TFM

Detección de fraude en empresa de telecomunicaciones

Introducción

El fraude es uno de los campos donde el machine learning está haciendo más avances en los últimos tiempos, empresas financieras ya utilizan este tipo de herramientas para detectar fraudes en transacciones de tarjetas de crédito con resultados altamente satisfactorios, no obstante, el mercado de las telecomunicaciones creo que no ha desarrollado de momento métodos fiables de detección de fraude, extraño ya que supone en torno a un 3% de los ingresos de todas las empresas globalmente.

El campo de las telecomunicaciones, como cualquier campo, tiene sus propias particularidades, algunas de ellas suponen una ventaja y otras una desventaja, la mayor ventaja es la cantidad ingente de datos que generan, generando patrones de uso, localización en tiempo real o datos de cliente, esto hace que sea un campo por explotar, aunque la cantidad de datos que existen y sobre todo la gran cantidad de patrones de consumo hace que la detección de fraude sea compleja, ya que se basa en la detección de anomalías, que cada persona tenga una línea de teléfono (mínimo) y el uso que de esa persona de la línea (por ejemplo si es de trabajo llamará mucho más que si es personal) hace que la detección de anomalías en el consumo sea complicada, y quizá no debería basarse únicamente en el consumo.

Hay muchos tipos de fraude, yo me voy a centrar en el fraude de suscripción, que es dar de alta un servicio (en este caso una línea de telefonía móvil) con fines lucrativos y sin intención de pagar por el servicio, en muchos casos se utilizan datos personales falsos para que el cobro sea más difícil de reclamar por parte de la empresa prestadora de servicios.

Tradicionalmente la detección de fraude en empresas de telecomunicación se basa en reglas, es decir no es predictivo sino que actúa cuando un cliente hace un consumo que según las reglas se considera fraude, algunas de las reglas más típicas son promedio de llamadas por hora, varias llamadas desde un mismo emisor al mismo tiempo o perfiles de cliente en función de su consumo mensual. Yo quiero ir un paso más adelante y detectar anomalías cuando suceden, de esta manera el fraude se detectaría más rápidamente y se salvarían ingresos que de otra manera están condenados a perderse.

Para realizar el proyecto voy a utilizar ficheros de facturación de postpago, previamente tratados para que sean anónimos, el proyecto simularía un proceso de detección de fraude en tiempo real, aunque en este caso no se utilizarían ficheros de facturación sino detalle de consumo. Cada evento de tráfico genera un CDR (call detail record) con información relevante para elaborar un modelo predictivo, como he dicho yo no voy a utilizar CDRs (que si se generarían en tiempo real) sino eventos de consumo facturables, que, si bien son menores en número, si que creo que pueden servir para generar un modelo de clasificación de consumo que podría utilizarse para clasificar consumo como fraudulento.

He explorado diferentes métodos, mi idea inicial era clasificar el consumo de manera no supervisada, pero todos los acercamientos que he hecho en este sentido no han salido como esperaba, aunque creo que si hay posibilidades

de utilizar estos métodos de aprendizaje, creo que se necesita una cantidad ingente de datos que yo no he podido conseguir.

Yo me voy a centrar en uno de los tipos de tráfico que más generan fraude, y por tanto impagos, utilizando un método de aprendizaje supervisado y basado en impagos de clientes.

Descripción del proyecto

Procesado de datos

Como he indicado en la introducción, estoy utilizando ficheros de facturación, concretamente son ficheros para generar las facturas y disponen de varios bloques:

1. El primer bloque es común a todos los ficheros, podríamos llamarlo el encabezado y contiene metadatos globales sobre el fichero, estas son: Parte del fichero, Fecha de creación, Numero de partes que componen el fichero, número de clientes incluidos en el fichero, total de facturas incluidas en el fichero, total de suscripciones incluidas en el fichero. Esta parte la he ignorado en el TFM aunque creo que tiene información valiosa, sobre todo para procesar correctamente el fichero
2. El siguiente bloque es el de los detalles de facturación, y se compone del desglose de la información del cliente (en forma de metadatos), que se divide a su vez en tres partes:
    - Datos del titular: contiene datos personales del cliente, he ignorado esta parte.
    - Cuenta de facturación: contiene la información de facturación del cliente, he ignorado esta parte
    - Datos de factura: contiene datos de la factura, id, importes con y sin IVA, etc. Como me interesaba más el consumo que la factura también he ignorado esta parte
    - Información de suscripción: Esta es la parte que más me interesaba, ya que contiene un encabezado con varios ids internos, el importe total de la factura (que como he comentado antes no me interesa) y sobre todo, los registros de tráfico incluidos en la factura, estos, a su vez tienen sus propias características.

Registros de tráfico:

Se dividen en varios campos, son los siguientes: fecha del evento, usecase del evento (aparece dos veces en dos campos consecutivos), tipo de evento (yo me he centrado en las llamadas de voz salientes), número receptor, número de eventos (para llamadas siempre es 1), volumen (en el caso de las llamadas de voz sería la duración en segundos), base imponible, tipo de base impositiva y precio total (esta información la he omitido también)

En esta parte también aparece la información de los productos contratados, tarifas, bonos adicionales, etc. Al no ser tráfico generado por el cliente también las he ignorado.

El procesado de ficheros se ha realizado sobre el tráfico de 1 año, desde septiembre de 2017 hasta agosto de 2018, han sido 427 ficheros con un total aproximado de 37GB, aunque realmente es menos porque he descartado bastante información que estaba incluida en los ficheros y me he quedado con una pequeña parte de los registros de tráfico, solo he utilizado los registros de llamadas con destinos internacionales.

Lo primero que he hecho es importar las librerías que necesitaba, establecer un path hacia la carpeta de los ficheros y estableces las columnas, de las originales 14 columnas me he quedado con 6.

También he mapeado los eventos de tráfico para que sean legibles y poder filtrar más tarde más fácilmente.

Hay un dato que he creído conveniente procesar, es el del número receptor, al utilizar llamadas internacionales, y para mantener los datos anónimos, lo que he utilizado es únicamente el prefijo, aquí había un problema ya que cada país tiene un prefijo con distinta longitud entre uno y tres dígitos, he tenido que procesar otro fichero para limpiarlo y hacer un set con los prefijos para luego mapearlos en el dataset.

Con estos detalles ya solventados, empezamos a generar el dataset final, he leído los ficheros uno a uno, lo primero que creí necesario era extender un id a una columna, esto ha sido complicado porque los ids no están en la parte de la suscripción, sino que están dentro de los metadatos, por lo que he tenido que utilizar una columna, que afortunadamente estaba repetida, para generar una columna que vaya en línea con los eventos de tráfico, es decir que determina la separación entre los eventos de un cliente y de otro.

También he tenido que sacar de los metadatos el tipo de cliente, al revisar los datos me di cuenta de que había un tipo de cliente que podía ocasionar ruido, este tipo de clientes son los empleados, por ello he realizado un proceso parecido al de los ids para luego filtrarlos.

También he formateado la duración a tipo numérico, y las fechas a datetime, este último campo lo he dividido en dos columnas, una con la fecha y otra con las horas para revisar los patrones de consumo.

Por último, he añadido todo a una lista y la he concatenado en un dataframe, con lo que casi tenemos el set de datos definitivo, con el que he generado un fichero csv.

Revisión y visualización de datos

La primera parte de este bloque quizá se podría considerar parte del procesado de datos pero he creído conveniente hacerlo en esta parte, se trata de hashear los ids de cliente para que no sean públicos, lo he limitado a menos de 8 dígitos para una mayor comodidad.

Una vez anonimizado he generado un nuevo csv.

Lo primero que he hecho es formatear de nuevo las fechas (al venir de un csv perdimos parte del procesado) no es problema porque las he utilizado de diferente manera, he extraído el día de la semana, el mes del evento y el año para ver si hay diferencias, con esto he añadido tres columnas al dataset. También he añadido una columna con las llamadas por minuto en lugar de por segundo, ya que creo que se pueden ver los datos de manera más “natural” que con segundos.

Tras revisar el estado general del fichero, he empezado revisando el número de llamadas por duración, se ve claramente que la gran parte de las llamadas tienen una duración menor de 20 minutos y solo unas pocas tienen una duración mayor de 40. Esto ya nos dice que el patrón “normal” serían llamadas de menos de 20 minutos.

Luego he revisado las llamadas por hora del día, la verdad es que los datos no ofrecen mucha información y parece que siguen un patrón bastante normal, siendo la mayoría entre las 10:00 y las 22:00, se ve un pico descendente desde las 00:00 de la noche.

La duración por tipo de llamada es bastante normal y no ofrece demasiada información, el patrón es bastante parecido a si nos fijamos en la primera gráfica, y la gráfica de densidad es bastante parecida, muchas llamadas de corta duración y muy pocas de más de 2000 segundos.

Al realizar un scatterplot con la hora de llamada y la duración, vemos que hay bastantes registros parecidos, aunque también se ven los outliers claramente, al contrario que en otros acercamientos, estos outliers son muy importantes para nosotros, no olvidemos que en la clasificación de eventos de fraude lo importante es la detección de anomalías, que suelen ser los outliers.

Al añadir el dato de días de la semana al scatterplot, no se ve ningún patrón extraño relacionado con el día de la semana, y al mirar por el tipo de tráfico tampoco, incluso es llamativo que los fines de semana haya menos eventos de este tipo.

Tipo de tráfico por horas o por semanas parece que sigue una distribución bastante lógica, y por año tampoco parece que sea fácil encontrar patrones de fraude.

En resumen, los datos parece que siguen unos patrones bastante lógicos y que no se están produciendo patrones extraños, lo cual es normal, recordemos que en torno a un 3% del tráfico es fraudulento por lo que lo normal es que no se vea claramente estos patrones anómalos.

Modelado

En esta parte he realizado algunos cambios sobre mi idea original, mi idea era realizar un aprendizaje no supervisado, para lo cual he explorado varias maneras, la que me pareció que tenía más sentido era utilizar un random forest, concretamente la versión isolation forest, que utiliza la misma metodología pero detecta anomalías, es un algoritmo muy utilizado en otros tipos de fraude y me pareció que tenía sentido, no obstante no conseguí resultados que me convenciesen.

El modelo debe generar un grado de anomalía entre -0.5 y 0.5, siendo -0.5 tráfico normal y 0.5 tráfico anómalo, pero la realidad es que tras varios intentos el mejor valor que conseguí para el tráfico anómalo fue de 0.12, no me pareció valido por lo cual lo descarté

Pensé que no iba a poder hacer un modelo no supervisado así que intenté explorar el modo supervisado, tenía un problema, mis datos no estaban clasificados, al mirar como clasificarlo vi que no tenía disponibles clientes marcados como fraude, por lo que tampoco podía clasificar eventos de tráfico como fraudulentos.

Tras analizar la situación, vi que la gran mayoría de clientes que realizaban fraude con llamadas de voz internacional acaban impagando la factura, es cierto que esto genera un margen de error, pero si bien no podía identificar tráfico fraudulento, si que podía generar trafico que indujera al impago.

De nuevo cargué la tabla y separé la fecha en año y mes, ya que los impagos se producen mensualmente, hice un merge de ambas tablas por id y generé un fichero con llamadas que inducen a impago y llamadas que no, por lo que ya tenía el tráfico clasificado para realizar un modelo supervisado.

Empecé probando con un acercamiento con Bayes Naive, pero tras varios intentos los resultados no me convencieron, ya que el modelo no conseguía clasificar bien el tráfico, de hecho, muchas veces me daba un 100% de éxito, lo cual no es posible, creo que los datos no ayudaban, ya que hice un onehotencoding sobre el tipo de tráfico y no le debió gustar al modelo.

Finalmente he decidido utilizar un clasificador KNN.

Empecé separando el dataset en pagos e impagos y vi que había un 3% de llamadas que inducen al impago.

Tras esto decidí mirar la duración de las llamadas por pagos y por impagos, vi que las gráficas eran bastante parecidas, y al empezar a modelar pensé que este dato generaba mucho ruido, por eso apliqué un escalado estándar de la duración.

El dato de tipo de tráfico quise usarlo porque creo que es bastante relevante, asi que lo convertí en una columna con valores categóricos con label encoding, cada uno ya tenía un valor numerico con el que podía entrenar el modelo.

Eliminé las columnas que no aportaban nada al modelo, según la visualización de los datos vi que la fecha y la hora no aportaban demasiado, el tipo de cliente era una variable categórica que no tenía sentido para entrenar el algoritmo, y los ids los añadí como índice para no usarlos en el modelo.

Separé el data set en dos (X e Y) y lo separé en entrenamiento y test.

Con estos datos intenté predecir el valor optimo de K, que lo establecí en 3 y ya tenía todo para entrenar el modelo.

Generé la matriz de confusión para ver la precisión, el recall y el f1.

Luego he intentado realizar lo mismo pero agrupando por mes y por cliente, siguiendo la lógica de que el impago lo realiza el cliente, no la llamada, por lo que puede tener más sentido, pero se generó un dataset con dos índices que creo que no al algoritmo no le ha gustado demasiado porque al generar la matriz de confusión me han salido unos datos “extraños”.

Los resultados, como se pueden ver, son muy buenos a la hora de predecir tráfico que no induzca al impago, mientras que al predecir llamadas que induzcan al impago son bastante bajas, en torno a un 60%, creo que el modelo sufre de underfitting, no he sido capaz de encontrar variables que sirvan para crear un modelo robusto.

Creo que este proyecto con más datos y con un algoritmo más potente (siempre pensé que una red neuronal sería muy buena) podría detectar patrones más fácilmente, pero con los datos que tenía, y la gran confusión que hay entre datos con imapgo y sin el, creo que el algoritmo lo ha hecho suficientemente bien, aunque no es un proyecto que sirva para utilizarlo en la vida real.