Regresión Lineal y Logística. Redes Neuronales. Introducción a Keras.

Práctica 2

Requisitos

La distribución de Python Anaconda, con una versión de Python 3.6 o superior. Podés encontrar un tutorial sobre como instalar Anaconda y ejecutar python o jupyter en: http://facundoq.github.io/courses/aa2018/jupyter.html

Además, debés instalar las librerías *jupyter*, *ipython*, *keras* (versión 2.x o superior) y *tensorflow* (versión 1.4). La instalación puede realizarse con el comando:

pip install ipython jupyter tensorflow keras

Objetivos

Estos ejercicios tienen como objetivo que te familiarices con la librería Keras implementando modelos simples de Regresión Lineal y Logística, y luego de Redes Neuronales.

Para comenzar, en los dos primeros ejercicios, **no** utilizaremos Keras, pero ofrecemos una implementación muy simple y didáctica de regresión lineal con una variable de entrada (ejercicio 1) o múltiples (ejercicio 2). Si bien ya proveemos la implementación del descenso de gradiente y el cálculo de derivadas, recomendamos estudiar brevemente las mismas.

En el ejercicio 3, haremos exactamente lo mismo que en el 2 pero utilizando Keras. El objetivo de esta progresión es comprender mejor qué hace Keras internamente.

Preparación

Desde una terminal de Anaconda ubicarse en el directorio de la práctica donde se encuentra los archivos con extensión ".ipynb" (notebooks) y escribir: "jupyter notebook". Desde el entorno web, acceder a la dirección del servidor (generalmente http://localhost:8888/) para abrir los archivos.

Ejercicio 1 En este ejercicio deberás implementar la función de *predicción* o función *forward* de un modelo de regresión lineal con 1 variable de entrada y 1 de salida. Verifica que la implementación sea correcta con los casos de prueba, y luego que el modelo entrene correctamente.

Archivo: Ejercicio 1 - Regresion Lineal Univariada.ipynb

Ejercicio 2 Igual que el ejercicio 1, pero con un modelo con múltiples variables de entrada y solo una de salida.

Archivo: Ejercicio 2 - Regresion Lineal.ipynb

Ejercicio 3 El código de este ejercicio reimplementa el modelo del ejercicio 2 pero con la librería Keras. Si bien no hay nada que implementar, ejecutá el código y comparalo con el anterior. Probá también distintas combinaciones de la tasa de aprendizaje (**lr**), la cantidad de iteraciones (**epochs**) y el tamaño del batch (**batch_size**) para ver su efecto en el aprendizaje.

Archivo: Ejercicio 3 - Regresion Lineal con Keras.ipynb

Ejercicio 4 El código de ejercicio implementa regresión logística con Keras. Si bien no hay nada que implementar, el objetivo del ejercicio es comprender cómo utilizar la API de Keras para entrenar un modelo con la función de activación de la regresión logística y la función de error asociada, así como la forma de codificar la salida.

Archivo: Ejercicio 4 - Regresion Logistica con Keras.ipynb

Ejercicio 5 En este ejercicio, proponemos entrenar modelos de regresión logística para varios conjuntos de datos de clasificación, y visualizar sus resultados. Los conjuntos de datos son de problemas de clasificación de dos o varias clases, y nos permitirán comprender los límites de la regresión logística (y lineal). Los conjuntos de datos *iris* y *diabetes* tienen más de 2 dimensiones de entrada por lo que no podrán ser visualizados.

Anota los mejores resultados y cómo los obtuviste.

Archivo: Ejercicio 5 - Datasets de Clasificación con Keras.ipynb

Conjuntos de datos:

Archivo	Variables de entrada	Cantidad de clases	Mejor accuracy	Mejor loss	épocas	Tasa de aprendizaje
2_clases_simple.csv						
6_clases_dificil.csv						
circulos.csv						
diabetes.csv						
iris.csv						
moons_mis-scaled.csv						
moons.csv						

Ejercicio 6 En este ejercicio, haremos lo mismo que en el 5, pero ahora utilizando redes neuronales, es decir, varias capas. Prueba variando la cantidad de capas y la función de activación de cada una.

Anota los mejores resultados y la topología.

Podés escribir la topología de una red de tres capas, donde la primera tiene activación ReLU y 3 salidas, la segunda TanH y 5 salidas, y la tercera softmax y 2 salidas como:
Dense(3,'relu') - Dense(5,'tanh') - Dense(2,'softmax').

Archivo: Ejercicio 6 - Datasets de Clasificación con Keras y Redes Neuronales.ipynb

C*onjuntos de datos*:

Archivo	Mejor accuracy	Mejor loss	Iteraciones	Tasa de aprendizaje	Topología
2_clases_simple.csv					
6_clases_dificil.csv					
circulos.csv					
diabetes.csv					
iris.csv					
moons_mis-scaled.csv					
moons.csv					