Modelado

David Guzmán Leyva - A01706417
Enrique Santos Fraire - A01705746
Leonardo Alvarado Menéndez - A01705998
Oscar Enrique Delgadillo Ochoa - A01705935

Índice

Bitácora de cambios

Versión	Fecha	Autor(es)	Modificaciones
1.0	11/11/2022	David Guzmán Leyva Enrique Santos Fraire Leonardo Alvarado Menéndez Oscar Enrique Delgadillo Ochoa	Línea base

Seleccionar la técnica de modelado

Técnica de modelado

Recordemos que nuestro objetivo es evaluar si en el recorrido a lo largo del día de los usuarios trabajaron por modalidad de Home Office o no, por lo tanto, estamos ante un problema de clasificación, siendo las posibles técnicas de modelado para llegar a una solución:

- Análisis discriminante
- Métodos de inducción de reglas
- Aprendizaje de árboles de decisión
- Redes neuronales
- K Nearest Neighbor
- Razonamiento basado en casos
- Algoritmos genéticos

En nuestro caso hemos seleccionado 2 diferentes técnicas para evaluar su desempeño.

Regresión logística
Este ya que es un modelo fácil de implementar, siendo una opción viable como benchmark para analizar el comportamiento general de nuestros datos.

Aprendizaje de árboles de decisión
Implementar modelos basados en árboles de decisión fue una mejor implementación, debido a que el problema de clasificación que tenemos entre manos funciona mucho mejor que con la regresión logística aplicada anteriormente.

Supuestos de modelado

Para llegar al dataset a utilizar en nuestros modelos se realizó un preprocesado de los datos, filtrando los registros de los usuarios en diferentes horarios, el primero desde la medianoche hasta las siete de la mañana y el segundo desde las 9 de la mañana hasta las 6 de la tarde, esto para suponer el horario de casa y el horario de trabajo.

Una vez que se hizo la separación de los datos, mediante la moda se obtiene la antena que tuvo la mayor cantidad de conexiones en ambos conjuntos de datos, de esta manera tendremos la antena de casa y la antena de trabajo.

Generar el diseño de las pruebas

Diseño de las pruebas

Una vez obtenido el dataset para el modelo se establecieron criterios para las pruebas, entre ellos:

- Contar con al menos un 60% de accuracy tanto en el entrenamiento como en la pruebas

Para el entrenamiento del modelo se definió la separación de datos en train y test de la siguiente manera:

- Train - 80%
- Test - 20%

Construyendo el modelo con el conjunto de train y evaluando con test.

Construir del modelo

Ajuste de parámetros

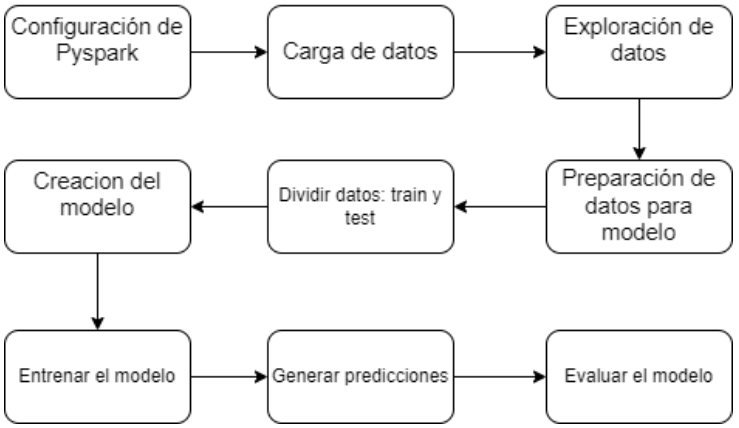
El ajuste de parámetros permite encontrar la mejor configuración de los métodos de optimización ante un determinado problema. Sin embargo, para la implementación de nuestros modelos no se hizo un ajuste demasiado significativo de hiperparametros.

Modelos

- Binomial logistic regression
- Gradient-Boosted Trees (GBTs)
- Xgboost

Descripción del modelo

Pasos cada modelo se siguieron los siguientes pasos generales:



- Binomial logistic regression:
    - Descripción:
      La Regresión Logística Binaria (RLB) se usa cuando se desea conocer la relación entre una variable dependiente cualitativa dicotómica (dependencia) y una o más variables independientes o explicativas, que pueden ser cualitativas (variables sociodemográficas) y/o cuantitativas, con el objetivo de obtener una estimación.
      La elección de este fue por su simplicidad, es un modelo sencillo de entrenar y fácil de implementar, además de que sus tiempos de ejecución suelen ser muy bajos.
    - Stack tecnológico: Apache Spark, Colab, Google Drive
    - Librerías: Pyspark, Spark ML, Mlib.

- Gradient-Boosted Trees (GBTs):
    - Descripción:
      Gradient boosting o Potenciación del gradiente produce un modelo predictivo en forma de un conjunto de modelos de predicción débiles, típicamente árboles de decisión. Construye el modelo de forma escalonada como lo hacen otros métodos de boosting, y los generaliza permitiendo la optimización arbitraria de una función de pérdida diferenciable. Dichos métodos los conocimos el semestre pasado para la solución del reto, por ello decidimos utilizarlo gracias a que podemos hacer que accuracy del modelo mejore cambiando el número de árboles o ajustando otros hiperparametros.
    - Stack tecnológico: Apache Spark, Colab, Google Drive
    - Librerías: Pyspark, Spark ML, Mlib

- XGBoost
    - Descripción:
      XGBoost es un algoritmo de aprendizaje automático basado en un árbol de decisión que utiliza un marco de refuerzo de gradiente. Su funcionamiento es de manera secuencial, este algoritmo posee la combinación óptima de técnicas para optimización entre software y hardware, pues cuenta con la

construcción de árboles paralelizados, poda de los mismos utilizando el primer enfoque de profundidad, reconocimiento de caché y computación fuera del núcleo, regularización y evaluación de overfitting y manejo eficiente de datos perdidos, así como posee la capacidad para realizar cross validation.

- Stack tecnológico: Colab, Google Drive
- Librerías: xgboost, sklearn.metrics, sklarn.model_selection, pandas

Evaluar el modelo

Evaluación del modelo

- Binomial logistic regression

```
Accuracy: 0.6119663927843331
logLoss: 0.6658319623967739
hammingLoss: 0.3880336072156669
```

```
confusionMatrix:
[[98595.   296.]
 [62515.   464.]]
```

- Gradient-Boosted Trees (GBTs)

```
Accuracy: 0.9128882189038743
logLoss: 0.2995635089596946
hammingLoss: 0.08711178109612569
```

```
confusionMatrix:
[[9.9220e+04 7.9000e+01]
 [1.4021e+04 4.8541e+04]]
```

- XGBoost

  Matriz de confusión de train

	0	1
0	395393	0
1	2851	248636

  Matriz de confusión de test

	0	1
0	98779	0
1	702	62239

```
Train accurancy: 0.9955926910709869
```

```
Test accurancy: 0.9956591639871383
```

Parámetros revisados y ajustados

Los principales parámetros que se debieron ajustar fueron las coordenadas en las que está mayormente conectado un dispositivo en los horarios de casa (0:00 a 7:00) y de oficina (9:00 a 18:00), pasando estos de categóricos a numéricos. Se realizó un proceso similar con la actividad de Home Office que se denotaba como “SI” y “NO”.