Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de
Matemàtiques i Informàtica

Universitat de Barcelona

PREDICCIÓ DE CLICS AMB TENSORFLOW

Josep Boldú Quirós

Director: Jordi Vitrià
Realitzat a: Departament de Matemàtiques i Informàtica, UB

Barcelona, 26 de Gener de 2017

Abstract

This project is based on a challenge provided by the website called Kaggle, which involves the creation of an automatic learning system that makes possible a prediction of clicks on online advertising.

Assuming this challenge, a program that is able to predict ad clicks has been created, but by using new technologies like Docker, which makes possible a virtualization of a customized Linux environment ideal for programming Tensorflow, a library that enables an automatic, fast and flexible learning.

The program's main algorithm makes an automatic learning using a linear regression model that feeds from the database provided by Kaggle, which is destined for this challenge. This database is composed of a 10 day history of ad clicks and no clicks.

Resum

Aquest projecte es basa en el repte proporcionat per la pàgina web *Kaggle*, que consisteix en la creació d'un sistema automàtic d'aprenentatge per a fer possible una predicció de clics de publicitat web.

Assumint aquest repte, s'ha creat un programa que és capaç de fer una predicció de clics web, però utilitzant noves tecnologies com *Docker*, que fa possible una virtualització d'un entorn personalitzat *Linux* ideal per la programació de *Tensorflow*, que és una llibreria que fa possible un aprenentatge automàtic, ràpid i flexible.

L'algoritme principal del programa fa un aprenentatge automàtic utilitzant un model de regressió lineal alimentant-se de la base de dades proporcionada pel *Kaggle*, destinada a aquest repte. Aquesta base de dades, es composa d'un històric de 10 dies de clics i no clics web.

Agraïments

M’agradaria donar les gràcies a la meva parella i als meus familiars pel seu suport incondicional.
També m’agradaria agrair al meu tutor del Treball Final de Grau, per tot el seu suport, el seu temps, la seva dedicació i la seva paciència durant la realització d’aquest treball. Sense la seva ajuda i la seva col·laboració, aquest projecte no hauria estat possible.

Índex

1. Introducció

Vivim en una societat clarament capitalista, orientada cada cop més al consum. La manera més ràpida i eficient per a fer arribar un major nombre de consumidors en els productes, és la publicitat web. Diàriament milions de persones naveguen per internet per cercar coses que necessiten o desitgen aconseguir.

Cada cop més, els empresaris comencen a veure internet com un canal a través del qual poden potenciar les seves vendes. Un dels avantatges principals dels enllaços de publicitat en les pagines web, és el seu sistema per mesurar l'eficàcia, sota la denominació de "cost per clic" (CPC). Un anunciant només paga per les visites dirigides a la seva pàgina web des dels cercadors en els que s'hagi donat d'alta. Aquest, controla la campanya en tot moment: decideix el que vol pagar per cada clic i, per tant, en quin lloc apareixerà a la pàgina d'un cercador quan s'introdueixin les paraules clau seleccionades(l'enllaç de la seva web apareixerà més amunt com més disposat estigui a pagar per cada visita).

Per això, és necessari un bon treball d'anàlisi, seguiment i optimització de la campanya que requereix d'un ús intensiu de recursos i un gran domini del medi. Una gestió eficaç d'aquest tipus de publicitat es fonamenta, entre d'altres aspectes, en aconseguir incrementar la visibilitat de la marca, producte o servei i, al mateix moment, optimitzar el retorn de la inversió (ROI[1]) sense tindre pèrdues de beneficis. Per a poder realitzar aquests treballs d'anàlisi, s'adopten estratègies com les implementades en aquest projecte, que consisteixen en fer estudis de predicció de clics i calculant quin producte o publicitat té més probabilitats de ser clicat i analitzar el CTR[2], que és la tassa de clics per mesurar el desenvolupament de la campanya publicitària.

Cada vegada més companyies, implementen en les seves anàlisis de resultats aquestes mètriques, per donar informació més detallada i de gran valor sobre el funcionament de les campanyes publicitàries, mesurant la seva efectivitat i permetent optimitzar el ROI de la mateixa.

2. Objectius i motivació

L'objectiu d'aquest projecte és assolir el repte escollit, proporcionat per la pàgina web Kaggle[3].
Aquesta pàgina web és una plataforma en línia per dur a terme competicions de mineria de dades, que proporciona un espai per a què les companyies publiquin les seves dades i iniciar un concurs obert per a que els experts en mineria de dades de tot el món, descarreguin aquestes dades i proposin solucions als problemes de la companyia en qüestió. La millor solució és recompensada amb un premi que pot arribar a varis milions de dòlars.
En aquest cas, ens proposa que a través d'unes dades facilitades en un fitxer CSV, formada per 10 dies de registres de clics i no clics web, ordenats de forma cronològica, es pugui implementar una predicció de clics web. Per dur a terme aquesta predicció, serà necessària la implementació d'un sistema d'aprenentatge automàtic composat per una regressió lineal, la qual rebrà d'entrada la base de dades, per a poder desenvolupar l'aprenentatge. A l'hora de realitzar la predicció, serà necessària la utilització del programa Docker i la llibreria Tensorflow. Un cop realitzada la predicció, es posarà a prova el sistema creat amb el càlcul del CTR (Click-Through Rate) de les prediccions i l'avaluació de la pàgina web del Kaggle sobre aquest repte, comparant els resultats obtinguts amb la resta de participants, obtenint així la seva eficàcia i eficiència.

Els camps de les dades facilitades per la pàgina web Kaggle, es composen de:
-id: identificador
-click: 0 si no ha sigut clicat i 1 si ha sigut clicat
-hour: format en YYMMDDHH
-C1 – variable categòrica anònima
-banner_pos
-site_id
-site_domain
-site_category
-app_id
-app_domain
-app_category
-device_id
-device_ip
-device_model
-device_type
-device_conn_type

-C14-C21 -- variables categòriques anònimes

Els tres primers camps són els més importants i la resta de camps formarien part de les característiques del registre.
La base de dades la qual es farà servir per testejar el programa, està creada per un terç de la base de dades original. En el procés de test, la utilització del clic serà merament informativa per comparar el resultat real amb el resultat de la predicció.
Donada la política de propietat intel·lectual de les dades, no es podrà mostrar el contingut de les mateixes ni fer-ne difusió.

Els sub-objectius d'aquest projecte són:

1. Instal·lar les noves tecnologies per a poder iniciar el projecte: Realitzar la instal·lació i configuració del Docker, el qual executant el paquet de software escollit de Tensorflow, permetrà crear un entorn de desenvolupament on començar a desenvolupar el projecte

2. Aprendre el funcionament de les noves tecnologies i llibreries: Aprendre el funcionament i l'execució de Docker. Per altra banda, conèixer i entendre el funcionament de Tensorflow i posteriorment, la seva codificació.

3. Programar una solució al repte adoptat: Utilitzant les noves tecnologies esmentades anteriorment, més les tècniques d'aprenentatge automàtic i el model de regressió lineal, es programarà una solució que sigui capaç de realitzar la predicció de clics web.

4. Analitzar els resultats obtinguts: Recopilar els resultats de l'execució del programa. comprovar la seva eficàcia i fiabilitat i realitzar l'avaluació del Kaggle.

La motivació en aquest projecte neix de la curiositat que sempre he tingut en el Machine Learning i en el processament de grans quantitats de dades. D'aquí hem desperta molt interès la gran varietat de resultats que es poden obtenir analitzant les dades i la varietat d'estratègies i mètodes que es poden aplicar. A través del meu tutor del projecte de Final de Grau, es va plantejar el repte de la pàgina web del Kaggle[1], però utilitzant noves tecnologies, dels quals vaig trobar molt interessant.
A més a més, amb l'evolució constant de la tecnologia, sorgeix la necessitat de mantenir-se al dia en l'aprenentatge de noves tecnologies i aquest repte hem dóna l'oportunitat de fer-ho.
En el meu transcurs laboral, m'he adonat de la conseqüència que suposa treballar amb una gran magnitud de dades utilitzant tecnologies obsoletes, fent-me veure la necessitat de profunditzar en la investigació d'aquests mètodes de processament de dades.

3. Metodologia

Els mètodes més destacats què s'han utilitzat per al desenvolupament d'aquest projecte són els següents:

- Docker(Windows10 Professional 64 bits)
- Tensorflow(DatascienceUB/deepUB)
- Algoritme de Regressió Lineal

2.1 Docker

Docker[4] és un projecte de codi obert que automatitza el desplegament d'altres aplicacions dins de contenidors de software, proporcionant una capa addicional d'abstracció i automatització de virtualització a nivell de sistema operatiu Linux.

Amb la tecnologia Docker es pot realitzar la virtualització d'un sistema operatiu Linux amb totes les aplicacions necessàries deixant fora les aplicacions innecessàries o irrellevants per a després empaquetar-ho i convertir-ho en un contenidor de software. Fent això, es pot crear un entorn de desenvolupament de manera que es pot executar fàcilment mitjançant una línia de comandaments.
Com que Tensorflow només funciona, de moment, en sistemes operatius Linux i Mac (a part d'Android i iOS), ha sigut necessària la utilització de Docker per dur a terme aquest projecte, ja que el sistema operatiu base utilitzat, ha sigut Windows 10 Professional de 64 bits.

A més a més, la diferència entre Docker i les màquines virtuals convencionals és que, mentre que cada màquina virtual necessita comptar amb un sistema operatiu complert (ocupant per exemple 1GB de memòria RAM), Docker aprofita el Kernel de la màquina i, carrega en memòria les llibreries i dependències necessàries per a l'execució de les aplicacions, despreciant tota la resta de components del sistema operatiu, ocupant aproximadament un 70% menys de memòria RAM. En la següent il·lustració es mostra visualment la comparativa:

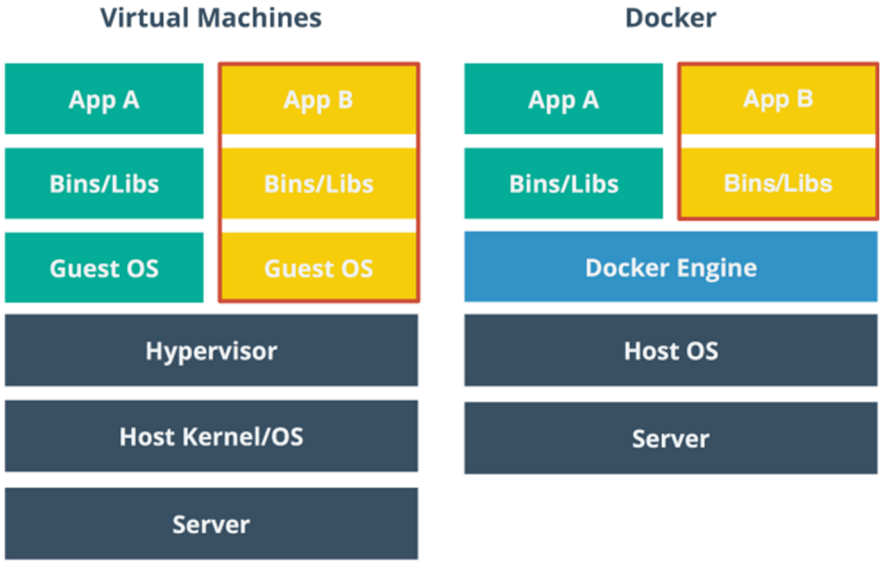

Figura 1. Comparativa entre màquina virtual i Docker

Docker accedeix a la virtualització del Kernel de Linux, ja sigui directament a través de la biblioteca "libcontainer" o indirectament a través de "libvirt", "LXC" o "systemd-nspawn" fent així possible la virtualització d'un entorn Linux:

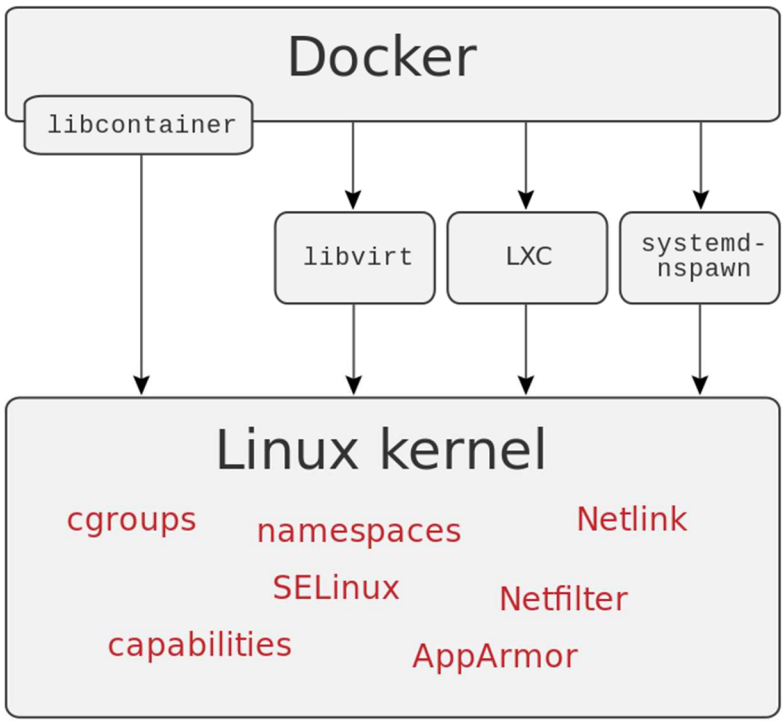

Figura 2. Diagrama de comunicació entre Docker i Kernel Linux

Les característiques principals dels contenidors són:

- Portabilitat: Els contenidors Docker es poden desplegar en qualsevol sistema operatiu, tenint així sempre l'entorn personalitzat.

- Lleuger: El pes dels contenidors no tenen comparació en el pes de qualsevol sistema de virtualització convencional.

- Autosuficiència: Un contenidor no conté tot un sistema operatiu, sinó únicament aquelles aplicacions, llibreries, configuració i arxius necessaris per a desplegar les funcionalitats de l'entorn i fer-lo funcionar.

Per la realització d'aquest projecte, s'ha utilitzat un contenidor de software ja creat especialment per la programació en Tensorflow (DatascienceUB/deepUB), que apart d'instal·lar tot el necessari de Tensorflow, instal·la aplicacions extra molt útils pel tractament i anàlisi de dades. A continuació es mostra detalladament totes les instruccions que Docker executa del contenidor de software la qual fa possible la virtualització, instal·lació i configuració de tot el sistema Linux:

- La instal·lació bàsica de Tensorflow i tots els components:

```
FROM ubuntu:14.04

MAINTAINER Craig Citro <craigcitro@google.com>

# Pick up some TF dependencies
RUN apt-get update && apt-get install -y --no-install-recommends \
        build-essential \
        curl \
        libfreetype6-dev \
        libpng12-dev \
        libzmq3-dev \
        pkg-config \
        python \
        python-dev \
        rsync \
        software-properties-common \
        unzip \
        && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

RUN curl -O https://bootstrap.pypa.io/get-pip.py && \
```

```
    python get-pip.py && \
    rm get-pip.py

RUN pip --no-cache-dir install \
        ipykernel \
        jupyter \
        matplotlib \
        numpy \
        scipy \
        sklearn \
        Pillow \
        && \
    python -m ipykernel.kernelspec

# --- DO NOT EDIT OR DELETE BETWEEN THE LINES --- #
# These lines will be edited automatically by parameterized_docker_build.sh. #
# COPY _PIP_FILE_ /
# RUN pip --no-cache-dir install /_PIP_FILE_
# RUN rm -f /_PIP_FILE_

# Install TensorFlow CPU version from central repo
RUN pip --no-cache-dir install \
    http://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-0.0.0-cp27-none-linux_x86_64.whl
# --- ~ DO NOT EDIT OR DELETE BETWEEN THE LINES --- #

# RUN ln -s /usr/bin/python3 /usr/bin/python#

# Set up our notebook config.
COPY jupyter_notebook_config.py /root/.jupyter/

# Copy sample notebooks.
COPY notebooks /notebooks

# Jupyter has issues with being run directly:
#   https://github.com/ipython/ipython/issues/7062
# We just add a little wrapper script.
COPY run_jupyter.sh /

# TensorBoard
EXPOSE 6006
# IPython
EXPOSE 8888

WORKDIR "/notebooks"

CMD ["/run_jupyter.sh"]
```

-La instal·lació de les aplicacions extra:

```
FROM gcr.io/tensorflow/tensorflow
  #Install packages
  RUN DEBIAN_FRONTEND=noninteractive apt-get update
  RUN DEBIAN_FRONTEND=noninteractive apt-get -qqy install wget python-pip git timidity unzip
  RUN DEBIAN_FRONTEND=noninteractive pip install --upgrade pip
  RUN DEBIAN_FRONTEND=noninteractive pip install tqdm pandas seaborn bokeh sklearn keras
h5py scikit-image
  RUN DEBIAN_FRONTEND=noninteractive pip install git+https://github.com/tflearn/tflearn.git

  RUN pip install --upgrade numpy
  RUN pip install --upgrade scikit-image

  #Remove examples
  RUN rm -Rf *
```

2.2 Tensorflow

L'aprenentatge automàtic ha sigut matèria d'investigació acadèmica durant dècades, però des de fa pocs anys es pot observar com incrementa la seva presència dins de les industries i corporacions, donada la necessitat de processar i analitzar la gran quantitat de dades de les que disposen.

En aquest camp, no hi ha dubte que Google és una de les grans corporacions on el Machine Learning està adquirint un gran protagonisme fonamental en les seves iniciatives i productes.
A arrel d'aquesta necessitat, Google va oferir fa poc, en codi obert, el Tensorflow.

Tensorflow[8] és una llibreria de programari de codi obert per al càlcul numèric utilitzant diagrames de flux de dades o grafs. Els nodes del graf representen operacions matemàtiques, mentre que les vores del graf representen els arranjaments de dades multidimensionals (tensors) comunicats entre ells.

Per altra banda, és el sistema de Machine Learning de Google de segona generació preparat per a xarxes neuronals, Intel·ligència Artificial i Deep Learning.

Va ser originalment desenvolupat per investigadors i enginyers que treballen en l'equip de Google Brain dins de l'organització de Machine Intelligence de Google a l'efecte de dur a terme el Machine Learning i profunditzar la investigació de xarxes neuronals. El sistema també és prou general per ser aplicable en una àmplia varietat d'altres dominis. Actualment, Google fa servir el Tensorflow en Gmail, Google Photos, recerques, reconeixements de veu, etc.

Les característiques més importants d'aquesta llibreria són:

- Flexibilitat: Proporciona eines per muntar gràfics per expressar diversos models d'aprenentatge automàtic. Les noves operacions poden ser escrits en Python i els operadors de dades de baix nivell, s'implementen utilitzant C ++.
- Portabilitat: S'executa en plataformes de CPU, GPU, d'escriptori, servidors o de computació mòbil, a més a més, fa que sigui molt adequat en diversos camps d'aplicació, per exemple, mèdica, finances, electrònica de consum, etc.
- Connexió entre investigació i producció: Tensorflow permet als investigadors una major rapidesa en la creació de prototips de productes. També proporciona als investigadors acadèmics, un marc de desenvolupament i una comunitat per discutir i donar suport a noves aplicacions.
- Auto diferenciació: Aquesta és una característica clau dins de la comunitat d'aprenentatge automàtic. Els algoritmes d'aprenentatge automàtic basats en gradients, es beneficien de les capacitats de diferenciació automàtica. Defineix l'arquitectura computacional per al seu model predictiu, el combina amb la seva funció objectiu i només ha d'afegir les dades per provar el seu model d'aprenentatge automàtic.
- Maximització del rendiment: Li permet treure el màxim profit del seu maquinari instal·lat.

Les funcions de Tensorflow són:

- Representar els càlculs en forma de grafs
- Executa els grafs en el context de sessions.
- Representa les dades com a tensors[5]
- Manté l'estat amb les variables
- S'alimenta de les dades i retorna els resultats en cada operació

Aquesta arquitectura flexible li permet implementar el càlcul d'una o més CPU o GPU en un ordinador d'escriptori, servidor o dispositiu mòbil sense reescriure el codi.

El nucli de Tensorflow està escrit en C++, que ofereix llibreries en Python i C++ i pot funcionar sobre CPU, GPU, Linux i Mac a més d'Android i iOS, tal i com es visualitza en el següent diagrama:

Figura 3. Diagrama de l'estructura de Tensorflow

Com moltes altres llibreries, Tensorflow, requereix característiques d'altres sistemes, no només per mantenir la llibreria, sinó per habilitar noves funcionalitats. Les principals dependències són el suport del controlador de NVIDIA CUDA i el suport dels llenguatges Python i C++. Per altra banda, hi han altres dependències que envolten la llibreria de Tensorflow, les quals es mostren en la següent il·lustració:

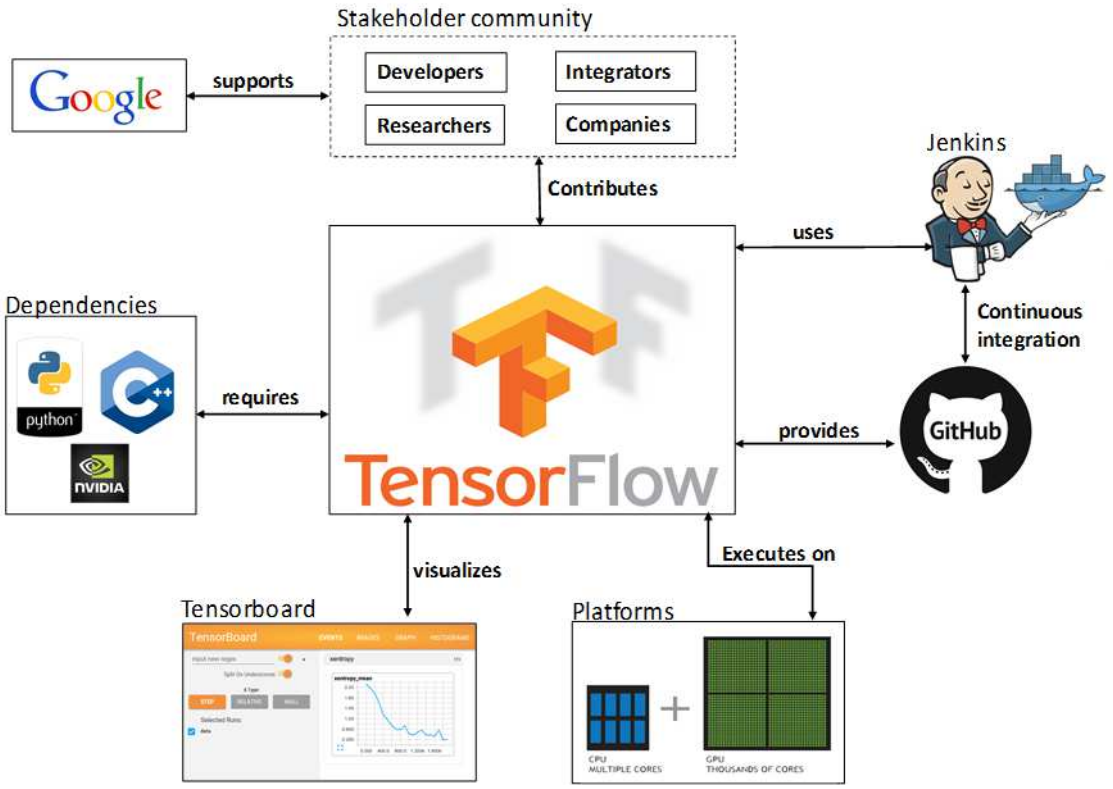

Figura 4. Diagrama de components que forma Tensorflow

2.3 Algoritme de Regressió Lineal

Analitzant diferents models de predicció, s'ha arribat a la conclusió que per a fer la predicció de clics amb Tensorflow era necessari realitzar una Regressió Lineal.

La regressió lineal és una tècnica estadística utilitzada per a mesurar la relació entre variables. El seu interès radica que l'algoritme que ho implementa no és complex conceptualment, i a més a més, s'adapta a una àmplia varietat de situacions. Serà de gran utilitat en aquest projecte, ja que es busca la relació entre les dades d'aprenentatge i les dades obtingudes en la predicció.

En aquest cas, s'ha implementat una Regressió Lineal composada per la fórmula:

$$p = (W * x) + b$$

On els seus components signifiquen:
-p: el valor calculat/predit.
-W: és la variable que determina la inclinació de la recta, és a dir, l'increment que es produeix en *p* quan la variable *W* augmenta una unitat.
-x: representa les dades d'entrada del model, que són una llista de característiques dels registres de la base de dades.
-b: és la ordenada a l'origen, és a dir, l'altura a la que la recta talla l'eix *y*. Es denomina també terme independent.

4. Desenvolupament

En el desenvolupament d'aquest projecte està format per una sèrie d'etapes:

4.1 Instal·lació

Donat que l'equip utilitzat per dur a terme aquest projecte funciona sobre un Windows 10 Professional, únicament ha sigut necessària la descàrrega e instal·lació de Docker en la seva versió més actual. En versions anteriors del sistema operatiu Windows seria necessària la instal·lació del Docker Toolbox.

Com s'ha esmentat anteriorment, s'ha fet servir un contenidor de software existent el qual ja té creat tot l'entorn adequat per la programació en Tensorflow.

Un dels inconvenients que ha sorgit, ha estat que les dades fora del propi contenidor de software no són persistents en la virtualització. Cada cop que s'executa el contenidor de software es crea una imatge nova del sistema operatiu Linux amb les característiques i aplicacions definides en el contenidor. Això comporta que tot el progrés de codificació es perd un cop finalitza la virtualització.

La solució a aquest problema és compartir el disc dur local de l'equip amb la virtualització mitjançant les propietats de Docker fent així que la virtualització visualitzi una carpeta en concret. Aquesta, resideix en el disc dur local al marge de la virtualització fent que les dades emmagatzemades allí, persisteixin un cop finalitzada la virtualització. En aquest cas, s'ha compartit la unitat C de l'equip.

Amb l'ajut de l'aplicació Jupyter, la qual forma part del contenidor de software, es pot establir una comunicació web mitjançant el port 8888 amb el navegador predeterminat que es tingui seleccionat per a poder crear o importar un notebook, el qual s'utilitzarà per a la codificació.

Tot això, es realitza mitjançant l'execució del comandament en la consola del sistema operatiu Windows amb privilegis d'Administrador:

```
docker run -it -p 8888:8888 -v C:\Users\Josep\Desktop\TFG:/notebooks datascienceub/deepub
```

En el comandament s'indica el port pel qual es podrà accedir a l'entorn de virtualització, la ruta compartida entre la virtualització i la màquina i el paquet de software el qual es vol executar en el Docker.

Un cop executat el comandament, cal obrir un navegador web com per exemple, el Google Chrome i anar a l'adreça http://localhost:8888. Fent això, es connectarà directament amb l'aplicació Jupiter dins de l'entorn de programació.

4.2 Aprenentatge de la llibreria:

Com que la llibreria Tensorflow és relativament nova i mai havia tingut l'ocasió de fer-la servir, ha sigut necessari passar per un procés d'aprenentatge de Tensorflow.

Voldria fer especial menció a la pàgina web learningtensorflow[6] i el llibre "Hello World en Tensorflow" de Jordi Torres[7], la qual ha sigut de gran utilitat a l'hora d'aprendre el funcionament de Tensorflow.

Amb l'ajut dels recursos esmentats i executant tots els exemples, es va assolir els coneixements necessaris per a fer possible la realització d'aquest projecte.

4.3 Codificació:

En aquest apartat es comentarà tota la codificació feta pas per pas(Annex1):

El primer pas, és fer les importacions necessàries per a l'execució del codi. S'importa la llibreria de *Tensorflow*, el paquet *numpy* per a la gestió de llistes, el paquet *matplotlib.pyplot* per a poder mostrar gràfiques dels resultats obtinguts en l'execució del codi, el paquet *csv* per la lectura i escriptura de fitxers en format csv i la llibreria *math* per a la realització de càlculs matemàtics.

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from math import exp, log, sqrt
from csv import DictReader
```

S'ha implementat el mètode data(path,D), el qual rep la ruta on s'ubica la base de dades i *D* que és el nombre màxim d'índex en l'arbre Hash i les dimensions de les variables del Model Lineal.
El mètode es basa en l'extracció de l'ID del registre, guardar-se si hi ha un clic o no i en construir la variable *x*. La variable *x* representa les dades d'entrada del model que és una llista que té un índex construït per l'algoritme Hash de dimensió *D* contenint les característiques de cada registre de les dades. Aquesta funció, mitjançant el yield, retorna en cada iteració del programa principal les dades extretes en la iteració interna del bucle del mètode.

```
def data(path, D):
    ''' GENERATOR: Apply hash-trick to the original csv row
                   and for simplicity, we one-hot-encode everything
        INPUT:
            path: path to training or testing file
            D: the max index that we can hash to

        YIELDS:
            ID: id of the instance, mainly useless
            x: a list of hashed and one-hot-encoded 'indices'
               we only need the index since all values are either 0 or 1
            y: y = 1 if we have a click, else we have y = 0
    '''
    for t, row in enumerate(DictReader(open(path))):
        # process ID
        ID = row['id']
        del row['id']

        # process clicks
        y = 0.
        if 'click' in row:
            if row['click'] == '1':
                y = 1.
            del row['click']

        # build x
        x = np.zeros(D)
        for key in row:
            value = row[key]

        # one-hot encode everything with hash trick
            index = abs(hash(key + '_' + value)) % D
            x[index]+=1

        yield t, ID, np.expand_dims(x,axis=0), np.expand_dims(y,axis=0)
```

S'ha implementat el mètode logloss(p,y) el qual té la funció de calcular la puntuació final del projecte segons la normativa de la competició del Kaggle. Rep com a entrada la predicció *p* i el clic *y*.

```
def logloss(p, y):
    '" FUNCTION: Bounded logloss

        INPUT:
            p: our prediction
            y: real answer

        OUTPUT:
            logarithmic loss of p given y
    '"
    p = max(min(p, 1. - 10e-15), 10e-15)
    return -log(p) if y == 1. else -log(1. - p)
```

Es defineixen les rutes estàtiques les quals estan les bases de dades d'aprenentatge i test, com també l'arxiu amb els resultats de les prediccions.

```
# Paths
train = 'train.csv'                     # path to training file
test = 'test.csv'                       # path to testing file
submission = 'submissionFINAL.csv'   # path of to be outputted submission file
```

Es defineix el rati d'aprenentatge de l'algoritme d'aprenentatge.

```
learning_rate = tf.Variable(0.001, name='learning_rate')     # learning rate
```

En *D* s'estableix la dimensió de l'arbre Hash, el qual composa els índex de *x* i les dimensions del model lineal. També es defineix el nombre d'èpoques d'aprenentatge que farà l'algoritme per assegurar-se un correcte aprenentatge i amb conseqüent obtenir una millor predicció.

```
D = 5000     # number of weights to use
epoch = 500   # learn training data for N passes
```

Es defineix la llista *cost_history* la qual obtindrà un històric de la mitjana de la funció de cada època en l'execució de l'algoritme.

La variable *aux* acumularà el resultat de la funció de cost obtinguda en cada època.

Es defineix e inicialitza un comptador el qual es farà servir per a obtenir la mitjana dels resultats de la funció de cost, en cada iteració fent possible veure el progrés de l'aprenentatge.

```
cost_history = []
aux=0.
cont = 1.
```

Es defineixen els tensors que seran utilitzats en tota la regressió lineal i en l'algoritme de càlcul del cost d'aprenentatge.

Les variables resideixen en el graf d'operacions que construeix Tensorflow internament.

Els placeholders estan destinats a poder ser manipulats durant l'execució del programa.

```
#Placeholders
X = tf.placeholder(tf.float32,[1, D])
Y = tf.placeholder(tf.float32,[1])

#Variables
W = tf.Variable(tf.random_normal(shape=[D,1],stddev=0.01), name="weight")
b = tf.Variable(tf.zeros([1]), name="bias")
```

En *p* es defineix el model lineal esmentat en apartats anteriors representant la fórmula:

$$p = (W * x) + b$$

```
#Prediction Model
p = tf.matmul(X, W) + b
```

Es defineix la funció de cost coneguda com "mean squad error"[9]. Amb aquesta funció de cost s'obté la mitjana dels errors en base a la distància entre el valor real i el valor estimat de cada una de les iteracions. Consisteix en fer el quadrat i seguidament la mitja, per obtenir la mitjana del cost entre els punts.

```
#Cost Function
cost = tf.reduce_mean(tf.square(p-Y))
```

En aquests moments ja es pot intuir que com més petit sigui el valor de la funció de cost, més precisa serà la predicció. Per tant, si es minimitza la funció de cost, es pot aconseguir una millor predicció en el model.

Per això, cal executar l'algoritme *FtrlOptimizer* de la llibreria de Tensorflow, el qual s'encarrega de minimitzar la funció de cost indicant-li la funció i afegint-li el rati d'aprenentatge, completant tot el model d'aprenentatge.

```
#Learning function
learner = tf.train.FtrlOptimizer(learning_rate).minimize(cost)
```

Fins aquest punt, únicament s'han definit mètodes i trucades a la llibreria de Tensorflow que van afegint informació al graf intern. El pas següent és executar l'algoritme, però abans, cal inicialitzar prèviament totes les variables amb la següent instrucció.

```
#Initialization of variables
init = tf.initialize_all_variables()
```

Per a l'execució de l'algoritme és necessari crear una sessió i "fer-la córrer".

```
#Create Session
sess = tf.Session()
sess.run(init)
```

S'inicia la iteració de les dades i de les èpoques estipulades per l'aprenentatge.

S'itera la *t* que representa el número d'iteració dins d'una època, l'ID del registre, la *y* representant el clic o no clic en el registre i la *x* amb la llista de les característiques del registre.

```
#Start Training
for e in range(epoch):
    for (t, ID, x, y) in data(train, D):
```

S'executa l'algoritme d'aprenentatge i es calcula el cost en cada iteració indicant quines són les variables que corresponen en el graf.

```
        sess.run(learner, feed_dict={X: x, Y: y})
        training_cost = sess.run(cost, feed_dict={X: x, Y: y})
```

S'acumula el resultat de la funció de cost en cada iteració

```
        aux+=training_cost
```

S'incrementa el comptador necessari per fer el càlcul de la mitjana del cost.

```
        cont +=1
```

Fetes totes les iteracions i en concloure una època d'aprenentatge, es mostra per pantalla la mitjana del cost total de tota l'època i es guarda en un històric de costos, el qual es farà servir més endavant per mostrar la corba d'aprenentatge.

```
    #Total Training Cost Epoch
    print("Epoch: ",e)
    print("Training Cost Epoch = ", aux/cont)
    cost_history.append(aux/cont)
```

Finalitzades totes les èpoques, es mostra la mitjana del cost total de tot l'aprenentatge.

```
#Total Training Cost
print("Total Training = ", aux/cont)
```

Per a l'obtenció d'un resultat més visual, es mostra la gràfica de la corba d'aprenentatge obtinguda mitjançant l'històric del cost d'aprenentatge, que s'ha anat acumulant durant tot el procés.

```
#Graph of the learning curve
plt.plot(cost_history)
plt.xlabel("Epochs")
plt.ylabel("Cost")
plt.show()
```

Un cop realitzat tot l'aprenentatge, el següent pas és calcular les prediccions i posar a prova tot l'aprenentatge realitzat. S'escriu en un fitxer l'ID del registre, el clic real (*y*) i el resultat del càlcul de la funció de predicció. En l'apartat de resultats es mostrarà el contingut d'aquest fitxer.
A més a més, es realitza el càlcul del CTR mencionat anteriorment per a comprovar la fiabilitat de tot el sistema de predicció i es realitzarà l'avaluació del projecte calculant la puntuació final. S'inicialitzen les variables nomenades *acum_ctr* i *count_eval i acum_logloss* que serviran per a calcular el CTR i la puntuació final.

```
#Writing the prediction
acum_ctr=0.
acum_logloss=0.
count_eval=0.
with open(submission, 'w') as outfile:
   outfile.write('id,real click,click predicted\n')
   for (t, ID, x, y) in data(test, D):
      prediction = sess.run(p[0],feed_dict={X: x,Y: y})
      outfile.write('%s,%s,%s\n' % (ID, y, prediction))
      if float(prediction) >= 0.5:
         acum_ctr+=1
      acum_logloss +=logloss(prediction,y)
      count_eval+=1

#Calculate CTR
ctr = (acum_ctr/count_eval)*100
print("CTR = ",ctr)

#Final Score
print("Final Score(Logarithmic Loss)= ", acum_logloss/count_eval)

print("END OF PROGRAM!")
```

5. Resultats

En aquest apartat s'analitza les impressions per pantalla, del programa i els resultats obtinguts en les diferents proves realitzades durant i després de l'execució del programa.

5.1 Anàlisi del cost d'aprenentatge

Durant l'execució del programa, s'ha configurat una impressió per pantalla de la mitjana del resultat obtingut de la funció de cost a cada finalització d'una època, en les quals es pot apreciar l'evolució del cost d'aprenentatge. Es pot visualitzar com el cost es va reduint progressivament en el transcurs de les èpoques i cada cop s'aproxima més a 0. Això significa, que cada cop la predicció està més a prop de la realitat i que l'aprenentatge es va realitzant correctament.
Com més èpoques s'executen, més fiable és l'aprenentatge, més precisa és la predicció i menor és el resultat de la funció de cost.

En aquesta visualització, es mostra la corba d'aprenentatge obtinguda en l'execució de 158 èpoques. Es pot apreciar com en el transcurs de les èpoques, el cost es va reduint significativament.
En l'annex 2 es poden visualitzar les dades desglossades obtingudes per la realització de la gràfica mostrada.

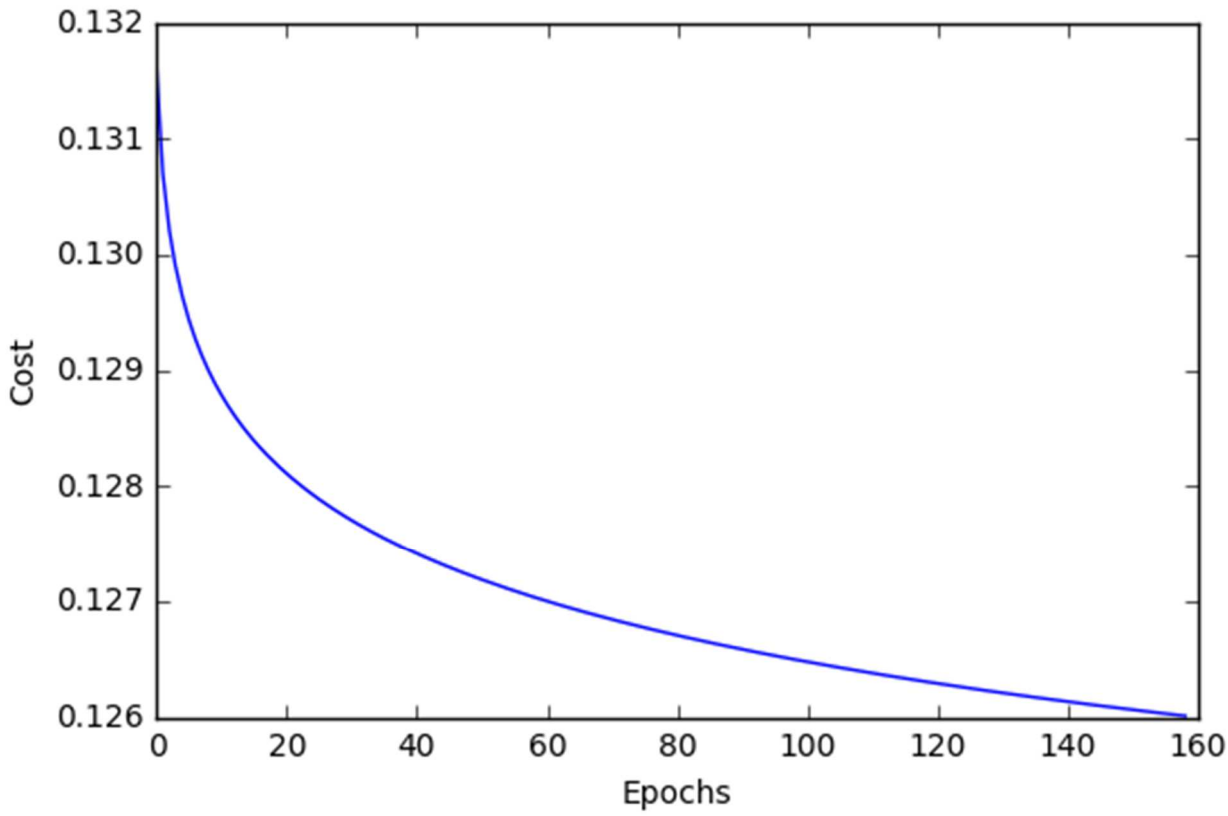

Figura 5. Gràfica de la corba d'aprenentatge

5.2 Anàlisi de prediccions

En aquest apartar s'analitzaran les prediccions fetes:

5.2.1 Desglossament de prediccions

En la següent taula, es mostra una part de l'arxiu que conté el resultats de les prediccions realitzades. Es pot apreciar com en gran part de registres amb 1 en el clic real, la predicció obtinguda és molt pròxima a 1 respecte les prediccions amb clic real de 0. Per aquest motiu, no significa que sempre es compleixi aquesta norma. Pot ser que un registre amb un clic real, no sigui un bon candidat a ser clicat novament i tingui un resultat de predicció més pròxim a 0. També pot succeir que un registre que no ha sigut clicat, tingui un resultat de predicció més pròxim a 1 respecte a la resta de prediccions, ja que pot ser que hi hagin registres molt similars al registre en qüestió, els quals sí han estat clicats. Cal recordar que són els resultats de l'execució de 158 èpoques i que pot ser, que es visualitzin resultats no concordants amb l'esperat. Com s'ha mencionat anteriorment, com més èpoques transcorren, millor són els resultats de la predicció.

Per a fer-ho més visual, és mostraran els requadres en colors dels següents casos:

- Verd: Els registres amb clic real d'1 amb una predicció pròxima a 1.
- Vermell: Els registres amb clic real d'1 amb una predicció pròxima a 0.
- Blau: Els registres amb clic real de 0 amb una predicció pròxima a 1.
- Blanc: La resta de registres.

id	Real Click	Click Predicted
1000009418151090000	0	0.17092077
10000169349117800000	0	0.25009465
10000371904215100000	0	0.16588736
10000640724480800000	0	0.24579562
10285796716761600000	1	0.50425267
10000720757801100000	0	0.16915292
10000724729988500000	0	0.22954045
10000918755742300000	0	0.3986038
10000949271186000000	1	0.25093475
10001264480619400000	0	0.17920619
10658953485482200000	1	0.50877011
10001966791793500000	0	0.2768037
10002028568167300000	0	-0.0806751
10002044883120800000	0	0.22671849
10002518649031400000	0	0.16795065
10003539039235300000	0	0.13402119
10003585669470200000	0	0.27067339

id	Real Click	Click Predicted
10018330170475700000	0	0.09398821
10018368562703100000	0	0.08279778
10018479281174600000	0	0.23307177
10018563981679900000	0	0.08223578
10018756062798000000	1	0.26652956
10018897422996600000	0	0.27299264
10018930650935200000	0	0.24069987
10019071520499500000	0	0.17158888
10440339404859100000	1	0.59408599
10019174283089500000	0	-0.03059
10019341288757400000	0	0.09728107
10019351860081900000	0	0.05886426
10019396062291300000	0	0.30530381
10019540781113300000	0	0.13270088
10019805607671100000	0	0.3571386
1002011100077870000	0	-0.0141116
10192397080531600000	1	0.54506844

10004105575081200000	0	0.25414261
10004181428767700000	0	0.15673663
10004482643316000000	0	-0.0038410
10004510652136400000	0	0.05680829
10004574413841500000	0	0.13825323
10004670021948900000	0	0.02453506
10471194032455000000	1	0.6987347
10005249248600800000	0	0.17377363
10005334911727400000	0	0.09289992
10787247478616200000	0	0.58978444
10005609489911200000	1	0.04907049
10005649443863200000	0	0.08620876
10005951398749600000	0	0.15647346
10006192453619700000	0	0.18124512
10006415976094800000	0	0.35873482
10006490708516100000	1	0.23189564
10006557235872300000	0	0.23599666
10006629065800200000	0	0.07134202
10006777279679600000	0	-0.1145679
10006789981076400000	0	0.03281388
10006958186789000000	1	0.15370433
10007163879183300000	0	0.04927771
10007164336863900000	1	0.17108269
10007197383452500000	0	0.21528281
10007446479189600000	0	0.10147482
10007768440836600000	0	0.12601909
10007830732992700000	0	0.00494302
10007847530896900000	1	0.36008769
10007908698866400000	0	0.10946618
10007944429976900000	1	0.2685132
10009147085943300000	0	0.31677556
10009190848778700000	0	0.34619096
10009635774586300000	0	0.03883105
10009699694430400000	1	0.12278208
10009807995169300000	0	0.22092225
10009910814812200000	1	0.084153
10010452321736300000	1	0.26347685
10010485868773700000	0	0.17677034
10010504760200400000	0	0.34895787
10010730108771300000	0	0.3854804
10010804179216200000	0	0.21608217
10010827185580900000	0	0.06323984
10010924186026100000	0	0.08857384
10010966574628100000	1	0.14343004
10011085150831300000	0	0.25597072
10011205200760000000	0	0.12667479
10011395950642400000	0	0.23557459
10011406079394700000	0	0.03654192
10011560478081700000	1	0.16826399
10011561503992800000	0	0.25193715

10020372821928700000	0	0.18040207
10020492815188800000	1	0.25486147
10020605367282200000	0	0.28286332
10020664336520300000	0	-0.0408897
108131030651209000	1	0.05835745
10020838610941300000	0	0.14268711
10021516630479100000	0	0.12497447
10021697715830900000	1	0.25359455
10021850042230300000	0	0.18475251
10021869890671400000	0	0.17218408
10022511657747300000	0	0.17492299
10022570284312900000	0	0.07056625
10022750613173700000	0	0.56557912
10022778968214200000	0	0.1627854
10022915835050300000	0	0.03446487
10022961149355200000	0	0.16035478
10023110164290000000	0	0.22006571
10023200351270300000	0	0.05236552
10023308754907100000	1	0.25279513
10023681101642900000	0	0.14634679
10023836977405100000	0	0.15327294
10024331030544300000	1	0.15690006
10024365483589100000	0	0.12480822
10024438216634600000	0	0.2336774
10024590463713200000	0	0.19714071
10024660782388100000	1	0.55604309
10024728239031000000	0	0.09600794
1002474033882040000	1	0.53510785
10024810769855500000	1	0.33214864
10024869997513100000	1	0.23209496
10024931014706400000	0	0.05317042
10025131253983100000	0	0.1780418
10025422497693300000	0	0.10776304
10025480338481600000	0	0.26403344
10025633842336100000	0	0.03792152
10025704690568200000	0	0.24609552
10027325869092400000	0	0.16593774
10027449057616700000	0	0.20861882
1070199585069640000	1	0.56281292
10027581429278900000	0	0.1401844
10027643527627900000	0	0.07739609
10028118836233400000	0	0.01619832
10028158991193600000	0	0.18992485
10028335185239000000	1	0.58432573
10028586077501400000	0	0.24563842
10028659945951000000	0	0.1216066
10028663841911500000	1	0.47416779
10028762069414900000	0	0.04143056
10028808005964400000	0	0.02674666
10028926381063300000	0	0.09419501

10011650513707900000	0	0.04013048
10011658782619000000	1	0.22043677
10601716052055600000	1	0.49628243
10011792892936000000	0	0.24148192
10012212068904300000	0	0.06107024
10012222478217600000	0	0.25513792
10012820175855400000	0	0.21200338
10013076841337900000	0	0.18867098
10013222055782900000	0	0.48289216
10013330254346400000	0	0.10111199
10013378798301800000	1	0.29186961
10013493678511700000	0	0.11213651
10013552540914000000	0	0.16451989
10013750748974100000	0	0.1991768
10013786915988000000	0	0.27435654
10013840276980900000	0	0.22459109
10013846047025200000	0	0.24042794
10014026899633500000	0	0.21693544
10014063680973100000	0	0.12435183
10014190212266300000	1	0.3360309
10014285064795200000	1	0.19105645
10942958331190500000	1	0.07604986
10014630626523000000	0	0.17791864
10014764617325700000	0	0.23261942
10014885175555300000	0	0.27192137
10014887683839700000	1	0.22624378
10015140740686500000	0	0.05120764
10015211672544600000	0	0.29845732
10015376300289300000	0	0.23976935
10268393552034000000	1	0.5809648
10015629448289600000	1	0.22876149
10015698048687000000	0	0.21105757
10015745448500200000	0	0.19108337
10015794997785100000	0	0.08786093
10015857277683800000	0	0.02580579
10015944270539800000	0	0.24537016
10016451205115600000	1	0.3402954
10016492574846400000	0	0.23131874
10016564691567500000	0	0.01447675
10016733002189100000	0	0.23869824
10016873305752100000	0	0.08415666
10016879928769800000	0	0.04000898
10016930469746000000	0	0.18266523
10017024728538900000	0	0.101394
10017246106627400000	1	0.48084709
10000094181510900000	0	0.17092077
10000169349117800000	0	0.25009465
10000371904215100000	0	0.16588736
10051655143706200000	0	0.02021199
10051685815199300000	0	0.14591935

10029116667172200000	0	0.12131329
10029127738769800000	0	0.04442457
10029289556764200000	0	0.17549311
10029327953743800000	0	0.1420728
10029329644865700000	1	0.16175592
1002948443498540000	0	0.01846081
10030054951926300000	1	0.47558811
10030114320949100000	0	0.06773508
10030228488972900000	0	0.16065301
10030476089267500000	0	0.2380196
10030883472074400000	0	0.02670789
10030892756124800000	0	0.04575646
10031105082914100000	0	0.05823678
10031107759553800000	0	0.07627161
10031141284541800000	1	0.26273593
10031235229753400000	0	0.23306581
1003130369744010000	0	0.17179476
10031428004855900000	0	-0.0107273
10031581777340400000	0	0.09803644
10031650492433700000	0	0.05669338
10031998322520600000	0	0.47161686
10032047607672800000	0	0.09394634
10032235721168200000	0	0.19478142
10032264153126100000	0	0.19416958
10032510561377900000	0	0.08567461
10032540532403900000	1	0.37465525
10032841083776200000	0	0.04385381
10033270906068200000	1	0.42760122
10033341691920500000	0	0.25815752
10033414113913900000	0	0.22759597
10033513319859100000	0	0.00166784
10033581435242200000	0	0.27292618
10033581569437700000	1	0.24712189
10033696441237900000	0	0.25131413
10033757931529400000	1	0.13263717
1003378270994620 00	0	0.11627871
10033811465033800000	0	0.02621944
10033908124156100000	0	0.1382404
10034004925027300000	0	0.04310815
10034242394638800000	0	0.20786673
10034295299322600000	0	-0.0056586
10877269648590700000	1	0.4877257
10034702490483200000	0	0.15440348
10035052803926100000	0	0.04597906
10035144090658100000	0	0.07722883
10035306195717500000	0	0.07417542
10035307561141100000	0	0.06350947
10035308909297400000	0	0.14027403
10066977634354300000	1	0.14773156
10067089693566300000	1	0.35838255

10817346796955300000	1	0.06013219
10051974001592400000	0	0.24447645
10052059252101400000	0	0.20869739
10052059370536000000	0	0.2209013
10052083806221300000	0	0.01626592
10052120460676200000	0	0.49710014
10052144210227000000	0	0.20977579
10052369014982500000	0	0.20381096
10052561582444800000	1	0.23215742
10052627552648800000	1	0.49882758
10053027485503800000	0	0.13692594
10053066225125700000	0	0.1047463
10053227075550500000	0	0.22789642
10053390278110800000	0	0.27874413
10053542360995000000	0	0.18735835
10053587463261000000	0	0.17865837
10053613642033900000	0	0.05104952
10053732565772400000	1	0.31604591
10053926221002500000	0	0.2525264
10054412349177600000	1	0.4454447
10054487144727400000	0	0.03030504
10054584303239100000	0	0.08445291
10054692613243400000	0	0.0396942
10054765563239700000	0	0.25716722
10055435465659300000	0	-0.0064707
10055436384941900000	0	0.11454377
10055700314239000000	1	0.35195717
10055850146571500000	0	0.1908987
10055852557156000000	1	0.22735348
10055941435962800000	0	0.1961683
10056162427008600000	0	0.18612072
10056536800612500000	0	0.16858044
10056857270243100000	1	0.33299404
10057092029491400000	0	0.20683323
10057181813599000000	0	0.08542746
10057186875506400000	0	0.03936151
10057276843216200000	0	0.0381389
10057382084158300000	0	0.05707454
10645192256443200000	1	0.58456928
10057814194606200000	0	0.17407988
10058113986759900000	0	0.01910255

10067115719656200000	0	0.07067026
10067131127418100000	0	0.05540413
10067265691595700	0	-0.0324685
10067309148384600000	0	0.04416986
10067426592461500000	1	0.27865836
10067489634429600000	1	0.22568841
10067609744031000000	0	-0.0105381
10067997832137800000	0	0.03937523
10068016967403600000	0	0.21321638
10068147499871400000	0	0.29106376
10068277922049200000	0	0.21175094
10068367305376800000	0	0.1617976
10068417048966500000	0	0.22821099
10068650175042800000	0	0.08317554
10068661785149400000	0	0.2156921
10068932501228900000	0	0.04107684
10710639684452700000	1	0.05882024
10069047454621500000	0	0.28557497
10069255771604700000	1	0.14680184
10069312519787700000	0	0.06565326
10069556708214200000	0	0.1130361
10069582201268700000	0	0.23351677
10069716898669600000	0	0.02231476
10070182618859400000	0	0.13637453
10070328440095900000	1	0.26579022
10070540892129300000	1	0.22235306
10070829472899800000	0	0.08111556
10071001216009800000	1	0.22320728
10071218506447100000	0	0.06592151
10071330487849300000	0	0.00287566
10071358731383900000	0	0.2221617
10071777313976300000	0	0.23238927
10072212624335100000	0	0.0572198
10072393214139400000	0	0.21071409
10072487305193100000	0	0.28853914
10072526372165700000	0	0.08731191
10072631714808000000	1	0.28941634
10072721786261700000	0	-0.0348413
10072726099769900000	0	0.0374186
10072877048246100000	0	0.29609847
10072884772099700000	0	0.5590874

5.2.2 Anàlisi del CTR

El CTR(Click-Through Rate) és el rati que indica la freqüència dels usuaris en què veuen l'anunci i hi fan clic. Com més alt sigui el CTR, més efectiva i rellevant està sent la campanya publicitària.
La fórmula del seu càlcul és:

CTR= (Número d'impressions / Número de clics) x 100

En l'execució de 158 èpoques en el programa, s'han predit una quantitat de clics en la qual s'ha mesurat el seu CTR i s'ha obtingut un percentatge del **6.15343347639 %.**

No hi ha un valor exacte per determinar un bon CTR o un mal CTR, tot depèn de la campanya i de molts altres factors. Tot i així, molts medis d'informació coincideixen en que un CTR al voltant del 10% és un bon resultat. Dit això, es pot arribar a la conclusió que **s'ha obtingut un rati acceptable i que la fiabilitat de la predicció és bona.**

Cal recordar que, com s'ha comentat anteriorment en diverses ocasions, com més èpoques s'executin en el programa, més fiable serà la predicció i major serà el CTR.

6.Avaluació

En aquest apartat es farà l'avaluació del repte segons la normativa de la competició[10]:
La pàgina web del Kaggle proposa com avaluació, el logaritme de la funció de probabilitat per a una distribució aleatòria de Bernoulli per a determinar l'eficàcia del projecte i la seva puntuació. La seva fórmula és:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i))$$

Figura 6. Fórmula de la funció de probabilitat per a una distribució aleatòria de Bernoulli

La funció rep la predicció feta *p* i el clic real *y* i realitza l'avaluació. La puntuació final és la mitja de totes les avaluacions de les prediccions.
A l'aplicar aquesta avaluació, el projecte obté un resultat de **0.39779338013070337 .** En la següent il·lustració, es pot comparar visualment el resultat obtingut d'aquest projecte amb la classificació real de la competició:

#	Δrank	Team Name * in the money	Score	Entries	Last Submission UTC (Best – Last Submission)
1	—	4 Idiots *	0.3791384	273	Mon, 09 Feb 2015 19:37:27 (-43.6h)
2	—	Owen *	0.3803652	94	Mon, 09 Feb 2015 02:34:23 (-0.5h)
3	—	Random Walker *	0.3806351	242	Mon, 09 Feb 2015 10:59:10
4	—	Julian de Wit	0.3810307	44	Mon, 09 Feb 2015 22:19:53
5	—	Dmitry Efimov	0.3810447	265	Mon, 09 Feb 2015 13:11:50
6	—	Marios & Abhishek	0.3828641	221	Mon, 09 Feb 2015 18:59:03
7	—	José A. Guerrero	0.3829448	72	Mon, 09 Feb 2015 23:58:05 (-2.5d)
8	↑1	Nishant Kumar	0.3829694	60	Mon, 09 Feb 2015 17:38:47 (-13.9h)
9	↓1	CodiLime.com	0.3831440	218	Mon, 09 Feb 2015 16:14:21
10	—	cydonia	0.3838636	190	Mon, 09 Feb 2015 15:18:43 (-0.4h)
11	—	Gilberto Titericz Junior	0.3848830	201	Mon, 09 Feb 2015 23:12:20 (-6.5d)
12	—	mymo	0.3851235	66	Mon, 09 Feb 2015 22:37:48
13	—	tks	0.3851583	43	Sun, 08 Feb 2015 01:59:09

Figura 7. Classificació de la competició del Kaggle

Un cop visualitzada la classificació, es pot observar que no s'ha aconseguit el millor resultat, però s'ha obtingut un molt bon resultat en comparació a la part superior de la classificació. **Es pot afirmar que aquest projecte ha aconseguit uns bons resultats i que les prediccions fetes són fiables, complint així amb l'objectiu principal del projecte.**

7. Conclusió

Per concloure el treball, destacarem si els objectius plantejats a l'inici del projecte s'han assolit:

En primer lloc, en la fase d'instal·lació, vaig trobar dificultats al voler treballar en Docker, ja que al ser una eina en constant evolució, existeixen diferents fonts d'informació obsoletes, que generen confusió a l'hora de la seva utilització. Gràcies al suport del meu tutor, vaig poder realitzar una instal·lació correcta permetent poder continuar amb el projecte.

En segon lloc, al realitzar l'aprenentatge de la llibreria Tensorflow, sorgeix el conflicte d'entendre de forma adequada el funcionament i l'estructura d'aquesta llibreria. Mitjançant la cerca d'informació i l'execució de varis exemples, vaig aconseguir entendre la forma adient de programar. També ha sigut molt interessant descobrir l'àmplia varietat d'estratègies i aplicacions que es poden realitzar amb la combinació de Docker i Tensorflow.

En tercer lloc, al desenvolupar el programa, van sorgir dificultats a nivell conceptual sobre l'algoritme a utilitzar, que en aquest cas és el model de regressió lineal, el qual gràcies al meu tutor vaig poder entendre tots els conceptes i components de l'algoritme. Un cop veient el funcionament del programa, va ser molt gratificant poder observar el seu comportament i apreciar els seus resultats positius.

En quart lloc, a l'anàlisi de resultats, es pot arribar a la conclusió que el model de regressió lineal és una bona estratègia a l'hora de realitzar prediccions, ja que en l'observació dels resultats, es pot apreciar que el model funciona i que fa una predicció pròxima a les dades reals. D'altra banda, hi ha l'inconvenient que la seva execució exigeix moltíssimes hores de càlcul i que per aconseguir una gran fiabilitat de predicció, cal executar una gran quantitat d'èpoques.

Abans de posar punt i final al projecte, cal destacar que el repte inicial s'ha pogut assolir gràcies a la motivació i interès que hem va despertar aquest projecte, al suport del tutor i al compliment de tots els objectius plantejats.

Per concloure, actualment i molt més de cara en un futur, veig que aquests tipus d'eines i mètodes, estan adquirint un pes molt important en les grans empreses, a l'hora de poder tractar grans quantitats de dades i adoptar diferents estratègies.

8. Referències

[1] Return on investment
https://es.wikipedia.org/wiki/Retorno_de_la_inversi%C3%B3n

[2] Click-through rate
https://en.wikipedia.org/wiki/Click-through_rate

[3] Pàgina web del Kaggle
https://www.kaggle.com/c/avazu-ctr-prediction

[4] Informació i pàgina web oficial de Docker
https://www.docker.com/
https://es.wikipedia.org/wiki/Docker_(software)

[5] Tensors
https://es.wikipedia.org/wiki/C%C3%A1lculo_tensorial

[6] Aprenentatge de Tensorflow
http://learningtensorflow.com/

[7] Llibre “Hello World en Tensorflow” d’en Jordi Torres
http://jorditorres.org/libro-hello-world-en-tensorflow/

[8] Pàgina web oficial de Tensorflow
https://www.tensorflow.org/

[9] Funció de cost Mean Squad Error
https://en.wikipedia.org/wiki/Mean_squared_error

[10] Avaluació de la competició
https://www.kaggle.com/c/avazu-ctr-prediction/details/evaluation

9. Índex de figures

10.Annexos

Annex1: Codi font del programa.

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from csv import DictReader
from math import exp, log, sqrt

def logloss(p, y):
    ''' FUNCTION: Bounded logloss

        INPUT:
            p: our prediction
            y: real answer

        OUTPUT:
            logarithmic loss of p given y
    '''
    p = max(min(p, 1. - 10e-15), 10e-15)
    return -log(p) if y == 1. else -log(1. - p)

def data(path, D):
    ''' GENERATOR: Apply hash-trick to the original csv row
                   and for simplicity, we one-hot-encode everything
        INPUT:
            path: path to training or testing file
            D: the max index that we can hash to

        YIELDS:
            ID: id of the instance, mainly useless
            x: a list of hashed and one-hot-encoded 'indices'
               we only need the index since all values are either 0 or 1
            y: y = 1 if we have a click, else we have y = 0
    '''
    for t, row in enumerate(DictReader(open(path))):
        # process ID
        ID = row['id']
        del row['id']

        # process clicks
        y = 0.
        if 'click' in row:
            if row['click'] == '1':
                y = 1.
            del row['click']

        # build x
        x = np.zeros(D)
        for key in row:
            value = row[key]
```

```
        # one-hot encode everything with hash trick
            index = abs(hash(key + '_' + value)) % D
            x[index]+=1

        yield t, ID, np.expand_dims(x,axis=0), np.expand_dims(y,axis=0)

#Paths
train = 'train.csv'               # path to training file
test = 'test.csv'                 # path to testing file
submission = 'submissionFINAL.csv'  # path of to be outputted submission file

learning_rate = tf.Variable(0.001, name='learning_rate')  # learning rate
D = 5000     # number of weights to use
epoch = 200   # learn training data for N passes

cost_history = []
aux=0.
cont = 1.

#Placeholders
X = tf.placeholder(tf.float32,[1, D])
Y = tf.placeholder(tf.float32,[1])

#Variables
W = tf.Variable(tf.random_normal(shape=[D,1],stddev=0.01), name="weight")
b = tf.Variable(tf.zeros([1]), name="bias")

#Prediction Model
p = tf.matmul(X, W) + b

#Cost Function
cost = tf.reduce_mean(tf.square(p-Y))

#Learning function
learner = tf.train.FtrlOptimizer(learning_rate).minimize(cost)

#Initialization of variables
init = tf.initialize_all_variables()

#Create Session
sess = tf.Session()
sess.run(init)

#Start Training
for e in range(epoch):
    for (t, ID, x, y) in data(train, D):
        sess.run(learner, feed_dict={X: x, Y: y})
        training_cost = sess.run(cost, feed_dict={X: x, Y: y})
        aux+=training_cost
        cont +=1
```

```
    #Total Training Cost Epoch
    print("Epoch: ",e)
    print("Training Cost Epoch = ", aux/cont)
    cost_history.append(aux/cont)

#Total Training Cost
print("Total Training Cost = ", aux/cont)

#Graph of the learning curve
plt.plot(cost_history)
plt.xlabel("Epochs")
plt.ylabel("Cost")
plt.show()

#Writing the prediction
acum_ctr=0.
acum_logloss=0.
count_eval=0.
with open(submission, 'w') as outfile:
    outfile.write('id,real click,click predicted\n')
    for (t, ID, x, y) in data(test, D):
        prediction = sess.run(p[0],feed_dict={X: x,Y: y})
        outfile.write('%s,%s,%s\n' % (ID, y, prediction))
        if float(prediction) >= 0.5:
            acum_ctr+=1
        acum_logloss +=logloss(prediction,y)
        count_eval+=1

#Calculate CTR
ctr = (acum_ctr/count_eval)*100
print("CTR = ",ctr)

#Final Score
print("Final Score(Logarithmic Loss)= ", acum_logloss/count_eval)

print("END OF PROGRAM!")
```