.

Code:

```
 ### Build the model Lasso

lambda = seq(-6,6,0.1)

set.seed(123)
lasso <- train(
  log(item_cnt_day) ~., data = data_train, method = "glmnet",
  trControl = trainControl("cv", number = 5),
  tuneGrid = expand.grid(alpha = 1, lambda = lambda)
)
```

Este modelo utilizó todas las variables provistas.

A continuación se presenta el gráfico las trazas Lasso.

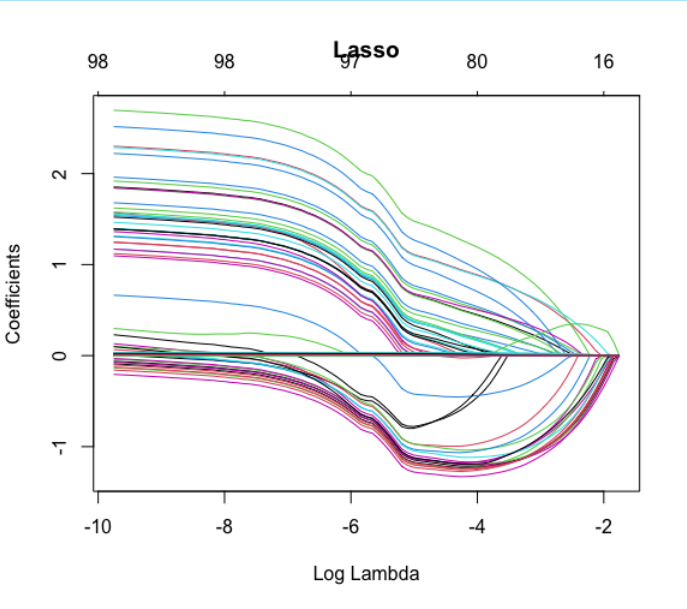

Figura 15: Gráfico de Trazas Lasso

Se aprecia que Lasso logra estabilizar sus coeficientes a un menor valor de lambda (log de lambda) en relación al modelo Ridge, donde ya a partir de un valor de lambda de -2 sus coeficientes ya se encuentran completamente estabilizados.

El resultado de la predicciones realizadas por este modelo se reflejan en la tabla a continuación, junto con los resultados de los anteriores modelos.

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596
Ridge	1.809513	0.238284	0.5924713
Lasso	1.808392	0.2403912	0.5932643

Se aprecia que el desempeño de este modelo presentó una mejora respecto al modelo Ridge, en cuanto a el error RMSE y el R^2, pero en cuanto al MAPE fue livianamente inferior por 0.001. En comparación a los demás modelos solo presenta un mejor MAPE.

d) Ajuste un modelo de regresión elasticnet. Compare sus resultados con los obtenidos en II.a, II.b. y II.c. En el ajuste considere el tunning de hiperparámetros mediante Validación cruzada. Gráfique las trazas elasticnet.

Solución:
El modelo de regresión Elastic Net se ajusto con 10 dobladas y un tuneLength de 5. Además para configurar λ se tiene una amplitud de búsqueda de -6 a 6 particionada en 0,1. Este ajuste se puede visualizar en el siguiente bloque de código.

Code:

```
lambda = seq(-6,6,0.1)
### Build the model Elastic Net
set.seed(123)
elastic <- train(
  log(item_cnt_day) ~., data = data_train, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10
)
```

Este modelo utilizó todas las variables provistas.

A continuación se presenta el gráfico las trazas.

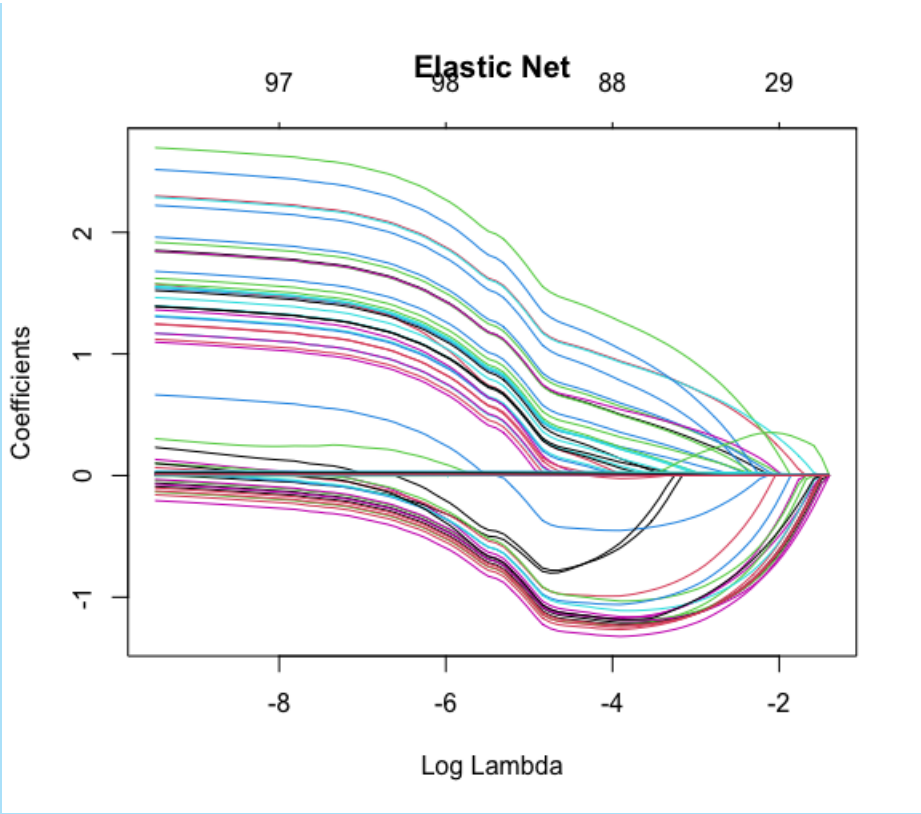

Figura 16: Gráfico de Trazas Elastic Net

A partir del gráfico, se puede observar que tiene un comportamiento similar al que tuvo el modelo Lasso, en el cual los coeficientes del modelo se ajustaron a partir de un lambda (log de lambda) de -2 aproximadamente. Con la sutil diferencia, de que la estandarización de los coeficientes fue con un valor de lambda levemente mayor que como lo hicieron para el modelo Lasso. Lo que tiene sentido, considerando que el modelo Elastic Net contiene tanto al Modelo Lasso y al Modelo Ridge cuando alpha es 1 y 0 respectivamente. Y este leve desplazamiento podría deberse a que el modelo Ridge estabiliza sus coeficientes para valores de lambda mayores que los que presenta el modelo Lasso.

Por otra parte, dada la semejanza con las trazas del modelo Lasso, se podría inferir que el modelo Elastic Net podría tener su parámetro alpha igual 1 o ser cercano a este. Y efectivamente el mejor valor de alpha para este modelo fue de 0,7.

El desempeño de este modelo se reflejan en la tabla a continuación, junto con los resultados de los anteriores modelos

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596
Ridge	1.809513	0.238284	0.5924713
Lasso	1.808392	0.2403912	0.5932643
Elastic Net	1.806857	0.2409718	0.5955038

De esta se concluye que, presenta mejoras con respecto a Ridge y Lasso en cuanto a el error RMSE y el R^2, pero con un $MAPE$ inferior a ambos modelos. Con respecto a todos los modelos, presenta el mejor R^2, y errores relativamente buenos en comparación a los demás. De modo que si se tuviera que elegir un modelo dependerá de a que factor se le dará mayor importancia, por ejemplo, si lo que importa más para el tomador de decisiones es el R^2 obtenido, el mejor modelo será el Elastic Net.

e) Concluya cuál es el modelo que genera el mejor MAPE de validación.

Solución:
Tal y como se ha presentado en las tablas de rendimiento de los distintos modelos, tenemos que el MAPE fue:

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596
Ridge	1.809513	0.238284	0.5924713
Lasso	1.808392	0.2403912	0.5932643
Elastic Net	1.806857	0.2409718	0.5955038

Donde el modelo que generó un mejor MAPE es el modelo Ridge con un MAPE de 0,5924713.

f) Interprete sus resultados en función de las elasticidades de precio. ¿Cómo interpretaría los resultados para un producto en particular?

Solución:
Para realizar una comparación exclusivamente entre las variables item precio y la demanda de un solo producto. Generamos un modelo de regresión, en el cual se incorporó solamente la covariable Item_price_Y (la cual refleja el ln() del precio del item_Y a predecir). Este modelo de regresión lineal nos entregó un coeficiente de 0.08592911 para la única covariable (Item_price_Y), lo que nos

indica la elasticidad precio de la demanda, este coeficiente tuvo un valor de 0.08592911. Lo que se interpreta como: para un aumento en una unidad del precio del artículo, la demanda refleja un aumento de 0.08592911. Sin embargo, el R^2 obtenido es sumamente bajo, resultando ser tan solo un 0.003917037, lo que nos indica que generar un modelo con tal variable independiente no logra explicar o no es relevante para explicar por si sola a la variable dependiente. Por lo que podemos concluir que la elasticidad precio para el producto 20949, no es suficientemente representativa para poder obtener resultados aceptables de la demanda de aquel artículo.

A continuación, se presenta una tabla de resumen, que contiene todas las regresiones realizadas y los desempeños respectivos.

	Puntaje		
Modelo de regresión lineal múltiple.	**RMSE**	R^2	**MAPE**
Clásico	1.8069	0.2407914	0.5960582
Stepwise forward	1.806851	0.2406399	0.5961596
Ridge	1.809513	0.238284	0.5924713
Lasso	1.808392	0.2403912	0.5932643
Elastic Net	1.806857	0.2409718	0.5955038
Elasticidad Precio	**1.899468**	**0.003917037**	**0.3734148**

3. Anexo

Cuadro 1: Variables Categóricas

	shop_id	**item_id**	**item_category_id**
n.non.miss	2935849	2935849	2935849
n.miss	0	0	0
n.miss.percent	0	0	0
n.unique	60	21807	84
cat_1	31	20949	40
freq_1	235636	31340	564652
cat_2	25	5822	30
freq_2	186104	9408	351591
cat_3	54	17717	55
freq_3	143480	9067	339585
cat_4	28	2808	19
freq_4	142234	7479	208219
cat_5	57	4181	37
freq_5	117428	6853	192674
cat_6	42	7856	23
freq_6	109253	6602	146789
cat_7	27	3732	28
freq_7	105366	6475	121539
cat_8	6	2308	20
freq_8	82663	6320	79058
cat_9	58	4870	63
freq_9	71441	5811	53845
cat_10	56	3734	65
freq_10	69573	5805	53227

Cuadro 2: Atributos numéricos

	date_block_num	**item_price**	**item_cnt_day**
non-missing	2935849	2935849	2935849
missing	0	0	0
missing percent	0	0	0
unique	34	19993	198
mean	14.57	890.85	1.24
min	0	-1	-22
p1	0	5	1
p5	1	99	1
p10	2	149	1
p25	7	249	1
p50	14	399	1
p75	23	999	1
p90	28	1999	2
p95	31	2690	2
p99	33	5999	5
max	33	307980	2169