Redes Neuronales Artificales

Instalar Theano

pip install --upgrade --no-deps git+git://github.com/Theano/Theano.git

Instalar Tensorflow y Keras

conda install -c conda-forge keras

Parte 1 - Pre procesado de datos

Cómo importar las librerías

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Importar el data set

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947

Enter your authorization code:
··········
Mounted at /content/drive
```

```
dataset = pd.read_csv('/content/drive/My Drive/Curso de Deep Learning de la A a la Z/datas

X = dataset.iloc[:, 3:13].values
y = dataset.iloc[:, 13].values
```

```
dataset.iloc[0,:].values
```

```
array([1, 15634602, 'Hargrave', 619, 'France', 'Female', 42, 2, 0.0, 1, 1,
       1, 101348.88, 1], dtype=object)
```

```
X[0,:]
```

```
array([0.0, 0.0, 619, 0, 42, 2, 0.0, 1, 1, 1, 101348.88], dtype=object)
```

Codificar datos categóricos

```
from sklearn.preprocessing import LabelEncoder
labelencoder_X_1 = LabelEncoder()
X[:, 1] = labelencoder_X_1.fit_transform(X[:, 1])
labelencoder_X_2 = LabelEncoder()
X[:, 2] = labelencoder_X_2.fit_transform(X[:, 2])
```

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer

transformer = ColumnTransformer(
    transformers=[
        ("Churn_Modelling",        # Un nombre de la transformación
         OneHotEncoder(categories='auto'), # La clase a la que transformar
         [1]            # Las columnas a transformar.
         )
    ], remainder='passthrough'
)

X = transformer.fit_transform(X)
X = X[:, 1:]
```

Dividir el data set en conjunto de entrenamiento y conjunto de testing

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state =
```

Escalado de variables

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

Parte 2 - Construir la RNA

```
# Importar Keras y librerías adicionales
import keras
from keras.models import Sequential
from keras.layers import Dense
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the %t

```
# Inicializar la RNA
classifier = Sequential()

# Añadir las capas de entrada y primera capa oculta
classifier.add(Dense(units = 6, kernel_initializer = "uniform",
                     activation = "relu", input_dim = 11))

# Añadir la segunda capa oculta
classifier.add(Dense(units = 6, kernel_initializer = "uniform",  activation = "relu"))

# Añadir la capa de salida
classifier.add(Dense(units = 1, kernel_initializer = "uniform",  activation = "sigmoid"))

# Compilar la RNA
classifier.compile(optimizer = "adam", loss = "binary_crossentropy", metrics = ["accuracy"

# Ajustamos la RNA al Conjunto de Entrenamiento
classifier.fit(X_train, y_train,  batch_size = 10, epochs = 100)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/pyt
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

Epoch 1/100
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensc

8000/8000 [==============================] - 2s 231us/step - loss: 0.5070 - acc: 0
Epoch 2/100
8000/8000 [==============================] - 1s 124us/step - loss: 0.4087 - acc: 0
Epoch 3/100
8000/8000 [==============================] - 1s 126us/step - loss: 0.3945 - acc: 0
```

```
Epoch 4/100
8000/8000 [==============================] - 1s 122us/step - loss: 0.3857 - acc: 0
Epoch 5/100
8000/8000 [==============================] - 1s 125us/step - loss: 0.3794 - acc: 0
Epoch 6/100
8000/8000 [==============================] - 1s 123us/step - loss: 0.3749 - acc: 0
Epoch 7/100
8000/8000 [==============================] - 1s 124us/step - loss: 0.3705 - acc: 0
Epoch 8/100
8000/8000 [==============================] - 1s 129us/step - loss: 0.3671 - acc: 0
Epoch 9/100
8000/8000 [==============================] - 1s 123us/step - loss: 0.3644 - acc: 0
Epoch 10/100
8000/8000 [==============================] - 1s 124us/step - loss: 0.3618 - acc: 0
Epoch 11/100
8000/8000 [==============================] - 1s 125us/step - loss: 0.3595 - acc: 0
Epoch 12/100
8000/8000 [==============================] - 1s 126us/step - loss: 0.3582 - acc: 0
Epoch 13/100
8000/8000 [==============================] - 1s 125us/step - loss: 0.3566 - acc: 0
Epoch 14/100
```

Parte 3 - Evaluar el modelo y calcular predicciones finales

Predicción de los resultados con el Conjunto de Testing

```
y_pred  = classifier.predict(X_test)
y_pred = (y_pred>0.5)
```

Predecir una nueva observación

Utiliza nuestro modelo de RNA para predecir si el cliente con la siguiente información abandonará el banco:

- Geografia: Francia
- Puntaje de crédito: 600
- Género masculino
- Edad: 40 años de edad
- Tenencia: 3 años.
- Saldo: $ 60000
- Número de productos: 2
- ¿Este cliente tiene una tarjeta de crédito? Sí
- ¿Es este cliente un miembro activo? Sí
- Salario estimado: $ 50000

Entonces, ¿deberíamos decir adiós a ese cliente?

```
new_prediction·=·classifier.predict(sc_X.transform(np.array([[0,0,600,·1,·40,·3,·60000,·2,
print(new_prediction)
print(new_prediction·>·0.5)
```

```
[[0.04022561]]
[[False]]
```

Elaborar una matriz de confusión

```
from sklearn.metrics import confusion_matrix
cm·=·confusion_matrix(y_test,·y_pred)
```

```
(cm[0][0]+cm[1][1])/cm.sum()
```

```
0.866
```