ITBA

16.32 - Redes Neuronales en Bioingeniería

Bioingeniería

TP N°2: Rossmann Store Sales

Profesores:

Carlos Selmo

Rodrigo Cardenas

Alumno:

Juan Pablo Touron

Fecha de entrega: 28 de Octubre de 2022

Introducción

Rossmann es una cadena de *drug stores* con más de 3.000 tiendas en 7 países de Europa. Se presentó una competencia en *Kaggle* con el objetivo de pronosticar las ventas de las 6 próximas semanas. Las ventas están afectada por diversos factores, entre ellos, fecha, feriados escolares y estatales, promociones, ubicación de la tienda, entre otros. El objetivo de este trabajo es entonces predecir las ventas de distintas tiendas de Rossmann, en base a una gran cantidad de datos sobre muchas de estas variables y utilizando conceptos vistos en las clases de optimizadores y regularización. Se busca minimizar el RMSPE (*Root Mean Squared Percentage Error*) del cual se habla más en detalle más adelante.

Procesamiento de datos

En primer lugar, se leyeron los datos referentes a los *stores*, a los distintos estados, y al clima. En particular, hay 7 sets de datos a tener en cuenta:

- *train*: tiene la información sobre N° de tienda, día de la semana, fecha, volumen de ventas, cantidad de clientes, si estaba abierto, si había promoción, y si había feriado estatal o escolar que será utilizada para entrenar los modelos.
- *test*: cuenta con la misma información que *train* pero sin las columnas de ventas y clientes que justamente es lo que hay que estimar.
- *store*: tiene información similar a *train* y algunas columnas adicionales sobre las distintas tiendas.
- *store_states*: indica a qué estado pertenece cada una de las 1115 tiendas.
- *state_names*: contiene códigos de identificación de cada estado.
- *googletrend*
- *weather*

Lo primero que se verificó fue que no hubiera datos faltantes. Se aplicaron distintas estrategias de reemplazo para cada variable con datos faltantes. Se listan a continuación:

- Para **fechas faltantes** o **anteriores a 1990** se reemplazó la fecha por Enero de 1990
- Para **distancias faltantes** se reemplazó por la distancia máxima
- Para **datos faltantes referentes al clima** se reemplazo por parámetros que indicaran un clima tranquilo (mucha visibilidad, poco viento, pocas nubes,soleado)

Luego, se unen todas las tablas en 2: una para *train* y otra para *test*. En ambas se encuentra toda la información referente a los últimos 5 sets de datos pero en la primera se incluye los datos para entrenamiento y en la segunda, para testeo.

Se codifican las variables categóricas mediante el método *LabelEncoder()* de *SciKit-Learn*. Lo que hace esta función es convertir los valores de una variable categórica en valores de 0 al número de valores distintos menos 1. Por otra parte, se estandarizan las variables continuos mediante *StandardScaler()*, también de *SciKit-Learn*, que resta la media y divide por el desvío estándar.

Con esto concluye el procesamiento de datos y se pasa a la prueba de distintos modelos.

Modelos e hiperparámetros

Se pueden dividir los modelos en dos grupos: los que utilizan *embeddings* y los que hacen uso de árboles de decisión. Para todos los modelos se probaron distintos valores del *learning rate*. Para los modelos de *embeddings* se probaron distintos valores de L2. Finalmente, para los modelos de árboles se varió el número mínimo de casos por hoja.

Los 5 modelos con mejores resultados se resumen en la siguiente tabla:

Modelo	LR	L2	Epochs	Batch Size	Min Child Samples	Score
MLP con embedding	0.01	0.001	20	256	-	0.12643
MLP con embedding	0.001	0.001	20	256	-	0.17056
Light GBM	0.05	-	-	-	5	0.11227
Light GBM	0.25	-	-	-	5	0.15764
Light GBM	0.1	-	-	-	5	0.11668

Se puede observar que para los modelos de redes neuronales con capas de *embedding* el mejor resultado fue de 0.126 aproximadamente. Pero al modificar un poco el LR, el score varía considerablemente.
Además, en dos de los modelos de árboles se ve un mejor resultado (menor error).
Estos resultados muestran que, si bien los modelos basados en árboles de decisión obtienen *scores* muy satisfactorios, las redes neuronales con capas de *embedding* aparecen como un alternativa viable para resolver problemas en los que intervienen variables categóricas u otro tipo de datos estructurados.

Embeddings obtenidos

Con respecto a los *embeddings* que se obtuvieron, a continuación se muestran algunos casos. Particularmente los de tiendas y días de la semana. Si bien hay varios más, estos

dos son quizás los más interesantes para analizar.

Store

Los *embeddings* de las tiendas se distribuyeron como se ve en la Figura 1.

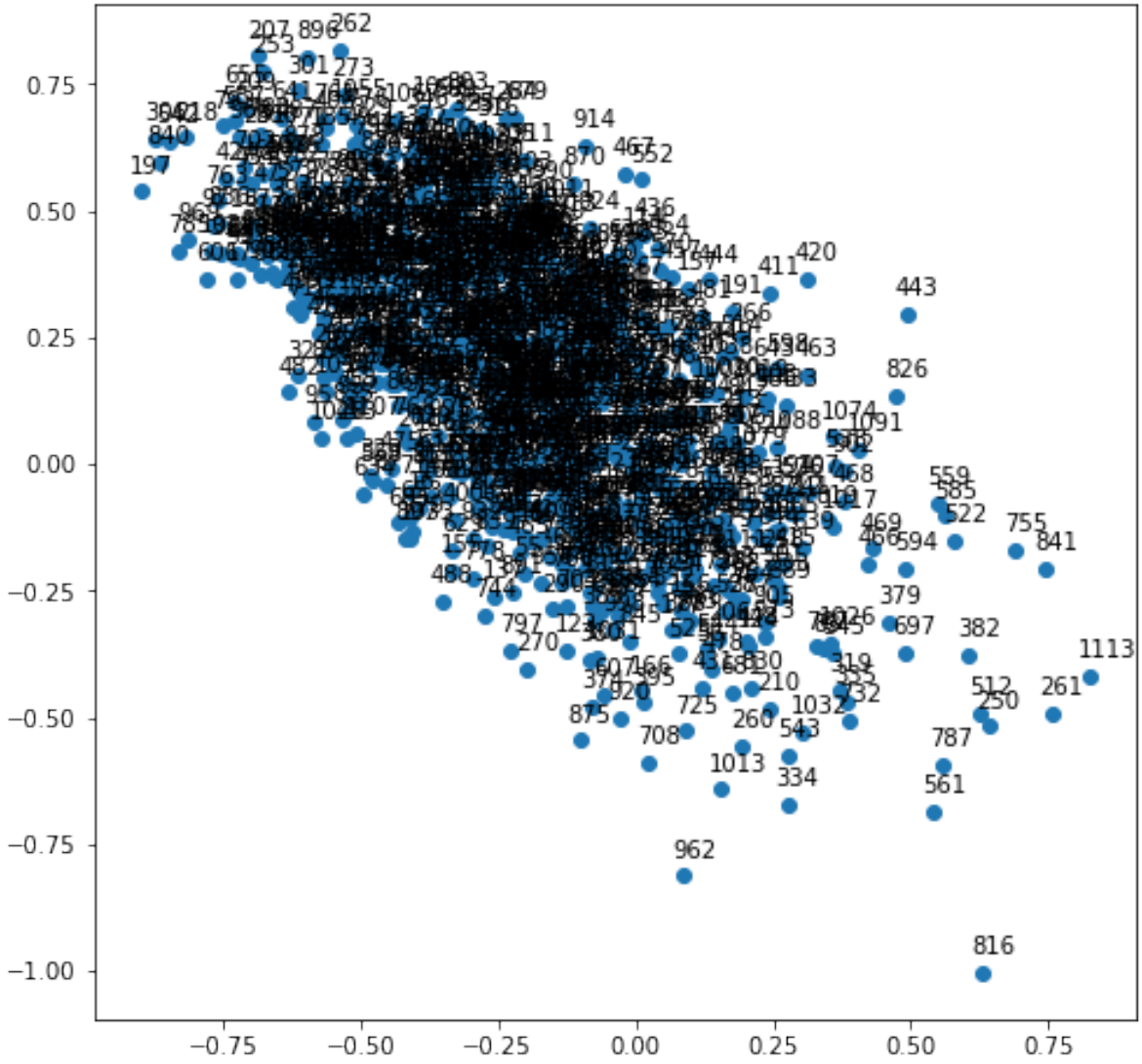

Figura 1: Gráfico de *embeddings* de la categoría *Store*

Se ve, quizás como característica más destacable que la tienda número 816 se encuentra bastante alejada del resto. Esto podría explicarse con la Figura 2. Se ven en azul las ventas de la tienda número 816 y se puede ver que su volumen de ventas es considerablemente menor al del resto (al menos el máximo, habría que ver en total). Esto podría ser una explicación a porque los *embeddings* se distribuyeron así.

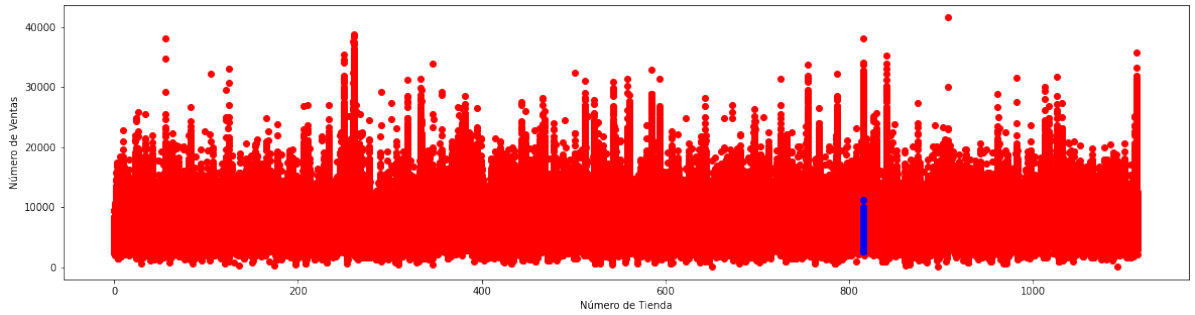

Figura 2: Gráfico de ventas en función de la tienda

Día de la semana

Con respecto a los *embeddings* de la variable *DayOfWeek*, se distribuyeron como muestra la Figura 3.

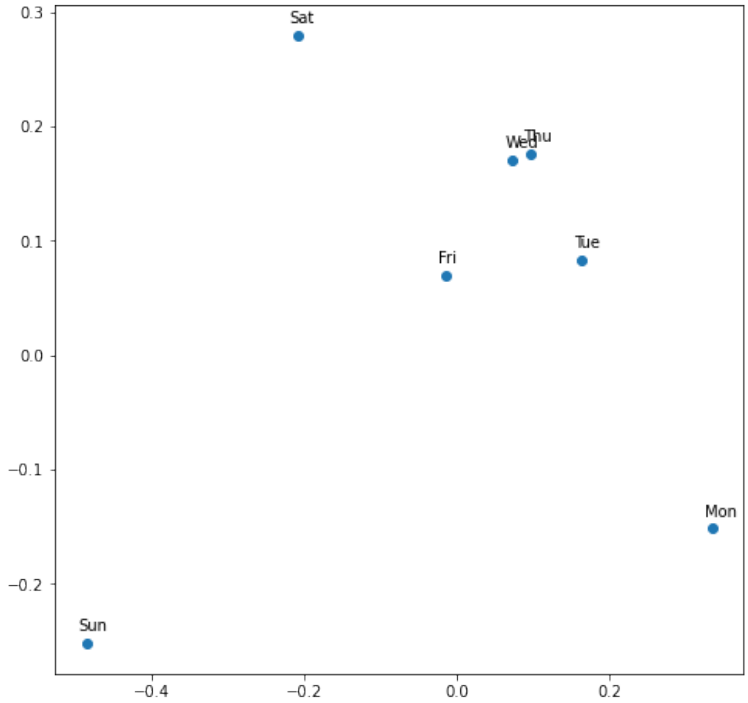

Figura 3: Gráfico de *embeddings* de la categoría *DayOfWeek*

Como dato a destacar, se ve una cercanía muy notoria entre los días miércoles y jueves. En la siguiente tabla se expresa el promedio de ventas por día. Se observa una gran similitud entre los valores de ventas de miércoles y jueves. Sin embargo, mucho más similares son las ventas de lunes y martes (y, sin embargo, sus *embeddings* se encuentran bastante alejados). Por lo cual, la similitud entre los valores *Miércoles* y *Jueves* seguramente se deba a otro motivo.

Día	Ventas
Lunes	8216
Martes	7088
Miercoles	6729
Jueves	6768
Viernes	7073
Sábado	5875
Domingo	8225

Preguntas teóricas

¿Qué son los *entity embeddings* y cómo se relacionan con las variables categóricas?

Los *entity embeddings* son representaciones vectoriales que se utilizan para mapear cada estado de una entidad a un vector de cierta dimensión. La idea es poder convertir variables categóricas a variables continuas, ya que las redes neuronales trabajan mucho mejor con este tipo de variable. Esto último es porque en muchos de los métodos de optimización o minimización del error se utilizan derivadas, por lo que la continuidad de las variables es condición necesaria. Los datos no estructurados son mucho más adecuados para trabajar con redes neuronales, mientras que para los datos estructurados es muy común utilizar otros métodos de aprendizaje automático (por ejemplo, basados en árboles de decisión).
Una estrategia muy utilizada para evitar trabajar con datos estructurados en redes neuronales es la utilización de *one hot encoding*. Este método tiene dos problemas principales:

- Variables de muchos valores implican vectores de muy alta dimensión
- No se tiene en cuenta la relación entre distintos valores de una variable que si bien son distintos podrían ser bastante similares

Los *entity embeddings* solucionan ambos problemas. En primer lugar, no es necesario que el vector de salida tenga tantas posiciones como valores pueda tomar la variable. En segundo lugar, mapea a posiciones cercanas, valores que están cercanos entre sí. Por ejemplo, si se aplica a palabras (lo que se conoce como *word embeddings* y es anterior al desarrollo de *entity embeddings*) la resta de los vectores de Francia y París, es similar a la resta de los vectores de Alemania y Berlín (lo que denota que existe cierta relación, en este caso, de país con capital).

Explique la métrica utilizada en la competencia

La métrica utilizada en la competencia de *Kaggle* es la raíz del error cuadrático medio porcentual o RMSPE, que puede expresarse mediante la siguiente fórmula:

$$RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{y_i - \hat{y}_i}{y_i})^2}$$

Viendo la fórmula anterior, podemos decir que la métrica utilizada da una idea del error cuadrático que tiene cada estimación comparado con el valor real.