TensorFlow

Introducción y Aplicaciones

Daniel Jiménez M.

Universidad Nacional de Colombia

22 -10 -2020

Anotación especial

Esta presentación será solo teórica, los ejercicios y de más se trabajaran en un script.

¿Qué es TensorFlow?

Dada la evolución en los modelos de inteligencia artificial y machine learning, Google Brain Labs creo, una librería de código abierto que trabaja con alto desempeño de computación para el desarrollo de aprendizaje profundo (deep Learning) y aprendizaje de máquina.

¿Qué es TensorFlow?

TensorFlow es :

Una creación de Google Brain;
Es una libreria del tipo OpenSource;
Trabaja bajo API;
Se ejecuta a través de C++
Un Framework que permite la creación de modelos de Machine Learning y Deep Learning;
Como Framework funciona como una biblioteca;
Su trabajo se basa en la construcción de redes neuronales;

¿Qué es TensorFlow?

Tensorflow es popular para el trabajo de :

NLP: Comprensión de textos;
RNN (Redes Neuronales recurrentes): Análisis de datos en series temporales, la dimensión del tiempo es importante;
Visual recognition: Reconocimiento de Imagenes

¿Qué es TensorFlow?

TensorFlow funciona con Tensores que son arrays de multiples dimensiones.

$$(A * B) + C$$

Donde :

A,B,C : son arrays
{A,B,C}: forman el tensor

Proceso del aprendizaje profundo

Carga de datos;
Procesamiento de datos;
Entrenamiento de los datos;
Validación de los modelos;
Generar Resultados.

Proceso del aprendizaje profundo

El mapa anterior nos obliga a entender lo siguiente:

Red Neuronal : Suma de neuronas o nodos que albergan información y procesos para la toma de decisiones.

Algoritmos que estan destinados a encontrar patrones en los datos.

Neurona : Arquitectura de modelos estadísticos y matemáticos que comprende la información de una manera jerárquica.

Proceso del aprendizaje profundo

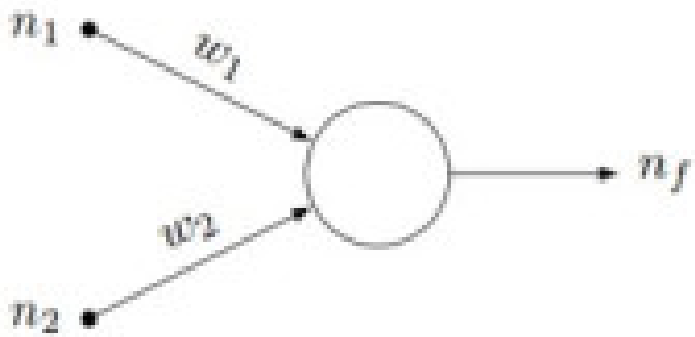


Gráfico de un perceptron

Proceso del aprendizaje profundo

Las redes neuronales son de gran uso en datos no estructurados como:

Audios;
Imagenes;
Textos -> Caso que nos interesa!

¿Cómo funciona una red Neuronal?

Las redes neuronales tiene :

Entradas (x_i) : Variables explicativas
Pesos (w_{ij}) : Intensidad de la interacción entre neuronas
Propagación σ: Arquitectura de la Red

¿Cómo funciona una red Neuronal?

Suponga una regresión lineal

$$h_i(t) = \sum_{j}^{i=1} w_{ij} X_j$$

¿Cómo funciona una red Neuronal?

Función de Activación: Es la función que permite construir o predecir un dato con base a los patrones determinados.

Función de activación

Proporciona el estado de activación de la neurona en función del estado anterior o del actual

$a_i(t) = f_i(\alpha_i(t-1), h_i(t))$

La anterior formula indica que la función de activación puede que no dependa de su pasado, sino de su estado actual.

Funciones de Activación

Las funciones de activación se dividen en :

Lineales (o identidad) : Se usa cuando los outputs necesitan una regresión lineal (predecir el precio de venta de un carro en el 2021)

No lineales (o escalón) : Se usa cuando trabajamos con salidas categóricas.

Sigmoide : Su función es calcular una probabilidad, esta en el rango entre $\{0, 1\}$, se suele usar en problemas de clasificación, pero .... no es muy buena !
Hiperbólica : Sirve para clasificación pero su ventaja en comparación a la sigmoide es que su rango esta entre $\{-1, 1\}$
Relu (La favorita): Permite que las neuronas aprendan rápido y cuando en el proceso de aprendizaje la derivada indica poca información, genera la muerte de la neurona.
Softmax: Se usa para temas de Clasificación

¿Cómo trabajan las Neuronas?

$$(X, Y) \Longrightarrow X \in |R^{n>0} \text{ y } Y \in \{0, 1\}$$

Entonces

$$samples_m = \sum_{i}^{n} (x^{(i,n)}, y^{(i,n)})$$

Se diseña como la partición para el entrenamiento de los datos.

Función de Costo

La función de costo es la interpretación de la medición de las predicciones de los escenarios

$$\text{Loss Function} = \ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$

Función de Costo

Suponga que

$$y = 0 : \ell(\hat{y}, y) = -log(1 - \hat{y}) \Longrightarrow \log 1 - y \approx 0$$

Función de Costo

En el caso anterior la función de costo toma la forma :

$$J_{(w,b)} = \frac{1}{m} \sum \ell(\hat{y}, y^i)$$

Entonces

$$J_{(w,b)} = -1/m \sum_{i=1}^{m} [y^i log\hat{y} + (1 - y^{(i)} log(1 - \hat{y}^i))]$$

La demostración del caso en que $y = 1$ queda acargo del estudiante.

Gradiente Descendiente

Es un algoritmo de optimización iteractivo de primer orden para encontrar los mínimos locales de una función diferenciable (derivable), esto quiere decir que es la función que encuentra cual es el valor mínimo para la función de costo.

Una definición enciclopedica es :

"Función que encuentra w que miniza la función de coste"

Gradiente Descendiente

Recuerde que w es el parámetro de interés

$$J_{(w,b)} = -1/m \sum_{i=1}^{m} [y^i log\hat{y} + (1 - y^{(i)} log(1 - \hat{y}^i))]$$

Entonces

$$w := w - \alpha \frac{dJ(w,b)}{dw}$$

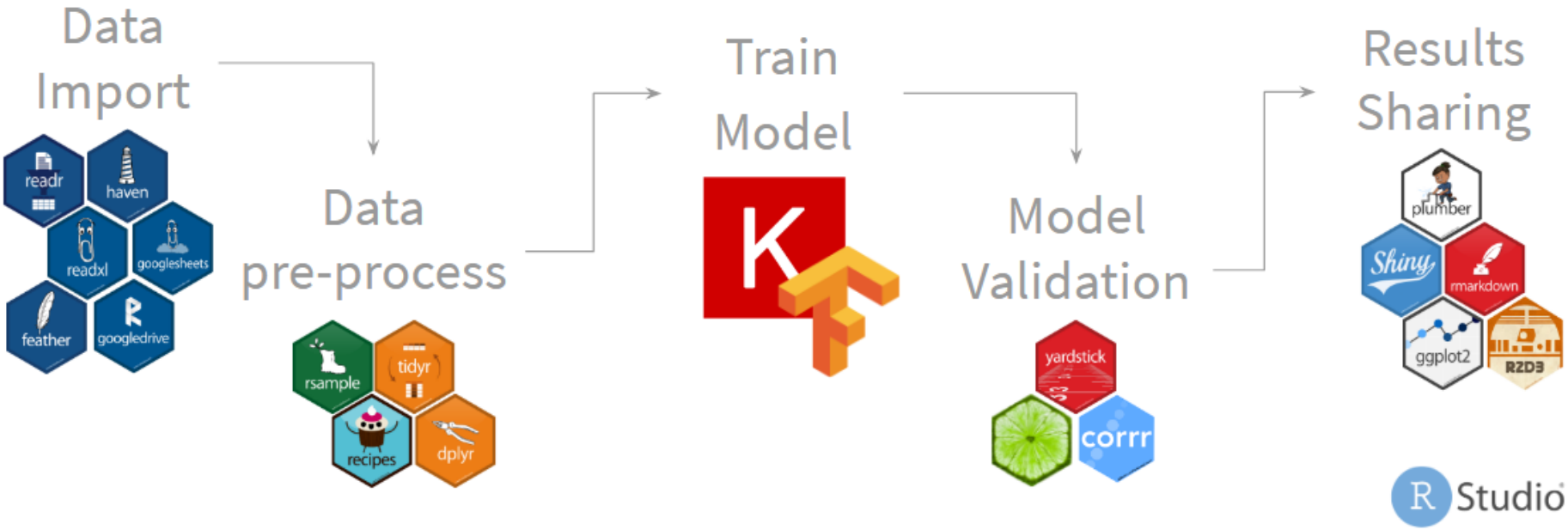
Recomendaciones

No es necesario que haga todas las matemáticas;
Pero si es necesario que las entienda !!
Repasen Derivadas y Matrices.

¿Qué hace Tensorflow?

"Entrena los modelos con la matemáticas antes expuesta"

¿Qué hace Tensorflow?



Instalar Tensorflow

Los comandos para instalar y trabajar con `tensorflow` son:

```
install.packages("tensorflow")
library(tensorflow)
tf_config()# Esto es para manejar la configuración
```

Tensorflow - Flujo de trabajo

Para trabajar con este framework es necesario iniciar la sección de la siguiente manera

```
library(tensorflow)
session=tf$Session()
print(session)
```

Tensorflow - Flujo de trabajo

Cuando termine de trabajar deberá correr el siguiente comando

```
session$close()
```

Tensorflow - Flujo de trabajo

Este sería un ejemplo de flujo de trabajo

```
library(tensorflow) # Se llama a la librería
## Crear la sección de trabajo
seccion<-tf$Session()
## Creamos una constante de prueba - esto se hace para practi
Hola_mundo<-tf$constant('Hola Mundo!!!')
## Se corre el trabajo hecho
print(seccion$run(Hola_mundo))
## Se finaliza el proceso
seccion$close()
```

```
## [1] "Hola Mundo!!!"
```

Tensorflow Ejemplo

Tomaremos un ejemplo de la conferencia de R Studio con Keras & Tensorflow

Si quiere ver el notebook de la presentación este es el link :https://github.com/sol-eng/tensorflow-w-r

Si quiere ver el video este es el link: https://rviews.rstudio.com/2018/04/03/r-and-tensorflow-presentations/

Tensorflow Ejemplo

Cargue las siguientes librerías

```
library(keras)
library(lime)
library(tidyverse)
library(rsample)
library(recipes)
library(yardstick)
library(corrr)
library(tensorflow)
```

Tensorflow Ejemplo

Cargue los datos

```
library(tidyverse)
if(!file.exists("customer_churn.csv")){
  download.file(
    "https://raw.githubusercontent.com/rstudio/keras-customer-
    "customer_churn.csv"
  )
}
churn_data_raw <- read_csv("customer_churn.csv")
```

```
##
## -- Column specification --------------------------------
## cols(
##   .default = col_character(),
##   SeniorCitizen = col_double(),
##   tenure = col_double(),
##   MonthlyCharges = col_double(),
```

Tensorflow Ejemplo

Genere los ambientes de entrenamiento

```
library(rsample)
set.seed(100)
train_test_split <- initial_split(
  churn_data_raw,
  prop = 0.3)
train_tbl <- training(train_test_split)
test_tbl  <- testing(train_test_split)
```

Tensorflow Ejemplo

Genere un pre procesamiento de los datos

```
library(recipes)
rec_obj <- train_tbl %>%
  recipe(Churn ~ .) %>%
  step_rm(customerID) %>%
  step_naomit(all_outcomes(), all_predictors()) %>%
  step_discretize(tenure, options = list(cuts = 6)) %>%
  step_log(TotalCharges) %>%
  step_mutate(Churn = ifelse(Churn == "Yes", 1, 0)) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_center(all_predictors(), -all_outcomes()) %>%
  step_scale(all_predictors(), -all_outcomes()) %>%
  prep()
```

Tensorflow Ejemplo

```
summary(rec_obj)%>%
  head()
```

```
## # A tibble: 6 x 4
##   variable       type    role      source
##   <chr>          <chr>   <chr>     <chr>
## 1 SeniorCitizen  numeric predictor original
## 2 MonthlyCharges numeric predictor original
## 3 TotalCharges   numeric predictor original
## 4 Churn          numeric outcome   original
## 5 gender_Male    numeric predictor derived
## 6 Partner_Yes    numeric predictor derived
```

Tensorflow Ejemplo

Ejecute los datos

```
x_train_tbl <- juice(rec_obj, all_predictors(), composition =
y_train_vec <- juice(rec_obj, all_outcomes()) %>% pull()
```

Tensorflow Ejemplo

Cree el escenario de los datos

```
baked_test <- bake(rec_obj, test_tbl)
x_test_tbl <- baked_test %>%
  select(-Churn) %>%
  as.matrix()
y_test_vec <- baked_test %>%
  select(Churn) %>%
  pull()
```

Tensorflow Ejemplo

Llame a tensorflow y Keras

```
library(tensorflow)
library(keras)
```

Tensorflow Ejemplo

Cree la red neuronal

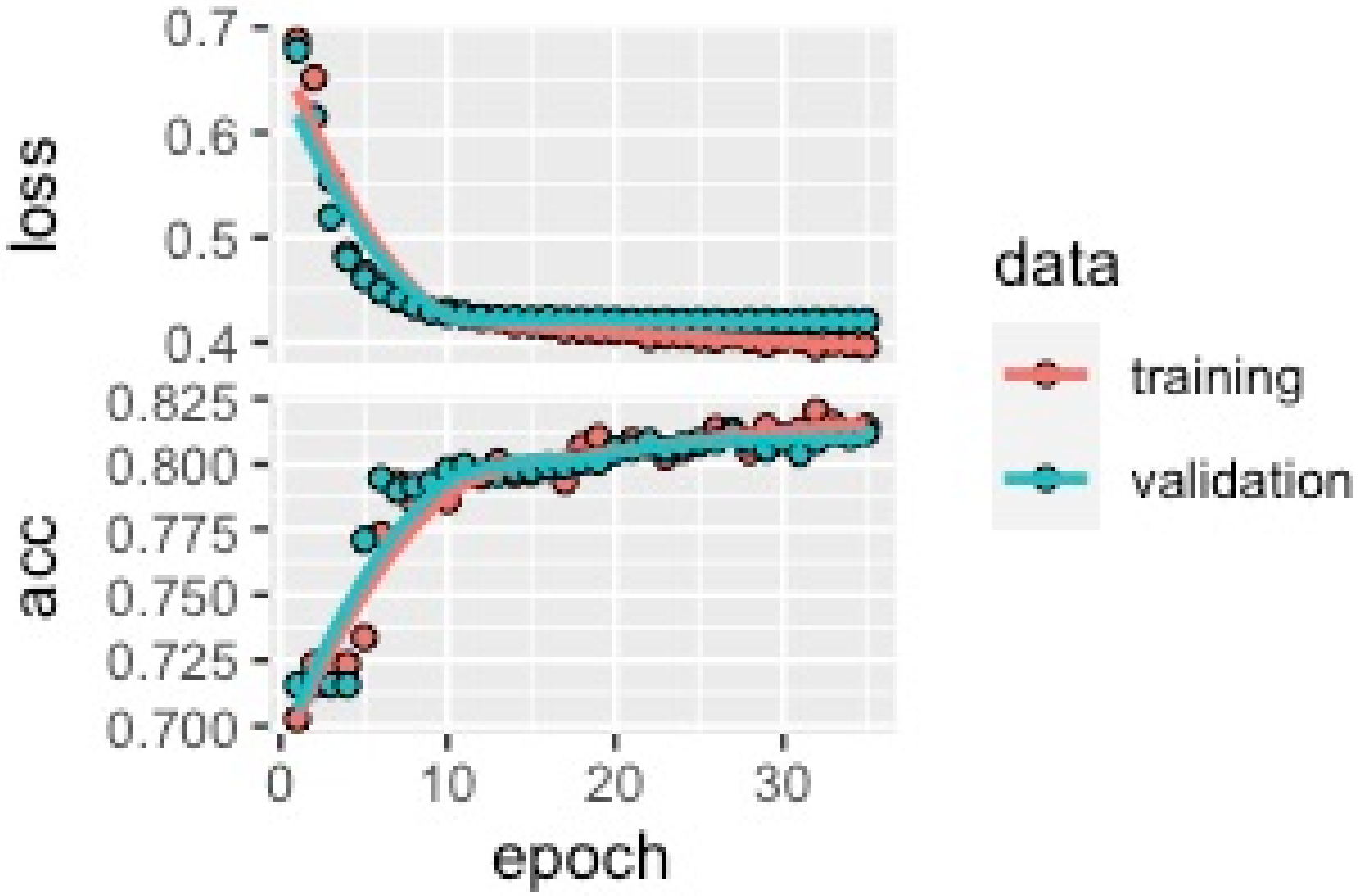
```
model_keras <- keras_model_sequential() %>%
  layer_dense(
    units = 16,
    kernel_initializer = "uniform",
    activation = "relu",
    input_shape = ncol(x_train_tbl)) %>%
  layer_dropout(rate = 0.1) %>%
  layer_dense(
    units = 16,
    kernel_initializer = "uniform",
    activation = "relu") %>%
  layer_dropout(rate = 0.1) %>%
  layer_dense(
    units = 1,
    kernel_initializer = "uniform",
    activation = "sigmoid") %>%
```

Tensorflow Ejemplo

Ajuste el modelo

```
history <- fit(
  object = model_keras,
  x = x_train_tbl,
  y = y_train_vec,
  batch_size = 50,
  epochs = 35,
  validation_split = 0.30,
  verbose = 0
)
```

Tensorflow Ejemplo



Tensorflow Ejemplo

Con este modelo entrenado, bonito y ajustado haga predicciones

```
yhat_keras_class_vec <- model_keras %>%
  predict_classes(x_test_tbl) %>%
  as.factor() %>%
  fct_recode(yes = "1", no = "0")
```

Tensorflow Ejemplo

Vea la probabilidad de la ocurrencia

```
yhat_keras_prob_vec  <- model_keras %>%
  predict_proba(x_test_tbl) %>%
  as.vector()
test_truth <- y_test_vec %>%
  as.factor() %>%
  fct_recode(yes = "1", no = "0")

yhat_keras_prob_vec%>%
  tbl_df()%>%
  head()
```

Tensorflow Ejemplo

Compare los resultados

```
estimates_keras_tbl <- tibble(
  truth      = test_truth,
  estimate   = yhat_keras_class_vec,
  class_prob = yhat_keras_prob_vec
)
```

truth	estimate	class_prob
no	yes	0.7730685
no	no	0.0496711
no	no	0.0612399
yes	yes	0.7358598
no	no	0.3896118
yes	yes	0.5731490

Tensorflow Ejemplo

Cree una matriz de confusión