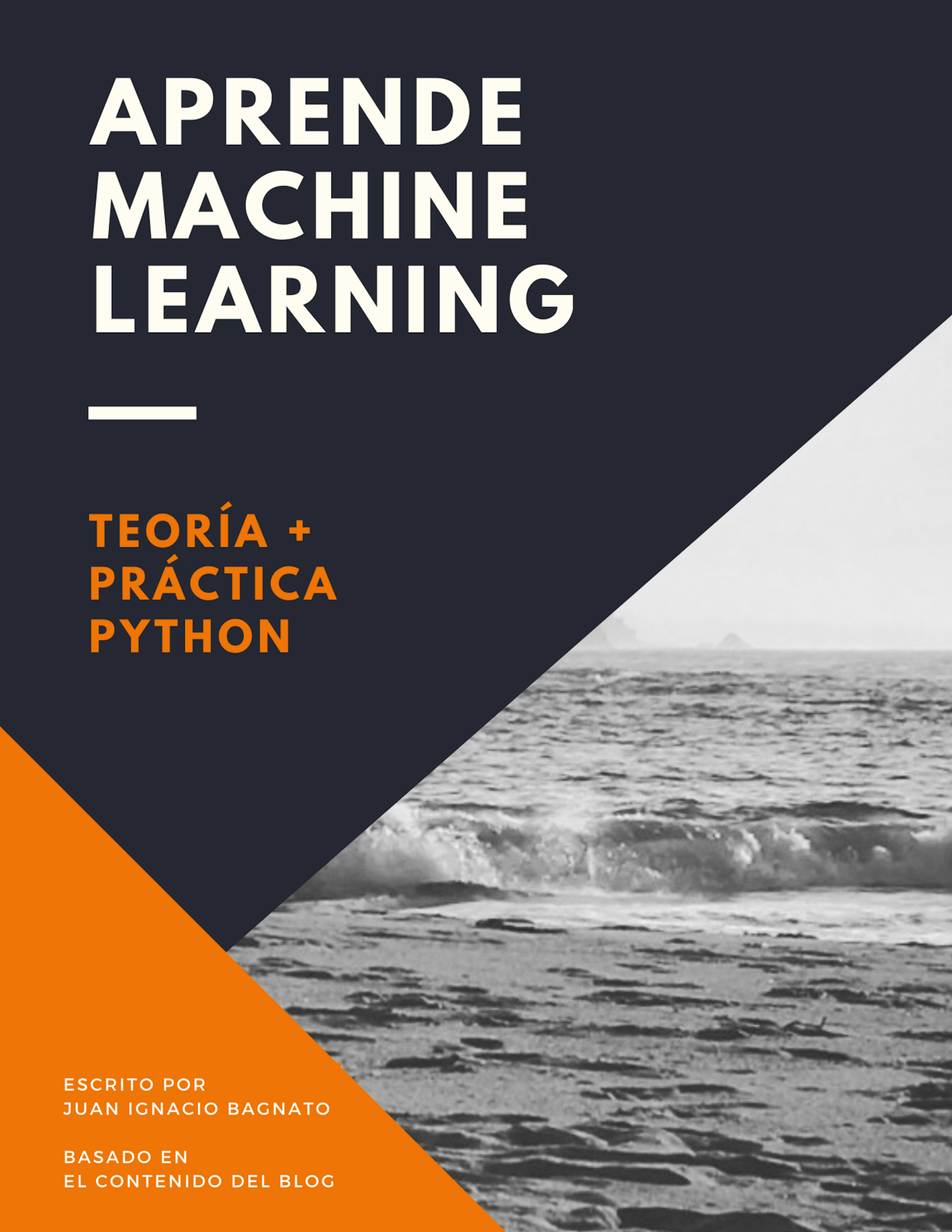
APRENDE MACHINE LEARNING

TEORÍA + PRÁCTICA PYTHON



ESCRITO POR
JUAN IGNACIO BAGNATO

BASADO EN
EL CONTENIDO DEL BLOG

Aprende Machine Learning en Español

Teoría + Práctica Python

Juan Ignacio Bagnato

Este libro está a la venta en http://leanpub.com/aprendeml

Esta versión se publicó en 2022-05-07

¡Tuitea sobre el libro!

Por favor ayuda a Juan Ignacio Bagnato hablando sobre el libro en Twitter!

El hashtag sugerido para este libro es #aprendeML.

Descubre lo que otra gente dice sobre el libro haciendo clic en este enlace para buscar el hashtag en Twitter:

#aprendeML

A mis padres Angel y Graciela, que siempre me dieron las herramientas para poder formarme y me enseñaron que lo importante es el camino.

Índice general

Nota Inicial

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Repositorio

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Tu opinión

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es el Machine Learning?

Veamos algunas definiciones existentes sobre Machine Learning para intentar dar comprensión a esta revolucionaria materia.

Definiendo Machine Learning

El Machine Learning -traducido al Español como Aprendizaje Automático ó Aprendizaje de máquinas- es un subcampo de la Inteligencia Artificial que busca resolver el "cómo construir programas de computadora que mejoran automáticamente adquiriendo experiencia".

Esta definición implica que el programa que se crea con ML no necesita que el programador indique explícitamente las reglas que debe seguir para lograr su tarea si no que este mejora automáticamente.

En los últimos años han surgido grandes volúmenes de datos de diversas fuentes públicas -big data- y el Aprendizaje Automático relacionado al campo estadístico consiste en extraer y reconocer patrones y tendencias para comprender qué nos dicen los datos. Para ello, se vale de algoritmos que pueden procesar Gygas y/o Terabytes en tiempos razonables y obtener información útil.

Una Definición Técnica

Podemos encontrar la siguiente definición técnica sobre Aprendizaje Automático:

> A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

La experiencia E hace referencia a grandes volúmenes de datos recolectados (Big Data) para la toma de decisiones T y la forma de medir su desempeño P para comprobar que esos algoritmos mejoran con la adquisición de más experiencia.

Diagrama de Venn

Drew Conway creó un simpático diagrama de Venn en el que inerrelaciona diversos campos. Aquí copio su versión al Español:

Diagrama de Venn

En esta aproximación al ML, podemos ver que es una intersección entre conocimientos de Matemáticas y Estadística con Habilidades de Hackeo del programador.

Aproximación para programadores

Los programadores sabemos que los algoritmos de búsqueda pueden tomar mucho tiempo en concluir y que cuanto mayor sea el espacio de búsqueda crecerán exponencialmente las posibilidades de combinación de una respuesta óptima, haciendo que los tiempos de respuesta tiendan al infinito o que tomen más tiempo de lo que un ser humano pueda tolerar (por quedarse sin vida o por impaciencia).

Para poder resolver este tipo de situaciones surgen soluciones de tipo heurísticas que intentan dar «intuición» al camino correcto a tomar para resolver un problema. Estos logran buenos resultados en tiempos menores de procesamiento pero muchas veces su intuición es arbitraria y pueden fallar.

Los algoritmos de ML intentan utilizar menos recursos computacionales para «entrenar» grandes volúmenes de datos e ir aprendiendo por sí mismos. Podemos dividir el ML en 2 grandes categorías: Aprendizaje Supervisado o Aprendizaje No Supervisado. Hay una tercer categoría llamada "Aprendizaje por Refuerzo" pero no será tratada en este libro.

Entre los Algoritmos más utilizados en Inteligencia Artificial encontramos:

- Arboles de Decisión
- Regresión Lineal
- Regresión Logística
- k Nearest Neighbor
- PCA / Principal Component Analysis
- SVM
- Gaussian Naive Bayes
- K-Means
- Redes Neuronales Artificiales
- Aprendizaje Profundo ó Deep Learning

Una mención especial a las Redes Neuronales Artificiales

Una mención distintiva merecen las "RNAs" ya que son algoritmos que imitan al comportamiento de las neuronas humanas y su capacidad de sinápsis para la obtención de resultados, interrelacionando diversas capas de neuronas para darle mayor poder de aprendizaje.

Aunque este código existe desde hace más de 70 años, en la última década han evolucionado notoriamente (en paralelo a la mayor capacidad tecnológica de procesamiento, memoria RAM y disco, la nube, etc.) y están logrando impresionantes resultados para analizar textos y síntesis de voz, traducción automática de idiomas, procesamiento de lenguaje natural, visión artificial, análisis de riesgo, clasificación y predicción y la creación de motores de recomendación.

Resumen

El Machine Learning es una nueva herramienta clave que posibilitará el desarrollo de un futuro mejor para la humanidad brindando inteligencia a robots, coches y hogares. Las Smart Cities, el IOT (Internet of things) ya se está volviendo una realidad y también las aplicaciones de Machine Learning en asistentes como Siri, las recomendaciones de Netflix o sistemas de navegación autónoma en drones. Para los ingenieros o informáticos es una disciplina fundamental para modelar, construir y transitar este nuevo futuro.

Instalar el Ambiente de Desarrollo Python

Para programar tu propia Máquina de Inteligencia Artificial necesitarás tener listo tu ambiente de desarrollo local, en tu computadora de escritorio o portátil. En este capítulo explicaremos una manera sencilla de obtener Python y las librerías necesarias para programar como un Científico de Datos y poder utilizar los algoritmos más conocidos de Machine Learning.

¿Por qué instalar Python y Anaconda en mi ordenador?

Python es un lenguaje sencillo, rápido y liviano y es ideal para aprender, experimentar, practicar y trabajar con machine learning, redes neuronales y aprendizaje profundo.

Utilizaremos la Suite gratuita de Anaconda que nos facilitará la tarea de instalar el ambiente e incluye las Jupyter Notebooks, que es una aplicación web que nos ayudará a hacer ejercicios paso a paso en Machine Learning, visualizacion de datos y escribir comentarios tal como si se tratase de un cuaderno de notas de la universidad.

Esta Suite es multiplataforma y se puede utilizar en Windows, Linux y Macintosh.

Agenda

Nuestra agenda de hoy incluye:

1. Descargar Anaconda
2. Instalar Anaconda
3. Iniciar y Actualizar Anaconda
4. Actualizar paquete scikit-learn
5. Instalar Librerías para Deep Learning

Comencemos!

1. Descargar Anaconda

Veamos como descargar Anaconda a nuestro disco y obtener esta suite científica de Python

Nos dirigimos a la Home de Anaconda e iremos a la sección de Download[1] (descargas)

Elegimos nuestra plataforma: Windows, Mac o Linux

> Atención: Elegir la versión de Python 3.7 (y no la de 2.7) y seleccionar el instalador Gráfico (Graphical Installer)

Con esto guardaremos en nuestro disco duro unos 460MB (según sistema operativo) y obtendremos un archivo con el nombre similar a Anaconda3-7.1.10-MacOSX-x86_64.pkg

2. Instalar Anaconda

En este paso instalaremos la app en nuestro sistema. (Deberá tener permisos de Administrador si instala para todos los usuarios).

Ejecutamos el archivo que descargamos haciendo doble click.

Se abrirá un Típico Wizard de instalación.

Seguiremos los pasos, podemos seleccionar instalación sólo para nuestro usuario, seleccionar la ruta en disco donde instalaremos y listo.

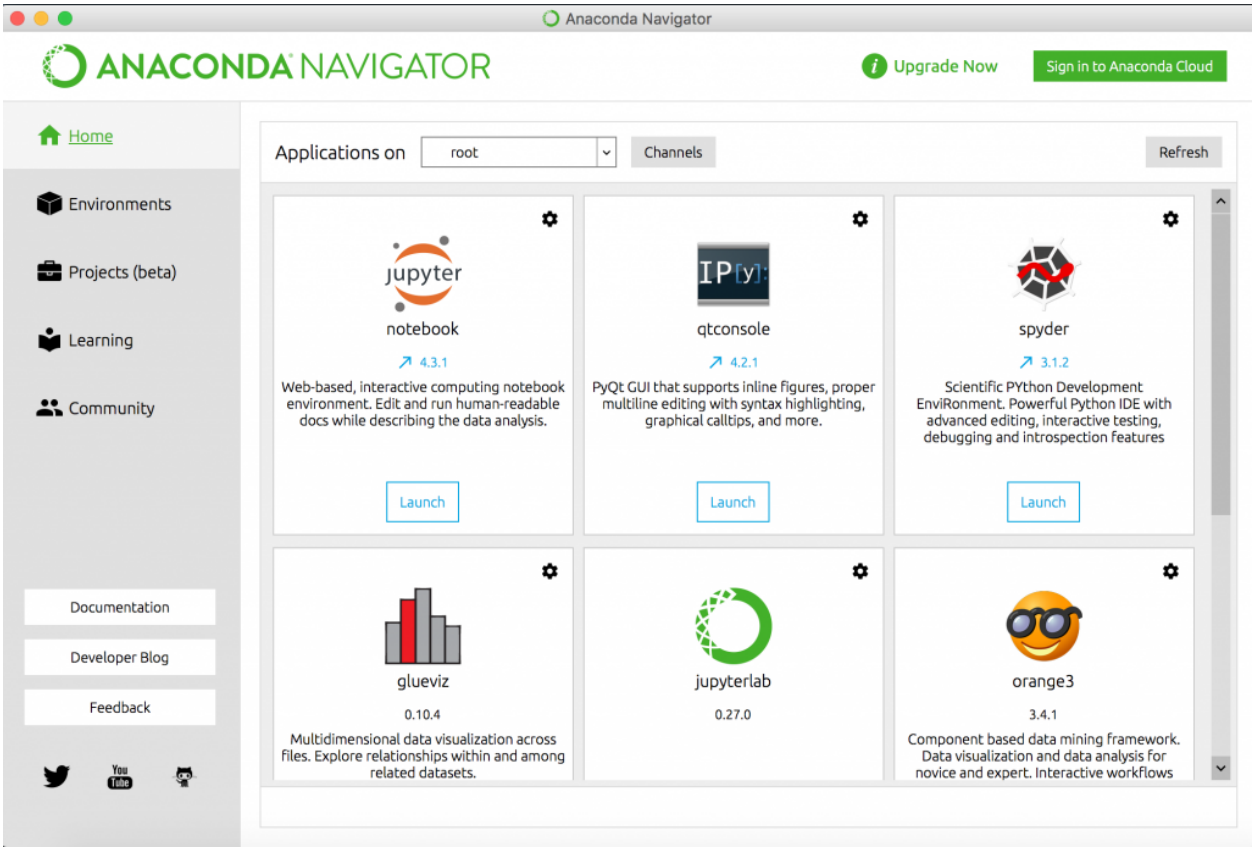
Al instalarse el tamaño total podrá superar 1Gb en disco.

3. Iniciar y Actualizar Anaconda

En este paso comprobaremos que se haya instalado correctamente y verificaremos tener la versión más reciente.

[1] https://www.anaconda.com/products/individual#download

Anaconda viene con una suite de herramientas gráficas llamada Anaconda Navigator. Iniciemos la aplicación y veremos una pantalla como esta:



Entre otros íconos vemos que podemos lanzar las Jupyter Notebooks!.

Para comprobar la instalación abrimos una Terminal de Mac/Linux/Ubuntu o la Línea de Comandos de Windows.

Escribimos

```
$ conda -V
```

y obtenemos la versión

```
conda 4.3.30
```

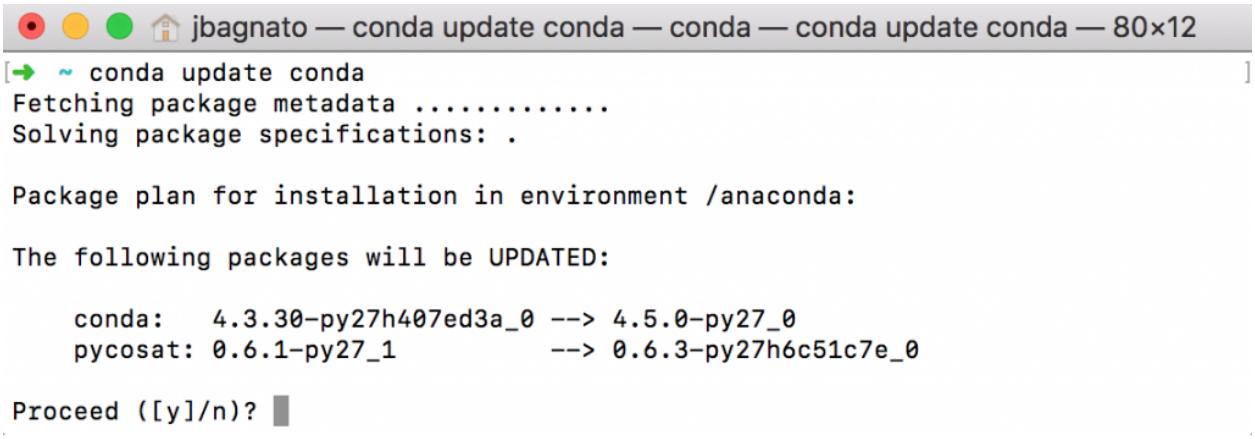
luego tipeamos

```
$ python -V
```

y verificamos la versión de Python de nuestro sistema.

Para asegurarnos de tener la versión más reciente de la suite ejecutaremos

```
$ conda update conda
```



debemos poner 'y' para confirmar y se descargarán. Luego ejecutamos

```
$ conda update anaconda
```

Para confirmar que todo funciona bien, crearemos un archivo de texto para escribir un breve script de python. Nombra al archivo versiones.py y su contenido será:

```
# scipy
import scipy
print('scipy: %s' % scipy.__version__)
# numpy
import numpy
print('numpy: %s' % numpy.__version__)
# matplotlib
import matplotlib
print('matplotlib: %s' % matplotlib.__version__)
# pandas
import pandas
print('pandas: %s' % pandas.__version__)
# statsmodels
import statsmodels
print('statsmodels: %s' % statsmodels.__version__)
# scikit-learn
import sklearn
print('sklearn: %s' % sklearn.__version__)
```

En la linea de comandos, en el mismo directorio donde está el archivo escribiremos:

```
$ python versiones.py
```

y deberemos ver una salida similar a esta:

```
scipy: 0.18.1
numpy: 1.12.1
matplotlib: 1.5.3
pandas: 0.19.2
statsmodels: 0.8.0
sklearn: 0.18.1
```

4. Actualizar libreria scikit-learn

En este paso actualizaremos la librería más usada para Machine Learning en python llamada Scikit-Learn

En la Terminal escribiremos

```
$ conda update scikit-learn
```

Deberemos confirmar la actualización poniendo ‘y’ en la terminal.

Podemos volver a verificar que todo es correcto ejecutando

```
$ python versiones.py
```

5. Instalar librerías para Deep Learning

En este paso instalaremos las librerías utilizadas para Aprendizaje profundo. Específicamente serán keras y la famosa y querida Tensorflow de Google.

Para ello ejecutaremos en nuestra línea de comandos

```
$ conda install -c conda-forge tensorflow
```

```
$ pip install keras
```

Y crearemos un nuevo script para probar que se instalaron correctamente. Le llamaremos versiones_deep.py y tendrá las siguientes lineas:

```
# tensorflow
import tensorflow
print('tensorflow: %s' % tensorflow.__version__)
# keras
import keras
print('keras: %s' % keras.__version__)
```

Ejecutamos en línea de comandos

```
$ python versiones_deep.py
```

en la terminal veremos la salida:

```
tensorflow: 1.0.1
Using TensorFlow backend.
keras: 2.0.2
```

Ya tenemos nuestro ambiente de desarrollo preparado para el combate.

Resumen

Para nuestra carrera en Machine Learning y el perfeccionamiento como Data Scientist necesitamos un buen entorno en el que programar y cacharrear -lease, probar cosas y divertirse-. Para ello contamos con la suite de herramientas gratuitas de Anaconda que nos ofrece un entorno amable y sencillo en el que crear nuestras máquinas en código Python.

Otros artículos de interés (en inglés)

Usar Anaconda Navigator[2]
Instalar Pip[3]
Instalar Tensorflow[4]
Instalación de Keras[5]

[2] https://docs.anaconda.com/anaconda/navigator/
[3] https://recursospython.com/guias-y-manuales/instalacion-y-utilizacion-de-pip-en-windows-linux-y-os-x/
[4] https://www.tensorflow.org/install/
[5] https://keras.io/#installation

Análisis Exploratorio de Datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es el EDA?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

EDA deconstruido

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué sacamos del EDA?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Técnicas para EDA

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Un EDA de pocos minutos con Pandas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Más cosas! (que se suelen hacer):

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

BONUS track: Notebook sobre manipulación de datos con Pandas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Más Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Regresión Lineal con Python

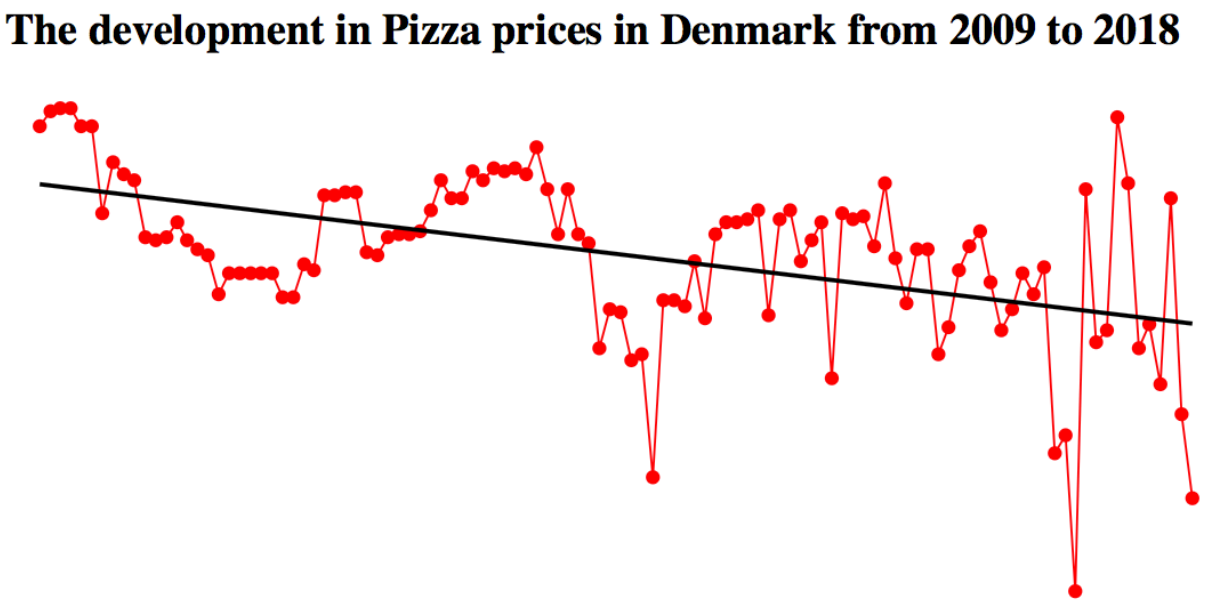
¿Qué es la regresión lineal?

La regresión lineal[6] es un algoritmo[7] de aprendizaje supervisado[8] que se utiliza en Machine Learning y en estadística. En su versión más sencilla, lo que haremos es "dibujar una recta" que *nos indicará la tendencia* de un conjunto de datos continuos (si fueran discretos, utilizaríamos Regresión Logística[9]). En estadísticas, *regresión lineal es una aproximación para modelar la relación entre una variable escalar dependiente "y" y una o mas variables explicativas nombradas con "X".*

Recordemos rápidamente la fórmula de la recta:

$$Y = mX + b$$

Donde Y es el resultado, X es la variable, m la pendiente (o coeficiente) de la recta y b la constante o también conocida como el "punto de corte con el eje Y" en la gráfica (cuando X=0)



Aqui vemos un ejemplo donde vemos datos recabados sobre los precios de las pizzas en Dinamarca (los puntos en rojo) y la linea negra es la tendencia. Esa es la línea de regresión que buscamos que el algoritmo aprenda y calcule sólo.

[6] https://es.wikipedia.org/wiki/Regresi%C3%B3n_lineal
[7] http://www.aprendemachinelearning.com/principales-algoritmos-usados-en-machine-learning/
[8] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/#supervisado
[9] http://www.aprendemachinelearning.com/regresion-logistica-con-python-paso-a-paso/

¿Cómo funciona el algoritmo de regresión lineal en Machine Learning?

Recordemos que los algoritmos de Machine Learning Supervisados[10], aprenden por sí mismos y -en este caso- a obtener automáticamente esa "recta" que buscamos con la tendencia de predicción. Para hacerlo se mide el error con respecto a los puntos de entrada y el valor "Y" de salida real. El algoritmo deberá minimizar el coste de una función de error cuadrático[11] y esos coeficientes corresponderán con la recta óptima. Hay diversos métodos para conseguir minimizar el coste. Lo más común es utilizar una versión vectorial y la llamada Ecuación Normal[12] que nos dará un resultado directo.

NOTA: cuando hablo de "recta" es en el caso particular de regresión lineal simple. Si hubiera más variables, hay que generalizar el término.

Un Ejercicio Práctico

En este ejemplo cargaremos un archivo .csv de entrada[13] obtenido por webscraping[14] que contiene diversas URLs a artículos sobre Machine Learning de algunos sitios muy importantes como Techcrunch[15] o KDnuggets[16] y como características de entrada -las columnas- tendremos:

- **Title**: titulo del artículo
- **url**: ruta al artículo
- **Word count**: la cantidad de palabras del artículo,
- **# of Links**: los enlaces externos que contiene,
- **# of comments**: cantidad de comentarios,
- **# Images video**: suma de imágenes (o videos),
- **Elapsed days**: la cantidad de días transcurridos (al momento de crear el archivo)
- **# Shares**: nuestra columna de salida que será la "cantidad de veces que se compartió el artículo".

A partir de las características de un artículo de machine learning intentaremos predecir, cuantas veces será compartido en Redes Sociales. Haremos una primer predicción de regresión lineal simple[17] -con una sola variable predictora- para poder graficar en 2 dimensiones (ejes X e Y) y luego un ejemplo de **regresión Lineal Múltiple**, en la que utilizaremos 3 dimensiones (X,Y,Z) y predicciones.

NOTA: el archivo .csv contiene mitad de datos reales, y otra mitad los generados de manera aleatoria, por lo que las predicciones que obtendremos no serán reales.

[10] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/#supervisado
[11] https://es.wikipedia.org/wiki/Error_cuadr%C3%A1tico_medio
[12] https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics)#Derivation_of_the_normal_equations
[13] http://www.aprendemachinelearning.com/articulos_ml/
[14] http://www.aprendemachinelearning.com/ejemplo-web-scraping-python-ibex35-bolsa-valores/
[15] https://techcrunch.com/tag/machine-learning/
[16] https://www.kdnuggets.com
[17] https://en.wikipedia.org/wiki/Simple_linear_regression

Requerimientos para hacer el Ejercicio

Para realizar este ejercicio, crearemos una Jupyter notebook[18] con código Python y la librería Scikit-Learn muy utilizada en Data Science. Recomendamos utilizar la suite de Anaconda[19].
Podrás descargar los archivos de entrada csv[20] o visualizar la notebook online[21].

Predecir cuántas veces será compartido un artículo de Machine Learning.

Regresión lineal simple en Python (con 1 variable)

Aqui vamos con nuestra notebook! Comencemos por importar las librerías que utilizaremos:

```
# Imports necesarios
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

Leemos el archivo csv y lo cargamos como un dataset de Pandas. Y vemos su tamaño.

```
#cargamos los datos de entrada
data = pd.read_csv("articulos_ml.csv")
#veamos cuantas dimensiones y registros contiene
data.shape
```

Nos devuelve (161,8) Veamos esas primeras filas:

[18] http://data-speaks.luca-d3.com/2018/03/python-para-todos-2-jupyternotebook.html
[19] https://www.anaconda.com/download/
[20] http://www.aprendemachinelearning.com/articulos_ml/
[21] https://github.com/jbagnato/machine-learning/blob/master/Ejercicio_Regresion_Lineal.ipynb

```
#son 161 registros con 8 columnas. Veamos los primeros registros
data.head()
```

	Title	url	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	1888	1	2.0	2	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	1742	9	NaN	9	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	962	6	0.0	1	10	42000
3	Dbrain and the Blockchain of Artificial Intell...	NaN	1221	3	NaN	2	68	200000
4	Nasa finds entire solar system filled with eig...	NaN	2039	1	104.0	4	131	200000

[22]

Se ven algunos campos con valores NaN (nulos) por ejemplo algunas urls o en comentarios. Veamos algunas estadísticas básicas de nuestros datos de entrada:

```
# Ahora veamos algunas estadísticas de nuestros datos
data.describe()
```

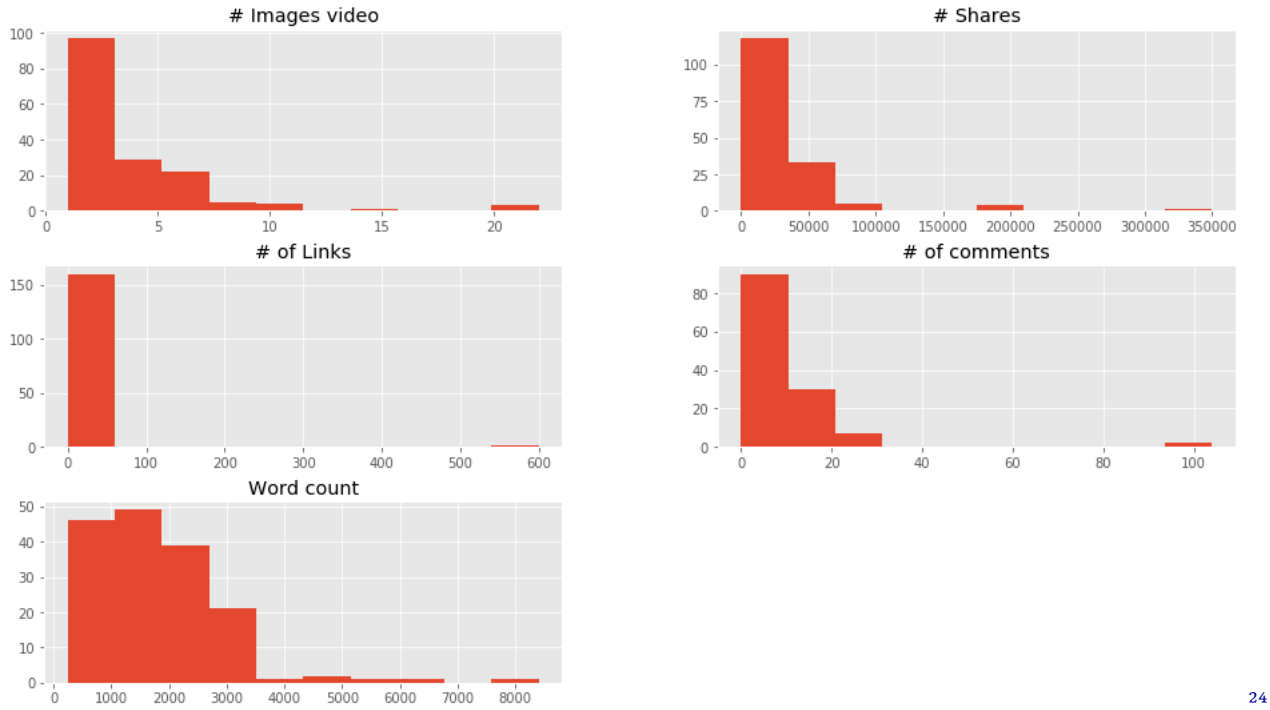
	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000
mean	1808.260870	9.739130	8.782946	3.670807	98.124224	27948.347826
std	1141.919385	47.271625	13.142822	3.418290	114.337535	43408.006839
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	990.000000	3.000000	2.000000	1.000000	31.000000	2800.000000
50%	1674.000000	5.000000	6.000000	3.000000	62.000000	16458.000000
75%	2369.000000	7.000000	12.000000	5.000000	124.000000	35691.000000
max	8401.000000	600.000000	104.000000	22.000000	1002.000000	350000.000000

[23]

Aqui vemos que la media de palabras en los artículos es de 1808. El artículo más corto tiene 250 palabras y el más extenso 8401. Intentaremos ver con nuestra relación lineal, si hay una correlación entre la cantidad de palabras del texto y la cantidad de Shares obtenidos. Hacemos una visualización en general de los datos de entrada:

```
# Visualizamos rápidamente las caraterísticas de entrada
data.drop(['Title','url', 'Elapsed days'],1).hist()
plt.show()
```

[22] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/reg_lineal_filas_iniciales.png
[23] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/reg_lineal_stats_base.png



[24]

En estas gráficas vemos entre qué valores se concentran la mayoría de registros. Vamos a filtrar los datos de cantidad de palabras para quedarnos con los registros con menos de 3500 palabras y también con los que tengan Cantidad de compartidos menor a 80000. Lo gratificaremos pintando en azul los puntos con menos de 1808 palabras (la media) y en naranja los que tengan más.

```
# Vamos a RECORTAR los datos en la zona donde se concentran más los puntos
# esto es en el eje X: entre 0 y 3.500
# y en el eje Y: entre 0 y 80.000
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <= 80000)]

colores=['orange','blue']
tamanios=[30,60]

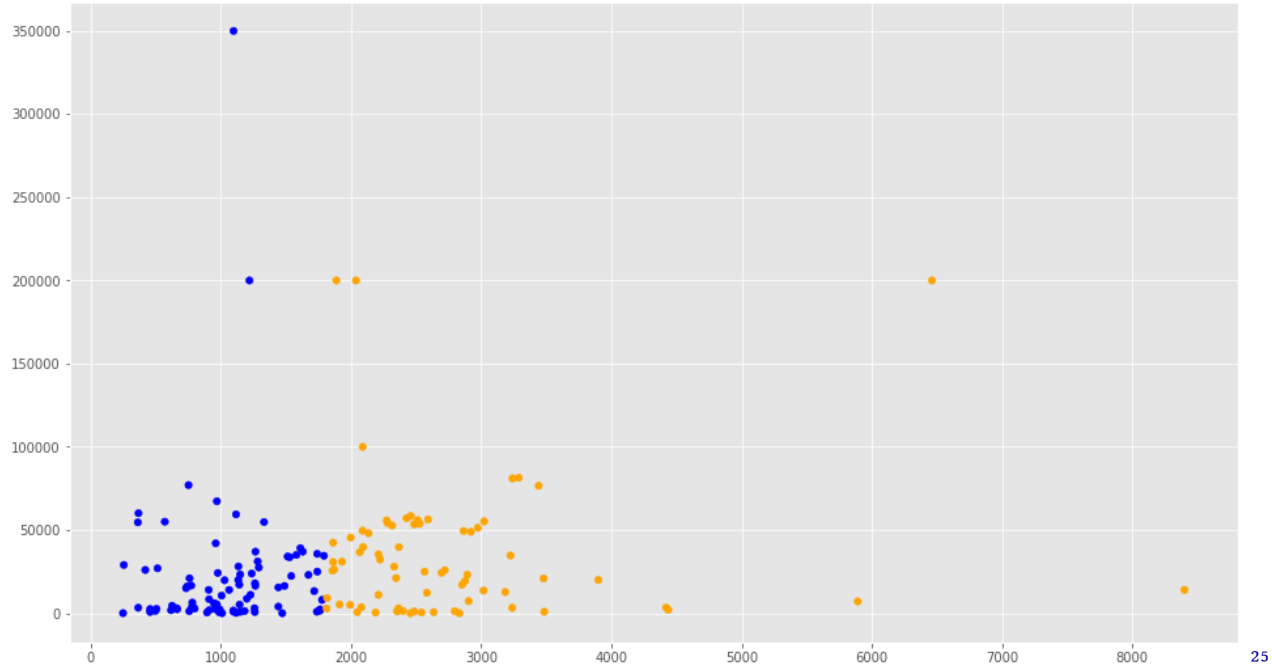
f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values

# Vamos a pintar en colores los puntos por debajo y por encima de la media de Cantid\
ad de Palabras
asignar=[]
for index, row in filtered_data.iterrows():
    if(row['Word count']>1808):
        asignar.append(colores[0])
    else:
```

[24] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/reg_lin_visualiza_entradas.png

```
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```



[25]

Regresión Lineal con Python y SKLearn

Vamos a crear nuestros datos de entrada por el momento sólo Word Count y como etiquetas los # Shares. Creamos el objeto LinearRegression y lo hacemos "encajar" (entrenar) con el método fit(). Finalmente imprimimos los coeficientes y puntajes obtenidos.

```
# Asignamos nuestra variable de entrada X para entrenamiento y las etiquetas Y.
dataX =filtered_data[["Word count"]]
X_train = np.array(dataX)
y_train = filtered_data['# Shares'].values

# Creamos el objeto de Regresión Linear
regr = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regr.fit(X_train, y_train)
```

[25] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/reg_lineal_grafica_pal_vs_shares.png

```

# Hacemos las predicciones que en definitiva una línea (en este caso, al ser 2D)
y_pred = regr.predict(X_train)

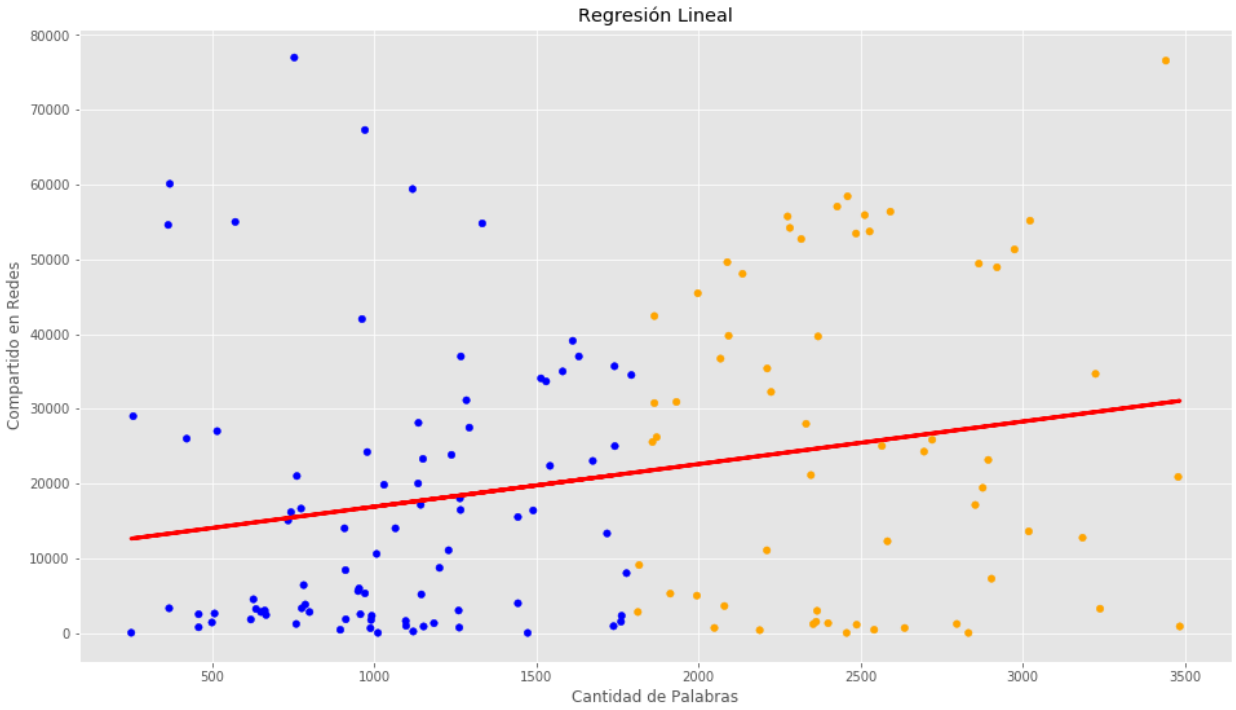
# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients: \n', regr.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term: \n', regr.intercept_)
# Error Cuadrado Medio
print("Mean squared error: %.2f" % mean_squared_error(y_train, y_pred))
# Puntaje de Varianza. El mejor puntaje es un 1.0
print('Variance score: %.2f' % r2_score(y_train, y_pred))
```

```
Coefficients: [5.69765366]
Independent term: 11200.303223074163
Mean squared error: 372888728.34
Variance score: 0.06
```

De la ecuación de la recta y = mX + b nuestra pendiente “m” es el coeficiente 5,69 y el término independiente “b” es 11200. Tenemos un Error Cuadrático medio enorme... por lo que en realidad este modelo no será muy bueno prediciendo ;) Pero estamos aprendiendo a usarlo, que es lo que nos importa ahora :) Esto también se ve reflejado en el puntaje de Varianza que debería ser cercano a 1.0.

Visualicemos la Recta

Veamos la recta que obtuvimos:



[26]

Predicción en regresión lineal simple

Vamos a intentar probar nuestro algoritmo, suponiendo que quisiéramos predecir cuántos "compartir" obtendrá un articulo sobre ML de 2000 palabras.

```
#Vamos a comprobar:
# Quiero predecir cuántos "Shares" voy a obtener por un artículo con 2.000 palabras,
# según nuestro modelo, hacemos:
y_Dosmil = regr.predict([[2000]])
print(int(y_Dosmil))
```

Nos devuelve una predicción de 22595 "Shares" para un artículo de 2000 palabras.

[26] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/reg_lineal_recta_1_variable.png

Regresión Lineal Múltiple en Python

(o “Regresión con Múltiples Variables”)

Vamos a extender el ejercicio utilizando más de una variable de entrada para el modelo. Esto le da **mayor poder** al algoritmo de Machine Learning, pues de esta manera podremos obtener predicciones más complejas. Nuestra “ecuación de la Recta”, ahora pasa a ser:

$$Y = b + m1\ X1 + m2\ X2 + \ldots + m(n)\ X(n)$$

(y deja de ser una recta) **En nuestro caso, utilizaremos 2 “variables predictivas” para poder graficar en 3D**, pero recordar que *para mejores predicciones podemos utilizar más de 2 entradas* y prescindir del gráfico. Nuestra primer variable seguirá siendo la **cantidad de palabras** y la segunda variable la crearemos artificialmente y será **la suma de 3 columnas de entrada**: la cantidad de enlaces, comentarios y cantidad de imágenes. Vamos a programar!

```
#Vamos a intentar mejorar el Modelo, con una dimensión más:
# Para poder graficar en 3D, haremos una variable nueva que será la suma de los enla\
ces, comentarios e imágenes
suma = (filtered_data["# of Links"] + filtered_data['# of comments'].fillna(0) + fil\
tered_data['# Images video'])

dataX2 =  pd.DataFrame()
dataX2["Word count"] = filtered_data["Word count"]
dataX2["suma"] = suma
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

Ya tenemos nuestras 2 variables de entrada en XY_train y **nuestra variable de salida pasa de ser “Y” a ser el eje “Z”**. Creamos un nuevo objeto de Regresión lineal con SKLearn pero esta vez tendrá las dos dimensiones que entrenar: las que contiene XY_train. Al igual que antes, imprimimos los coeficientes y puntajes obtenidos:

```
# Creamos un nuevo objeto de Regresión Lineal
regr2 = linear_model.LinearRegression()

# Entrenamos el modelo, esta vez, con 2 dimensiones
# obtendremos 2 coeficientes, para graficar un plano
regr2.fit(XY_train, z_train)

# Hacemos la predicción con la que tendremos puntos sobre el plano hallado
z_pred = regr2.predict(XY_train)

# Los coeficientes
print('Coefficients: \n', regr2.coef_)
# Error cuadrático medio
print("Mean squared error: %.2f" % mean_squared_error(z_train, z_pred))
# Evaluamos el puntaje de varianza (siendo 1.0 el mejor posible)
print('Variance score: %.2f' % r2_score(z_train, z_pred))
```

```
Coefficients: [ 6.63216324 -483.40753769]
Mean squared error: 352122816.48
Variance score: 0.11
```

Como vemos, obtenemos 2 coeficientes (cada uno correspondiente a nuestras 2 variables predictivas), pues ahora lo que graficamos no será una linea si no, **un plano en 3 Dimensiones**. El error obtenido sigue siendo grande, aunque algo mejor que el anterior y el puntaje de Varianza mejora casi el doble del anterior (aunque sigue siendo muy malo, muy lejos del 1).

Visualizar un plano en 3 Dimensiones en Python

Graficaremos nuestros puntos de las características de entrada en color azul y los puntos proyectados en el plano en rojo. Recordemos que en esta gráfica, el eje Z corresponde a la “altura” y representa la cantidad de Shares que obtendremos.

```
fig = plt.figure()
ax = Axes3D(fig)

# Creamos una malla, sobre la cual graficaremos el plano
xx, yy = np.meshgrid(np.linspace(0, 3500, num=10), np.linspace(0, 60, num=10))

# calculamos los valores del plano para los puntos x e y
nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)

# calculamos los correspondientes valores para z. Debemos sumar el punto de intercep\
ción
z = (nuevoX + nuevoY + regr2.intercept_)

# Graficamos el plano
ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')

# Graficamos en azul los puntos en 3D
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',s=30)

# Graficamos en rojo, los puntos que
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red',s=40)

# con esto situamos la "camara" con la que visualizamos
ax.view_init(elev=30., azim=65)

ax.set_xlabel('Cantidad de Palabras')
ax.set_ylabel('Cantidad de Enlaces,Comentarios e Imagenes')
ax.set_zlabel('Compartido en Redes')
ax.set_title('Regresión Lineal con Múltiples Variables')
```

[27]

Podemos rotar el gráfico para apreciar el plano desde diversos ángulos modificando el valor del parámetro **azim** en *view_init con números de 0 a 360.*

Predicción con el modelo de Mútiples Variables

Veamos ahora, que predicción tendremos para un artículo de 2000 palabras, con 10 enlaces, 4 comentarios y 6 imágenes.

```
# Si quiero predecir cuántos "Shares" voy a obtener por un artículo con:
# 2000 palabras y con enlaces: 10, comentarios: 4, imagenes: 6

z_Dosmil = regr2.predict([[2000, 10+4+6]])
print(int(z_Dosmil))
```

Esta predicción nos da 20518 y probablemente sea un poco mejor que nuestra predicción anterior con 1 variables.

Resumen

Hemos visto cómo utilizar SKLearn en Python para crear modelos de Regresión Lineal con 1 o múltiples variables. En nuestro ejercicio no tuvimos una gran confianza en las predicciónes. Por

[27] http://www.aprendemachinelearning.com/wp-content/uploads/2018/05/regresion_lineal_3d_plano.png

ejemplo en nuestro primer modelo, con 2000 palabras nos predice que podemos tener 22595 pero el margen de error haciendo raíz del error cuartico medio es más menos 19310. Es decir que escribiendo un artículo de 2000 palabras lo mismo tenemos 3285 Shares que 41905. En este caso usamos este modelo para aprender a usarlo y habrá que ver en otros casos en los que sí nos brinde predicciones acertadas. Para mejorar nuestro modelo, deberíamos utilizar más dimensiones y encontrar datos de entrada mejores.
Atención: también es posible, que no exista ninguna relación fuerte entre nuestras variables de entrada y el éxito en Shares del artículo...

Recursos y enlaces

- Descarga la Jupyter Notebook[28] y el archivo de entrada csv[29]
- ó puedes visualizar online[30]
- o ver y descargar desde mi cuenta github[31]

Otros enlaces con Artículos sobre Regresión Lineal (en Inglés)

- Introduction to Linear Regression using python[32]
- Linear Regression using Python SkLearn[33]
- Linear Regression Detailed View[34]
- How do you solve a linear regression problem in python[35]
- Python tutorial on LinearRegression with Batch Gradient Descent[36]

[28] http://www.aprendemachinelearning.com/ejercicio_regresion_lineal/
[29] http://www.aprendemachinelearning.com/articulos_ml/
[30] https://github.com/jbagnato/machine-learning/blob/master/Ejercicio_Regresion_Lineal.ipynb
[31] https://github.com/jbagnato/machine-learning
[32] https://aktechthoughts.wordpress.com/2018/03/26/introduction-to-linear-regression-using-python/
[33] https://dzone.com/articles/linear-regression-using-python-scikit-learn
[34] https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86
[35] http://lineardata.net/how-do-you-solve-a-linear-regression-machine-learning-problem-in-python/
[36] http://ozzieliu.com/2016/02/09/gradient-descent-tutorial/

Regresión Logística

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Introducción

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejercicio de Regresión Logística en Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Regresión Logística con SKLearn:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Identificar Sistema Operativo de los usuarios

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Visualización de Datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos el Modelo de Regresión Logística

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Validación de nuestro modelo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Reporte de Resultados del Modelo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Clasificación de nuevos valores

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Arbol de Decisión

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es un árbol de decisión?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué necesidad hay de usar el Algoritmo de Arbol?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona un árbol de decisión?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Indice Gini:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ganancia de información:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Arbol de Decisión con Scikit-Learn paso a paso

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Predicción del “Billboard 100”: ¿Qué artista llegará al número uno del ranking?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Obtención de los datos de entrada

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Análisis Exploratorio Inicial

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Balanceo de Datos: Pocos artistas llegan al número uno

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Preparamos los datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Mapeo de Datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Como quedan los top en relación a los datos mapeados

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Buscamos la profundidad para el árbol de decisión

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos el árbol y lo tuneamos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Visualización del árbol de decisión

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Análisis del árbol

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Predicción de Canciones al Billboard 100

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos y enlaces

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Qué es overfitting y cómo solucionarlo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Generalización del Conocimiento

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El problema de la Máquina al Generalizar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Overfitting en Machine Learning

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El equilibrio del Aprendizaje

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Prevenir el Sobreajuste de datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Datos desbalanceados

Agenda:

Problemas de clasificación con Clases desequilibradas

¿Cómo nos afectan los datos desbalanceados?

Métricas y Confusion Matrix

Vamos al Ejercicio con Python!

Análisis exploratorio

Estrategias para el manejo de Datos Desbalanceados:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Probando el Modelo sin estrategias

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Estrategia: Penalización para compensar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Estrategia: Subsampling en la clase mayoritaria

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Estrategia: Oversampling de la clase minoritaria

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Estrategia: Combinamos resampling con Smote-Tomek

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Estrategia: Ensamble de Modelos con Balanceo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resultados de las Estrategias

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Enlaces de interés

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Random Forest, el poder del Ensamble

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo surge Random Forest?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona Random Forest?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Por qué es aleatorio?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ventajas y Desventajas del uso de Random Forest

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Vamos al Código Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos el modelo y lo entrenamos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Los Hiperparámetros más importantes

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Evaluamos resultados

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparamos con el Baseline

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos y Adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Conjunto de Entrenamiento, Test y Validación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Un nuevo Mundo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Hágase el conjunto de Test

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Al Séptimo día Dios creo el Cross-Validation

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Técnicas de Validación Cruzada

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Cross-Validation: K-fold con 5 splits

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejemplo K-Folds en Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Más técnicas para Validación del modelo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Stratified K-Fold

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Leave P Out

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

ShuffleSplit

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Series Temporales: Atención al validar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

TimeSeriesSplit

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pero entonces? Cuando uso Cross-Validation?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Si ya estoy "conforme" y quiero llevar el modelo a un entorno de Producción?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recomendaciones finales:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos Adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

K-Means

K-Means es un algoritmo no supervisado[37] de Clustering[38]. Se utiliza cuando tenemos un montón de datos **sin etiquetar**. El objetivo de este algoritmo es el de encontrar "K" grupos (clusters) entre los datos crudos. En este artículo repasaremos sus conceptos básicos y veremos un ejemplo paso a paso en python.

Cómo funciona K-Means

El algoritmo trabaja iterativamente para asignar a cada "punto" (las filas de nuestro conjunto de entrada forman una coordenada) uno de los "K" grupos basado en sus características. Son agrupados en base a la similitud de sus features (las columnas). Como resultado de ejecutar el algoritmo tendremos:

- Los "centroids" de cada grupo que serán unas "coordenadas" de cada uno de los K conjuntos que se utilizarán para poder etiquetar nuevas muestras.
- Etiquetas para el conjunto de datos de entrenamiento. Cada etiqueta perteneciente a uno de los K grupos formados.

Los grupos se van definiendo de manera "orgánica", es decir que se va ajustando su posición en cada iteración del proceso, hasta que converge el algoritmo. Una vez hallados los centroids deberemos analizarlos para ver cuales son sus características únicas, frente a la de los otros grupos. Estos grupos son las etiquetas que genera el algoritmo.

Casos de Uso de K-Means

El algoritmo de Clustering K-means es uno de los más usados[39] para encontrar grupos ocultos, o sospechados en teoría sobre un conjunto de datos no etiquetado. Esto puede servir para confirmar -o desterrar- alguna teoría que teníamos asumida de nuestros datos. Y también puede ayudarnos a descubrir relaciones asombrosas entre conjuntos de datos, que de manera manual, no hubiéramos reconocido. Una vez que el algoritmo ha ejecutado y obtenido las etiquetas, será fácil clasificar nuevos valores o muestras entre los grupos obtenidos. Algunos casos de uso son:

- Segmentación por Comportamiento: relacionar el carrito de compras de un usuario, sus tiempos de acción e información del perfil.

[37] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/#no_supervisado
[38] http://www.aprendemachinelearning.com/principales-algoritmos-usados-en-machine-learning/#clustering
[39] http://www.aprendemachinelearning.com/principales-algoritmos-usados-en-machine-learning/

- Categorización de Inventario: agrupar productos por actividad en sus ventas
- Detectar anomalías o actividades sospechosas: según el comportamiento en una web reconocer un troll -o un bot- de un usuario normal

Datos de Entrada para K-Means

Las “features” o características que utilizaremos como entradas para aplicar el algoritmo k-means deberán ser de valores numéricos, continuos en lo posible. En caso de valores categóricos (por ej. Hombre/Mujer o Ciencia Ficción, Terror, Novela,etc) se puede intentar pasarlo a valor numérico, pero no es recomendable pues no hay una “distancia real” -como en el caso de géneros de película o libros-. Además es recomendable que los valores utilizados estén normalizados, manteniendo una misma escala. En algunos casos también funcionan mejor datos porcentuales en vez de absolutos. No conviene utilizar features que estén correlacionados o que sean escalares de otros.

El Algoritmo K-means

El algoritmo utiliza una proceso iterativo en el que se van ajustando los grupos para producir el resultado final. Para ejecutar el algoritmo deberemos pasar como entrada el conjunto de datos y un valor de K. El conjunto de datos serán las características o features para cada punto. Las posiciones iniciales de los K centroids serán asignadas de manera aleatoria de cualquier punto del conjunto de datos de entrada. Luego se itera en dos pasos:
1- Paso de Asignación de datos En este paso, cada “fila” de nuestro conjunto de datos se asigna al centroide más cercano basado en la distancia cuadrada Euclideana. Se utiliza la siguiente fórmula (donde dist() es la distancia Euclideana standard):

$$\underset{c_i \in C}{\arg\min}\ dist(c_i,\ x)^2$$

2-Paso de actualización de Centroid En este paso los centroid de cada grupo son recalculados. Esto se hace tomando una media de todos los puntos asignados en el paso anterior.

$$c_i = \frac{1}{|S_i|}\sum_{x_i \in S_i} x_i$$

El algoritmo itera entre estos pasos hasta cumplir un criterio de detención:

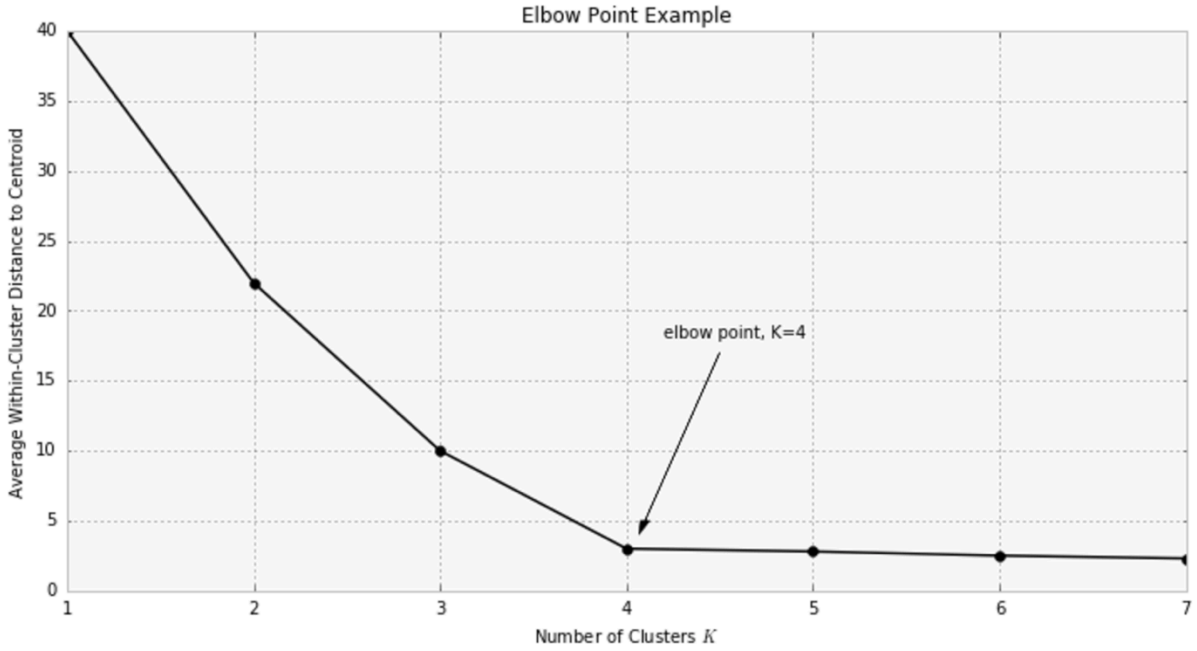
- si no hay cambios en los puntos asignados a los grupos,

- o si la suma de las distancias se minimiza,
- o se alcanza un número máximo de iteraciones.

El algoritmo converge a un resultado que puede ser el óptimo local, por lo que será conveniente volver a ejecutar más de una vez con puntos iniciales aleatorios para confirmar si hay una salida mejor.

Elegir el valor de K

Este algoritmo funciona pre-seleccionando un valor de K. Para encontrar el número de clusters en los datos, deberemos ejecutar el algoritmo para un rango de valores K, ver los resultados y comparar características de los grupos obtenidos. En general no hay un modo exacto de determinar el valor K, pero se puede estimar con aceptable precisión siguiendo la siguiente técnica: Una de las métricas usada para comparar resultados es la **distancia media entre los puntos de datos y su centroid**. Como el valor de la media diminuirá a medida de aumentemos el valor de K, deberemos utilizar la distancia media al centroide en función de K y entontrar el "punto codo", donde la tasa de descenso se "afila". Aquí vemos una gráfica a modo de ejemplo:



Ejemplo K-Means con Scikit-learn

Como ejemplo utilizaremos de entradas un conjunto de datos que obtuve de un proyecto propio, en el que se analizaban rasgos de la personalidad de usuarios de Twitter. He filtrado a 140 "famosos" del

mundo en diferentes areas: deporte, cantantes, actores, etc. Basado en una metodología de psicología conocida como "Ocean: The Big Five" tendemos como características de entrada:

- usuario (el nombre en Twitter)
- "op" = Openness to experience - grado de apertura mental a nuevas experiencias, curiosidad, arte
- "co" =Conscientiousness - grado de orden, prolijidad, organización
- "ex" = Extraversion - grado de timidez, solitario o participación ante el grupo social
- "ag" = Agreeableness - grado de empatía con los demás, temperamento
- "ne" = Neuroticism, - grado de neuroticismo, nervioso, irritabilidad, seguridad en sí mismo.
- Wordcount - Cantidad promedio de palabras usadas en sus tweets
- Categoria - Actividad laboral del usuario (actor, cantante, etc.)

Utilizaremos el algoritmo K-means para que agrupe estos usuarios -no por su actividad laboral- si no, por sus similitudes en la personalidad. Si bien tenemos 8 columnas de entrada, **sólo utilizaremos 3** en este ejemplo, de modo que podamos ver en un gráfico tridimensional -y sus proyecciones a 2D- los grupos resultantes. Pero para casos reales, podemos utilizar todas las dimensiones que necesitemos. Una de las hipótesis que podríamos tener es: "Todos los cantantes tendrán personalidad parecida" (y así con cada rubro laboral). Pues veremos si lo probamos, o por el contrario, los grupos no están relacionados necesariamente con la actividad de estas Celebridades.

Agrupar usuarios Twitter de acuerdo a su personalidad con K-means

Implementando K-means en Python con Sklearn

Comenzaremos importando las librerías que nos asistirán para ejecutar el algoritmo y graficar.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances_argmin_min

%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
```

Importamos el archivo csv[40] -para simplificar, suponemos que el archivo se encuentra en el mismo directorio que el notebook- y vemos los primeros 5 registros del archivo tabulados.

[40] http://www.aprendemachinelearning.com/wp-content/uploads/2018/03/analisis.csv

```
dataframe = pd.read_csv(r"analisis.csv")
dataframe.head()
```

	usuario	op	co	ex	ag	ne	wordcount	categoria
0	3gerardpique	34.297953	28.148819	41.948819	29.370315	9.841575	37.0945	7
1	aguerosergiokun	44.986842	20.525865	37.938947	24.279098	10.362406	78.7970	7
2	albertochicote	41.733854	13.745417	38.999896	34.645521	8.836979	49.2604	4
3	AlejandroSanz	40.377154	15.377462	52.337538	31.082154	5.032231	80.4538	2
4	alfredocasero1	36.664677	19.642258	48.530806	31.138871	7.305968	47.0645	4

También podemos ver una tabla de información estadística que nos provee Pandas dataframe:

```
dataframe.describe()
```

	op	co	ex	ag	ne	wordcount	categoria
count	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000
mean	44.414591	22.977135	40.764428	22.918528	8.000098	98.715484	4.050000
std	8.425723	5.816851	7.185246	7.657122	3.039248	44.714071	2.658839
min	30.020465	7.852756	18.693542	9.305985	1.030213	5.020800	1.000000
25%	38.206484	19.740299	36.095722	17.050993	6.086144	66.218475	2.000000
50%	44.507091	22.466718	41.457492	21.384554	7.839722	94.711400	3.500000
75%	49.365923	26.091606	45.197769	28.678867	9.758189	119.707925	7.000000
max	71.696129	49.637863	59.824844	40.583162	23.978462	217.183200	9.000000

El archivo contiene diferenciadas 9 categorías -actividades laborales- que son:

1. Actor/actriz
2. Cantante
3. Modelo
4. Tv, series
5. Radio
6. Tecnología
7. Deportes
8. Politica
9. Escritor

Para saber cuantos registros tenemos de cada uno hacemos:

```
print(dataframe.groupby('categoria').size())
```

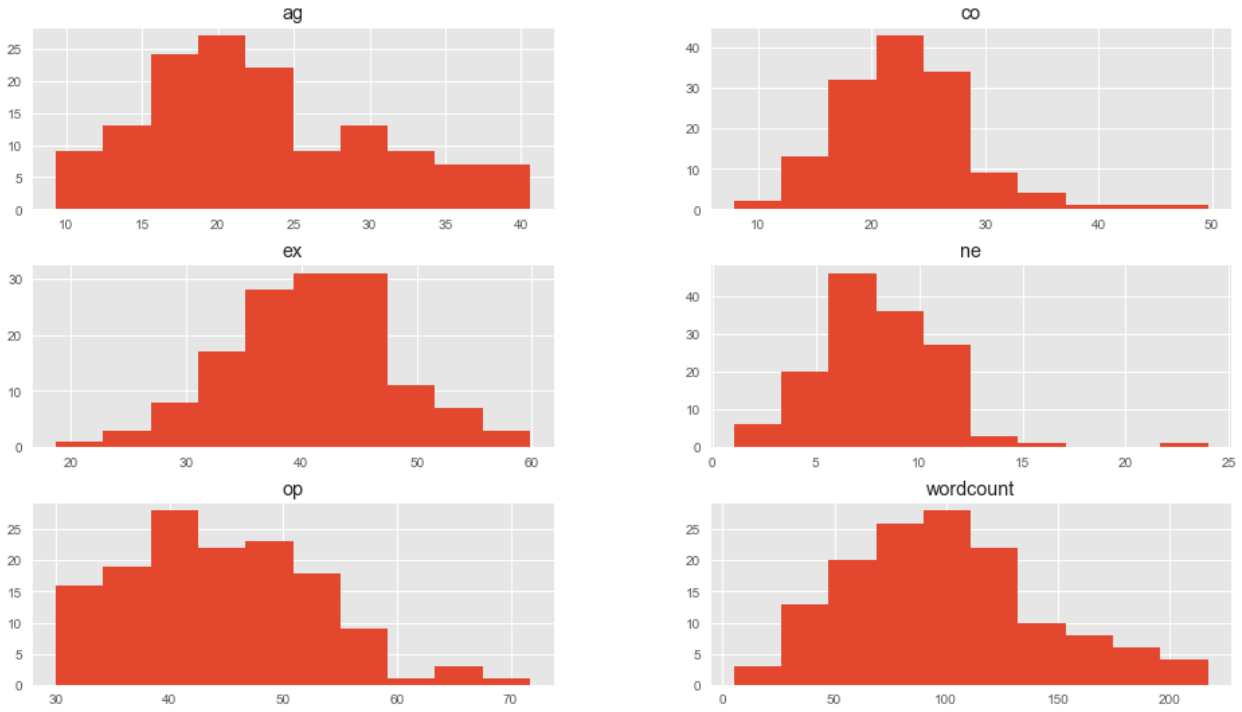
```
categoria
1    27
2    34
3     9
4    19
5     4
6     8
7    17
8    16
9     6
dtype: int64
```

Como vemos tenemos 34 cantantes, 27 actores, 17 deportistas, 16 políticos,etc.

Visualización de Datos

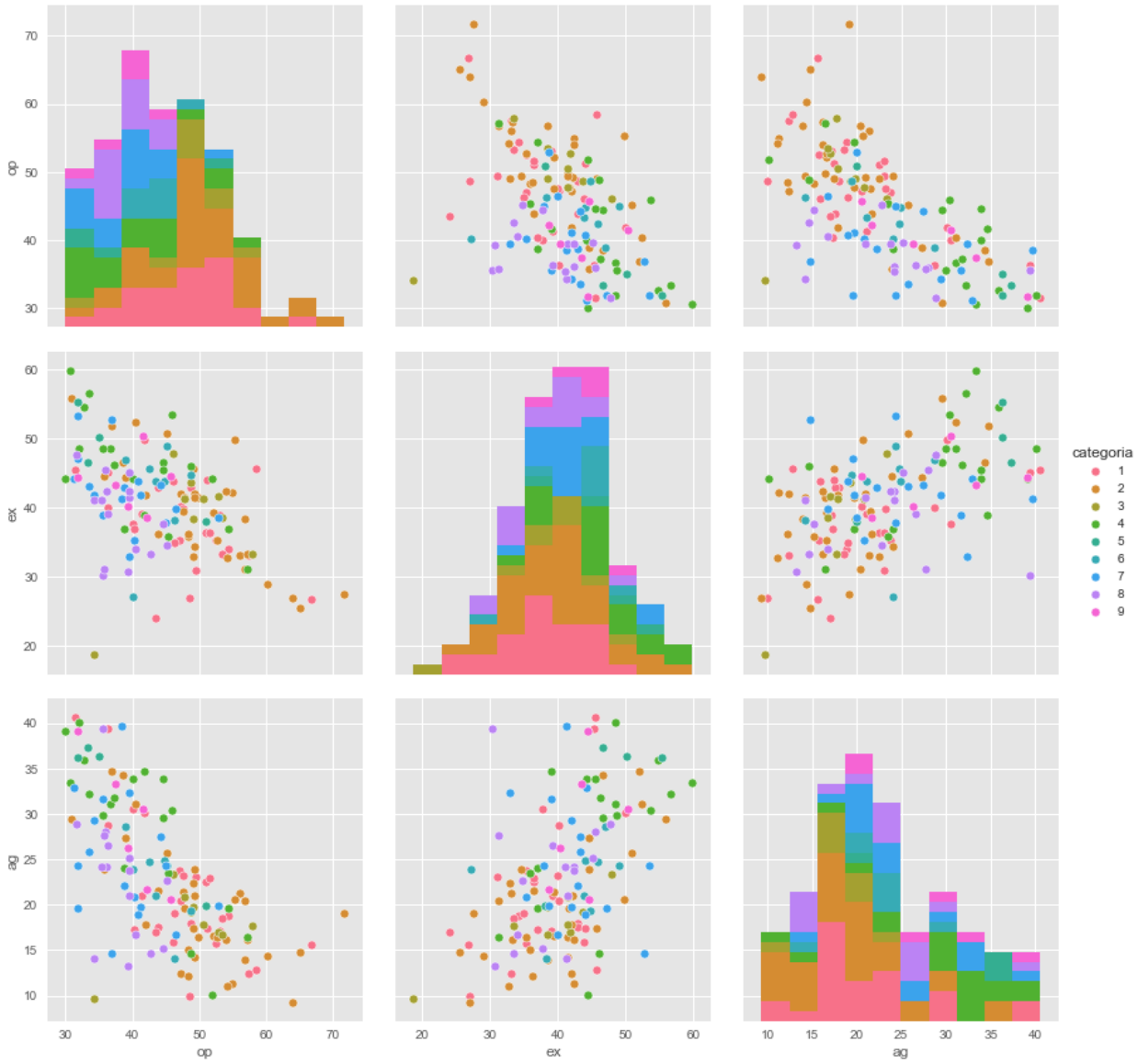
Veremos graficamente nuestros datos para tener una idea de la dispersión de los mismos:

```
dataframe.drop(['categoria'],1).hist()
plt.show()
```



En este caso seleccionamos 3 dimensiones: op, ex y ag y las cruzamos para ver si nos dan alguna pista de su agrupación y la relación con sus categorías.

```
sb.pairplot(dataframe.dropna(), hue='categoria',size=4,vars=["op","ex","ag"],kind='s\
catter')
```



Revisando la gráfica no pareciera que hay algún tipo de agrupación o correlación entre los usuarios y sus categorías.

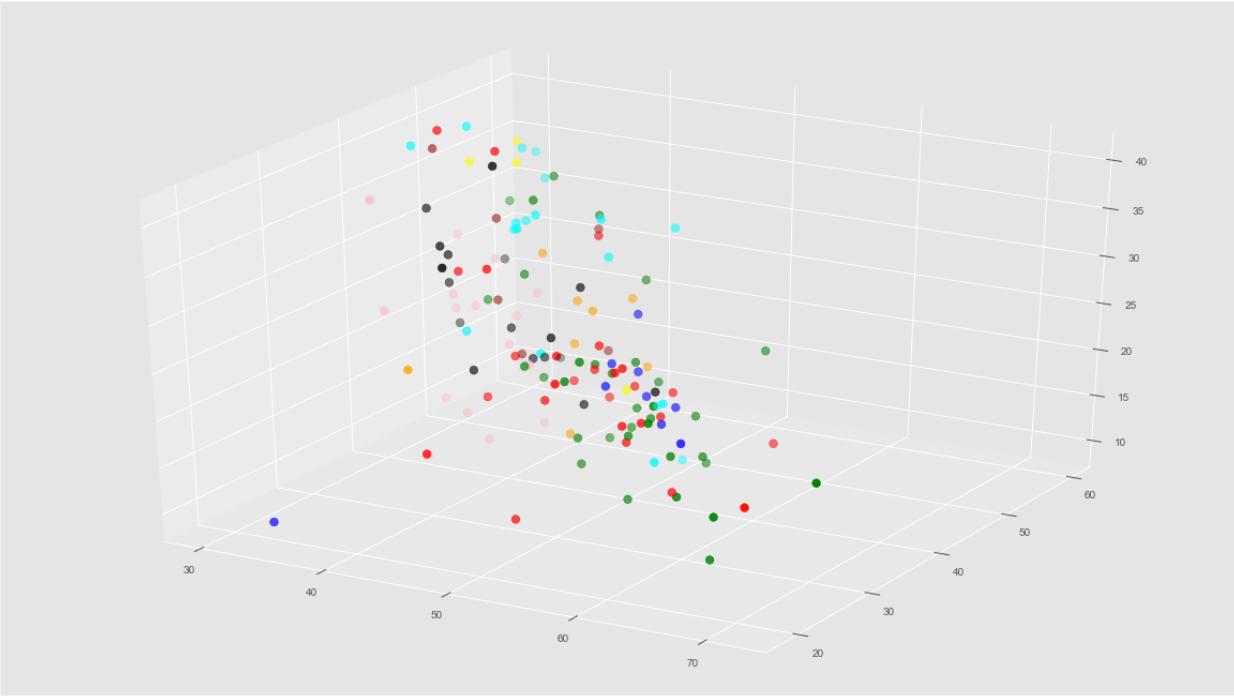
Definimos la entrada

Concretamos la estructura de datos que utilizaremos para alimentar el algoritmo. Como se ve, sólo cargamos las columnas op, ex y ag en nuestra variable X.

```
X = np.array(dataframe[["op","ex","ag"]])
y = np.array(dataframe['categoria'])
X.shape
```python

~~~
(140,3)
~~~

Ahora veremos una gráfica en 3D con 9 colores representando las categorías.
```python
fig = plt.figure()
ax = Axes3D(fig)
colores=['blue','red','green','blue','cyan','yellow','orange','black','pink','brown'\
,'purple']
asignar=[]
for row in y:
    asignar.append(colores[row])
ax.scatter(X[:, 0], X[:, 1], X[:, 2], c=asignar,s=60)
```

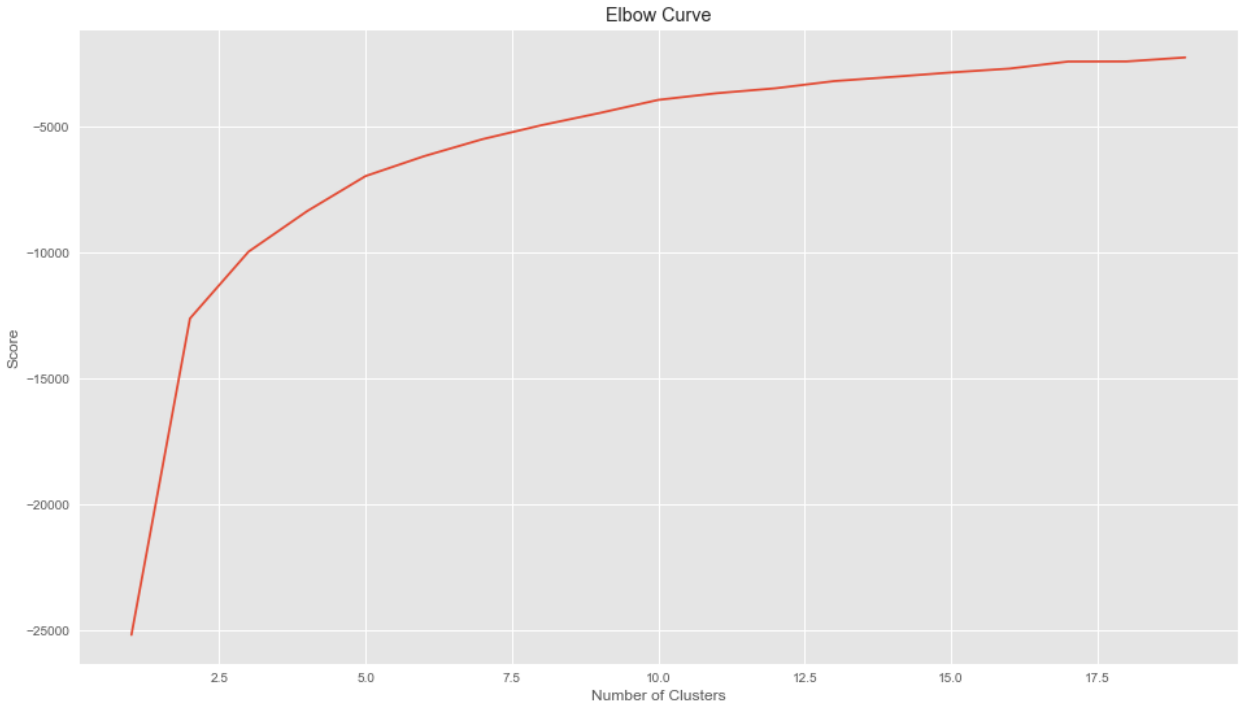


Veremos si con K-means, podemos "pintar" esta misma gráfica de otra manera, con clusters diferenciados.

Obtener el valor K

Vamos a hallar el valor de K haciendo una gráfica e intentando hallar el "punto de codo" que comentábamos antes. Este es nuestro resultado:

```
Nc = range(1, 20)
kmeans = [KMeans(n_clusters=i) for i in Nc]
kmeans
score = [kmeans[i].fit(X).score(X) for i in range(len(kmeans))]
score
plt.plot(Nc,score)
plt.xlabel('Number of Clusters')
plt.ylabel('Score')
plt.title('Elbow Curve')
plt.show()
```



Realmente la curva es bastante "suave". Considero a 5 como un buen número para K. Según vuestro criterio podría ser otro.

Ejecutamos K-Means

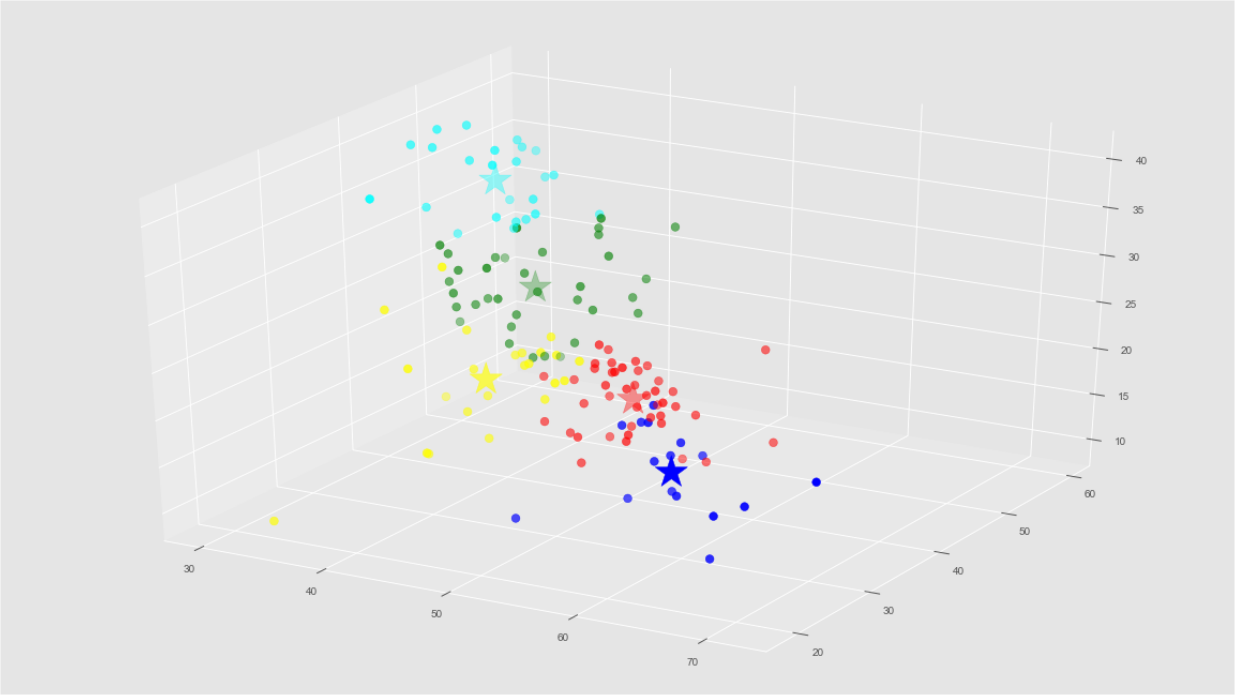
Ejecutamos el algoritmo para 5 clusters y obtenemos las etiquetas y los centroids.

```
kmeans = KMeans(n_clusters=5).fit(X)
centroids = kmeans.cluster_centers_
print(centroids)
```

```
[[ 49.80086386  40.8972579   17.48224326]
 [ 39.63830586  44.75784737  25.86962057]
 [ 58.58657531  31.02839375  15.6120435 ]
 [ 34.5303535   48.01261321  35.01749504]
 [ 42.302263    33.65449587  20.812626  ]]
```

Ahora veremos esto en una gráfica 3D con colores para los grupos y veremos si se diferencian: (las estrellas marcan el centro de cada cluster)

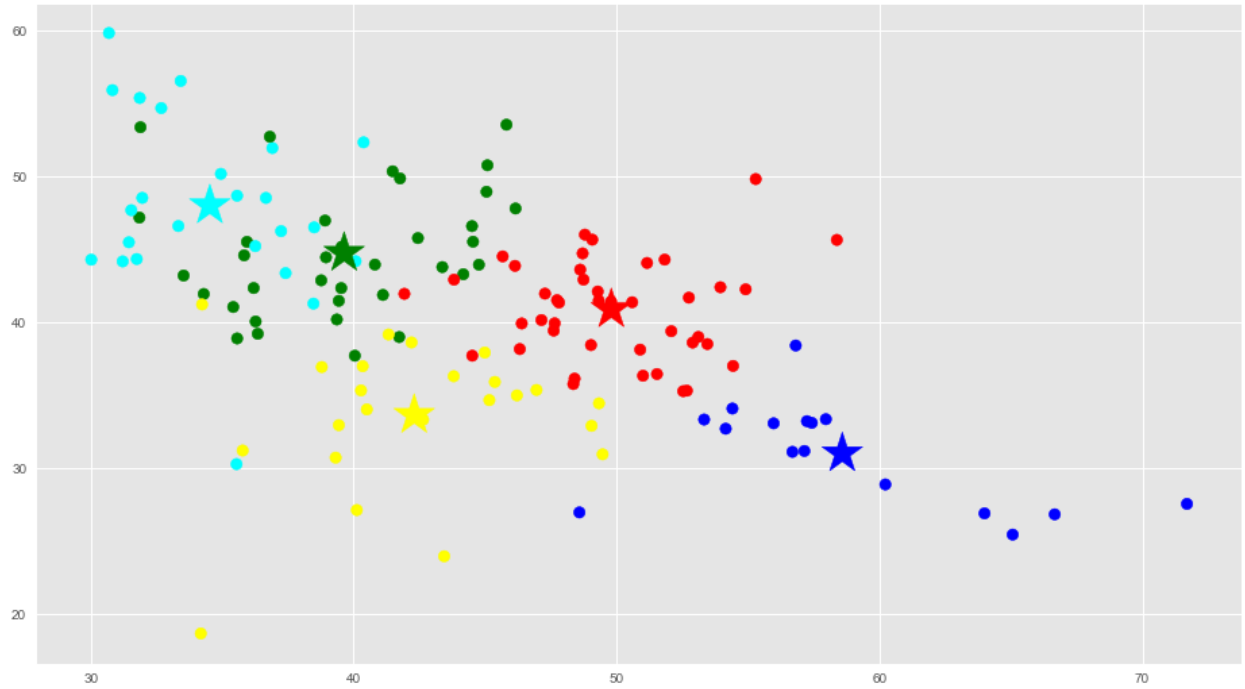
```
# Predicting the clusters
labels = kmeans.predict(X)
# Getting the cluster centers
C = kmeans.cluster_centers_
colores=['red','green','blue','cyan','yellow']
asignar=[]
for row in labels:
    asignar.append(colores[row])

fig = plt.figure()
ax = Axes3D(fig)
ax.scatter(X[:, 0], X[:, 1], X[:, 2], c=asignar,s=60)
ax.scatter(C[:, 0], C[:, 1], C[:, 2], marker='*', c=colores, s=1000)
```



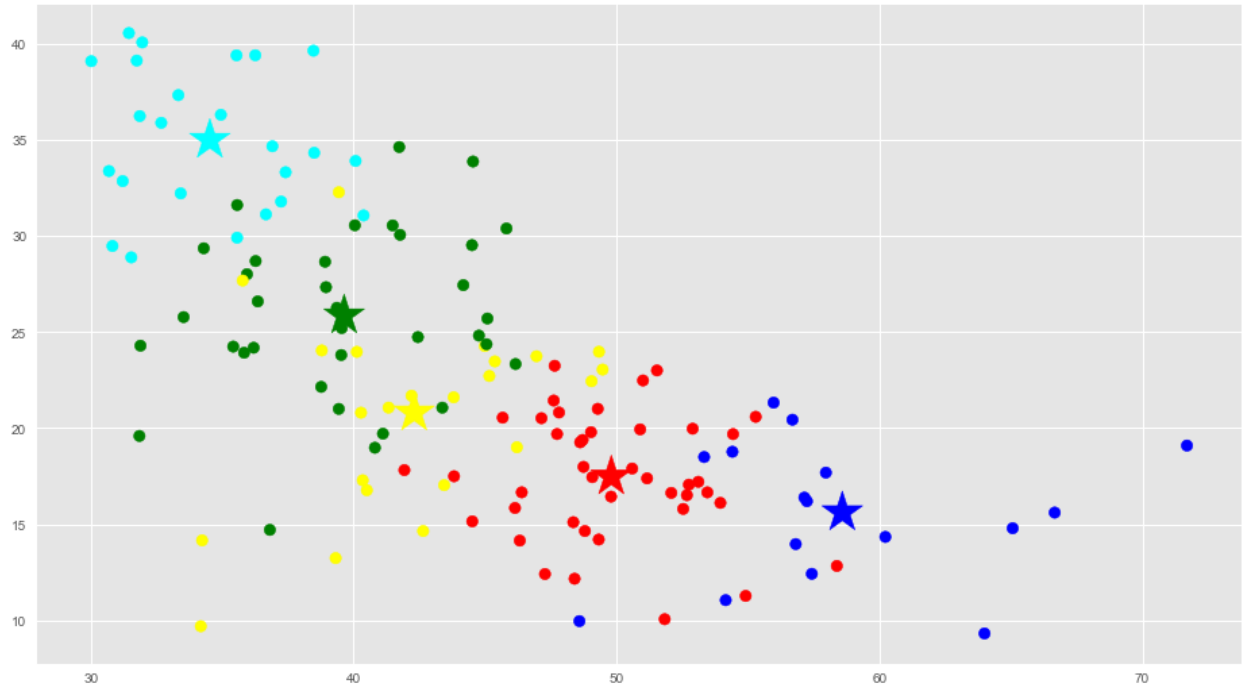
]

Aqui podemos ver que el Algoritmo de K-Means con K=5 ha agrupado a los 140 usuarios Twitter por su personalidad, teniendo en cuenta las 3 dimensiones que utilizamos: Openess, Extraversion y Agreeablenes. Pareciera que no hay necesariamente una relación en los grupos con sus actividades de Celebrity. Haremos 3 gráficas en 2 dimensiones con las proyecciones a partir de nuestra gráfica 3D para que nos ayude a visualizar los grupos y su clasificación:

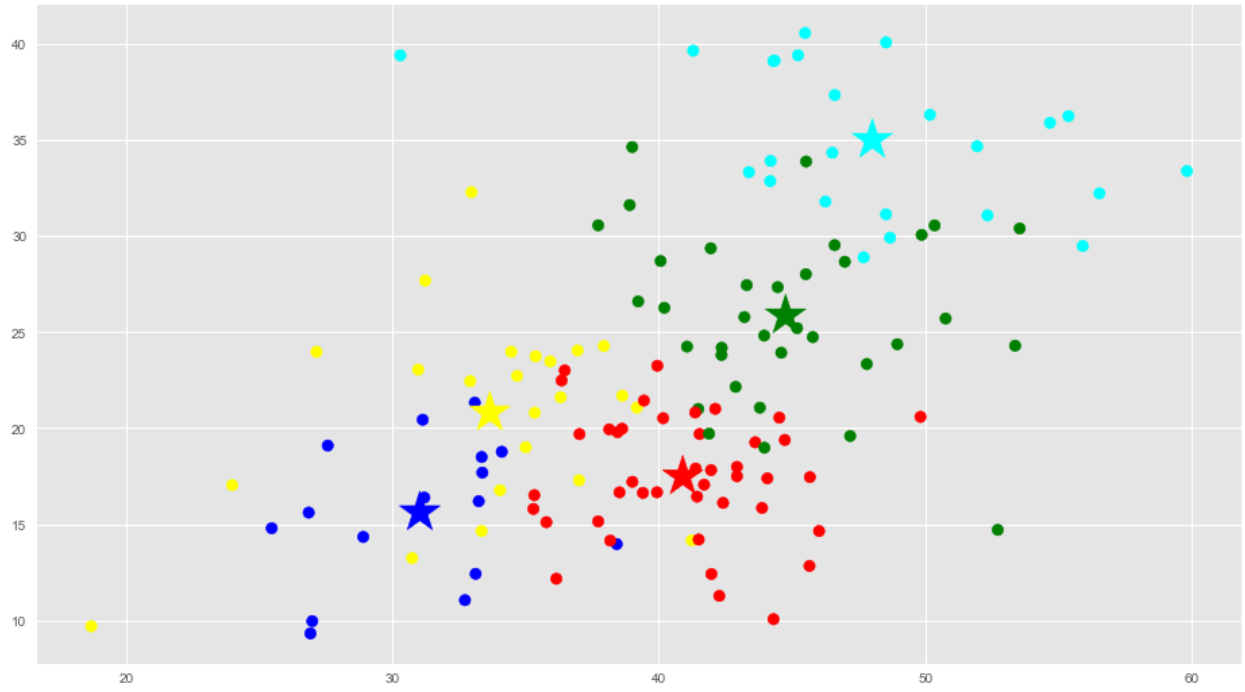
```
# Getting the values and plotting it
f1 = dataframe['op'].values
f2 = dataframe['ex'].values

plt.scatter(f1, f2, c=asignar, s=70)
plt.scatter(C[:, 0], C[:, 1], marker='*', c=colores, s=1000)
plt.show()
```




```
# Getting the values and plotting it
f1 = dataframe['op'].values
f2 = dataframe['ag'].values

plt.scatter(f1, f2, c=asignar, s=70)
plt.scatter(C[:, 0], C[:, 2], marker='*', c=colores, s=1000)
plt.show()
```




```
f1 = dataframe['ex'].values
f2 = dataframe['ag'].values

plt.scatter(f1, f2, c=asignar, s=70)
plt.scatter(C[:, 1], C[:, 2], marker='*', c=colores, s=1000)
plt.show()
```



En estas gráficas vemos que están bastante bien diferenciados los grupos. Podemos ver cada uno de los clusters cuantos usuarios tiene:

```
copy =  pd.DataFrame()
copy['usuario']=dataframe['usuario'].values
copy['categoria']=dataframe['categoria'].values
copy['label'] = labels;
cantidadGrupo =  pd.DataFrame()
cantidadGrupo['color']=colores
cantidadGrupo['cantidad']=copy.groupby('label').size()
cantidadGrupo
```

	color	cantidad
0	red	42
1	green	33
2	blue	16
3	cyan	27
4	yellow	22

Y podemos ver la diversidad en rubros laborales de cada uno. Por ejemplo en el grupo 0 (rojo), vemos que hay de todas las actividades laborales aunque predominan de actividad 1 y 2 correspondiente a Actores y Cantantes con 11 y 15 famosos.

```
group_referrer_index = copy['label'] ==0
group_referrals = copy[group_referrer_index]

diversidadGrupo =  pd.DataFrame()
diversidadGrupo['categoria']=[0,1,2,3,4,5,6,7,8,9]
diversidadGrupo['cantidad']=group_referrals.groupby('categoria').size()
diversidadGrupo
```

	categoria	cantidad
0	0	NaN
1	1	11.0
2	2	15.0
3	3	6.0
4	4	3.0
5	5	1.0
6	6	2.0
7	7	2.0
8	8	1.0
9	9	1.0

De categoría 3 “modelos” hay 6 sobre un total de 9. Buscaremos los usuarios que están más cerca a los centroids de cada grupo que podríamos decir que tienen los rasgos de personalidad característicos que representan a cada cluster:

```
#vemos el representante del grupo, el usuario cercano a su centroid
closest, _ = pairwise_distances_argmin_min(kmeans.cluster_centers_, X)
closest
```

```
array([21, 107, 82, 80, 91]) #posicion en el array de usuarios
```

```
users=dataframe['usuario'].values
for row in closest:
    print(users[row])
```

```
carmenelectra Pablo_Iglesias_ JudgeJudy JPVarsky kobebryant
```

En los centros vemos que tenemos una modelo, un político, presentadora de Tv, locutor de Radio y un deportista.

Clasificar nuevas muestras

Y finalmente podemos agrupar y etiquetar nuevos usuarios twitter con sus características y clasificarlos. Vemos el ejemplo con el usuario de David Guetta y nos devuelve que pertenece al grupo 1 (verde).

```
X_new = np.array([[45.92,57.74,15.66]]) #davidguetta

new_labels = kmeans.predict(X_new)
print(new_labels)
```

```
[1]
```

Resumen

El algoritmo de K-means nos ayudará a crear clusters cuando tengamos grandes grupos de datos sin etiquetar, cuando queramos intentar descubrir nuevas relaciones entre features o para probar o declinar hipótesis que tengamos de nuestro negocio.
Atención: Puede haber casos en los que **no existan grupos naturales**, o clusters que contengan una verdadera razón de ser. Si bien K-means siempre nos brindará "k clusters", quedará en nuestro criterio reconocer la utilidad de los mismos o bien revisar nuestras features y descartar las que no sirven o conseguir nuevas.
También tener en cuenta que en este ejemplo estamos utilizando como medida de similitud entre features la distancia Euclideana[41] pero podemos utilizar otras diversas funciones que podrían arrojar mejores resultados (como Manhattan[42], Lavenshtein[43], Mahalanobis[44], etc). Hemos visto una descripción del algoritmo, aplicaciones y un ejemplo python paso a paso, que podrán descargar también desde los siguientes enlaces:

[41] https://es.wikipedia.org/wiki/Distancia_euclidiana
[42] https://es.wikipedia.org/wiki/Geometr%C3%ADa_del_taxista
[43] https://en.wikipedia.org/wiki/Levenshtein_distance
[44] https://en.wikipedia.org/wiki/Mahalanobis_distance

- Notebook Jupiter Online[45]
- Descargar archivo csv[46] y notebook ejercicio K-means[47]
- Visualizar[48] y descargar desde jbagnato Github[49]

[45] http://nbviewer.jupyter.org/github/jbagnato/machine-learning/blob/master/Ejercicio_K_Means.ipynb
[46] http://www.aprendemachinelearning.com/wp-content/uploads/2018/03/analisis.csv
[47] http://www.aprendemachinelearning.com/wp-content/uploads/2018/03/Ejercicio_K_Means.ipynb
[48] https://github.com/jbagnato/machine-learning/blob/master/Ejercicio_K_Means.ipynb
[49] https://github.com/jbagnato/machine-learning

K-Nearest-Neighbor

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es el algoritmo k-Nearest Neighbor ?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Dónde se aplica k-Nearest Neighbor?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pros y contras

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona kNN?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Un ejemplo k-Nearest Neighbor en Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Exploremos el algoritmo con Scikit learn

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El Ejercicio: App Reviews

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comencemos con el código!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Un poco de Visualización

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Preparamos las entradas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Usemos k-Nearest Neighbor con Scikit Learn

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Precisión del modelo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Y ahora, la gráfica que queríamos ver!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Elegir el mejor valor de k

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

(sobre todo importante para desempatar o elegir los puntos frontera!)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Clasificar ó Predecir nuevas muestras

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos y enlaces

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Naive Bayes: ¿Comprar casa o Alquilar?

Los Datos de Entrada:

El teorema de Bayes

Clasificador Gaussian Naive Bayes

Visualización de Datos

Preparar los datos de entrada

Feature Selection ó Selección de Características

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crear el modelo Gaussian Naive Bayes con SKLearn

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Probemos el modelo: ¿Comprar o Alquilar?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos Adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Sistemas de Recomendación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crea en Python un motor de recomendación con Collaborative Filtering

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué son los Sistemas ó Motores de Recomendación?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Tipos de motores

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona Collaborative Filtering?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejemplo de Dataset

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Tipos de Collaborative Filtering

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Predecir gustos (User-based)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Calcular los Ratings

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejercicio en Python: “Sistema de Recomendación de Repositorios Github”

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Vamos al código!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos la matriz usuarios/ratings

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Sparcity

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Dividimos en Train y Test set

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Matriz de Similitud: Distancias por Coseno

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Predicciones -ó llamémosle “Sugeridos para ti”-

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Validemos el error

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Hay más...

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos del Artículo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Otros artículos de interés (en inglés)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Breve Historia de las Redes Neuronales Artificiales

Arquitecturas y Aplicaciones de las Redes Neuronales

Vamos a hacer un repaso por las diversas estructuras inventadas, mejoradas y utilizadas a lo largo de la historia para crear redes neuronales y sacar el mayor potencial al Deep Learning[50] para resolver toda clase de problemas[51] de regresión y clasificación.

Evolución de las Redes Neuronales en Ciencias de la Computación

Vamos a revisar las siguientes redes/arquitecturas:

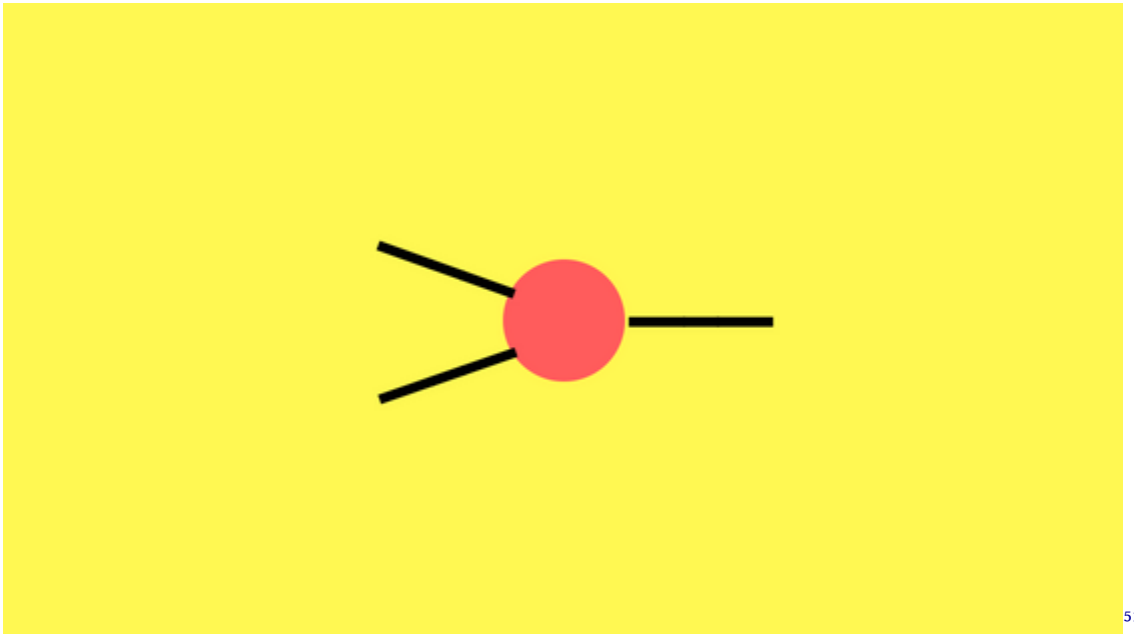
- 1958 - Perceptron
- 1965 - Multilayer Perceptron
- 1980's
  - Neuronas Sigmoidales
  - Redes Feedforward
  - Backpropagation
- 1989 - Convolutional Neural Networks (CNN) / Recurent Neural Networks (RNN)
- 1997 - Long Short Term Memory (LSTM)
- 2006 - Deep Belief Networks (DBN): **Nace el Deep Learning**
  - Restricted Boltzmann Machine
  - Encoder / Decoder = Auto-encoder
- 2014 - Generative Adversarial Networks (GAN)

Si bien esta lista no es exhaustiva y no se abarcan todos los modelos creados desde los años 50, he recopilado las que fueron **las redes y tecnologías más importantes** desarrolladas para llegar al punto en que estamos hoy: **el Aprendizaje Profundo**.

[50] http://www.aprendemachinelearning.com/aprendizaje-profundo-una-guia-rapida/
[51] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/

El inicio de todo: la neurona artificial

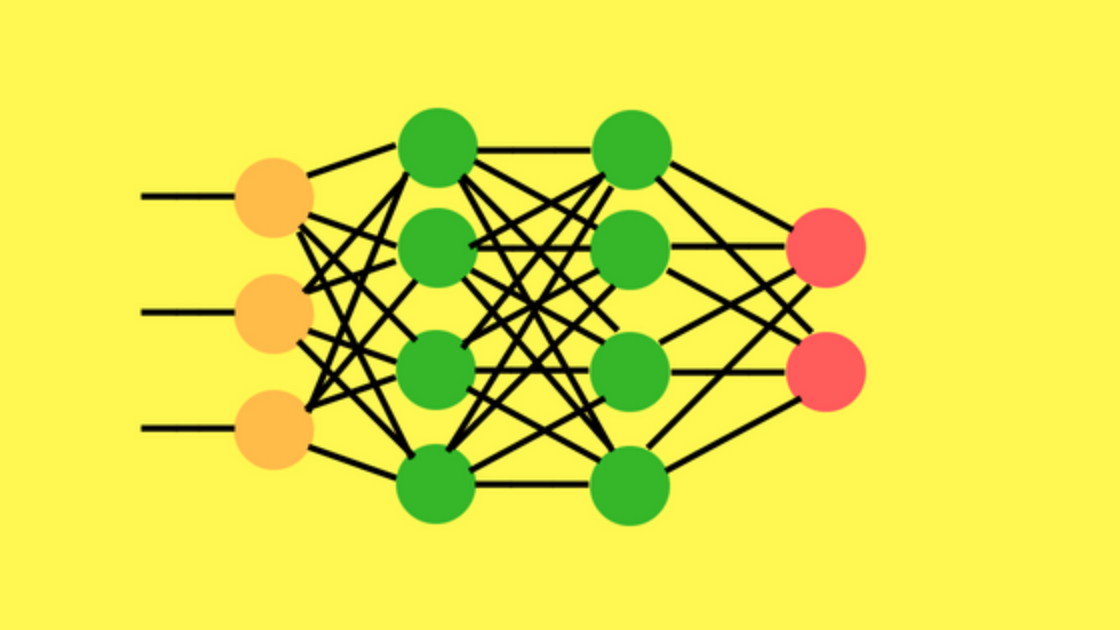
1958 - Perceptron

 [52]

Entre las décadas de 1950 y 1960 el científico Frank Rosenblatt[53], inspirado en el trabajo de Warren McCulloch y Walter Pitts creó el Perceptron, la unidad desde donde nacería y se potenciarían las redes neuronales artificiales. Un perceptron toma varias entradas binarias x1, x2, etc y produce una sóla **salida binaria**. Para calcular la salida, Rosenblatt introduce el concepto de *"pesos"* w1, w2, etc, un número real que expresa la importancia de la respectiva entrada con la salida. La salida de la neurona será 1 o 0 si la suma de la multiplicación de pesos por entradas es mayor o menor a un determinado *umbral*. Sus principales usos son decisiones binarias sencillas, o funciones lógicas como OR, AND.

[52] http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/net_perceptron.png
[53] https://es.wikipedia.org/wiki/Frank_Rosenblatt

1965 - Multilayer Perceptron

 [54]

Como se imaginarán, el multilayer perceptron es una "amplicación" del perceptrón de una única neurona a más de una. Además aparece el concepto de capas de entrada, oculta y salida. Pero con valores de entrada y salida binarios. No olvidemos que tanto el valor de los pesos como el de umbral de cada neurona **lo asignaba manualmente el científico**. Cuantos más perceptrones en las capas, mucho más difícil conseguir los pesos para obtener salidas deseadas.

Los 1980s: aprendizaje automático

Neuronas Sigmoides

Para poder lograr que las redes de neuronas aprendieran solas fue necesario introducir un nuevo tipo de neuronas. Las llamadas Neuronas Sigmoides son similares al perceptron, pero permiten que las entradas, en vez de ser ceros o unos, puedan tener valores reales como 0,5 ó 0,377 ó lo que sea. También aparecen las neuronas "bias" que siempre suman 1 en las diversas capas para resolver ciertas situaciones. Ahora las salidas en vez de ser 0 ó 1, será d(w . x + b) donde d será la función sigmoide definida como d(z) = 1/(1 +e-z). **Esta es la primer función de activación!**

[54] http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/net_multilayer.png

[55]

Imagen de la Curva Logística Normalizada de Wikipedia

Con esta nueva fórmula, se puede lograr que **pequeñas alteraciones en valores de los pesos (deltas) produzcan pequeñas alteraciones en la salida**. Por lo tanto, podemos ir ajustando muy de a poco los pesos de las conexiones e ir obteniendo las salidas deseadas.

Redes Feedforward

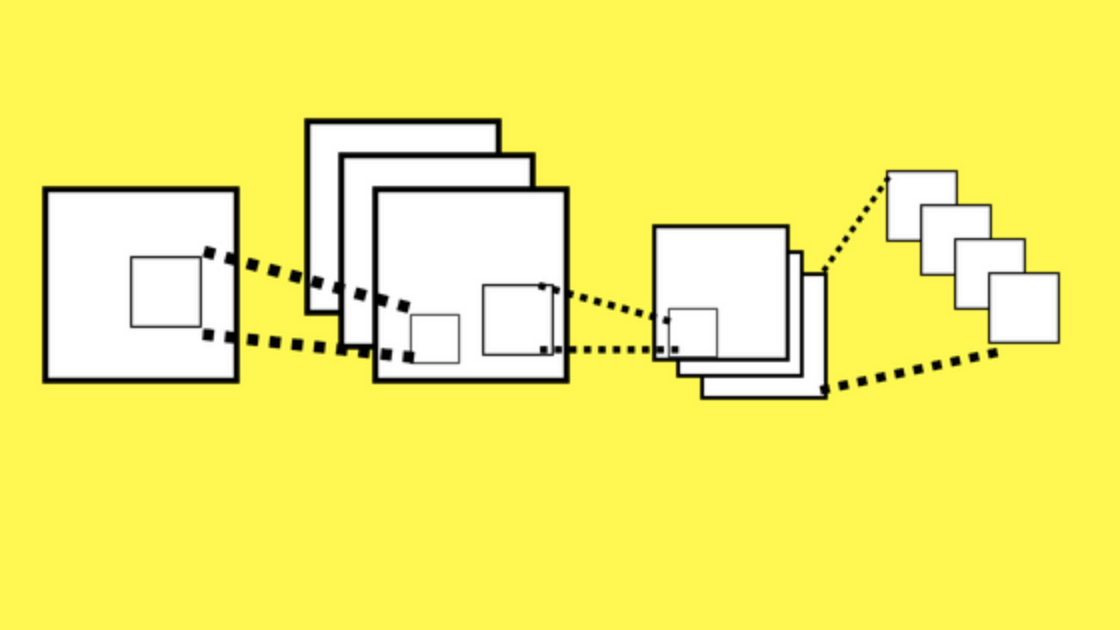
Se les llama así a las *redes en que las salidas de una capa son utilizadas como entradas en la próxima capa*. Esto quiere decir que no hay loops "hacia atrás". Siempre se "alimenta" de valores hacia adelante. Hay redes que veremos más adelante en las que sí que existen esos loops (Recurrent Neural Networks). Además existe el concepto de "fully connected Feedforward Networks" y se refiere a que todas las neuronas de entrada, están conectadas con todas las neuronas de la siguiente capa.

1986 - Backpropagation

Gracias al algoritmo de backpropagation[56] se hizo posible entrenar redes neuronales de multiples capas de manera supervisada. Al calcular el error obtenido en la salida e ir propagando hacia las capas anteriores se van haciendo ajustes pequeños (minimizando costo) en cada iteración para *lograr que la red aprenda* consiguiendo que la red pueda clasificar las entradas correctamente.

[55] http://www.aprendemachinelearning.com/wp-content/uploads/2018/07/Logistic-curve.png
[56] https://es.wikipedia.org/wiki/Propagaci%C3%B3n_hacia_atr%C3%A1s

1989 - Convolutional Neural Network

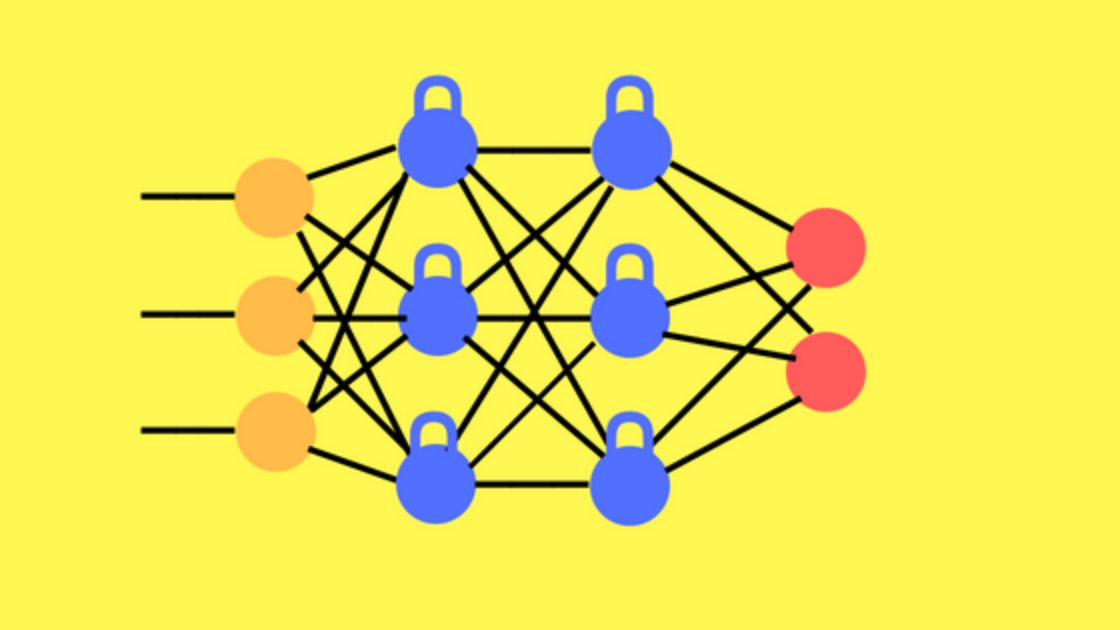
 [57]

Las Convolutional Neural Networks[58] son redes multilayered que toman su inspiración del cortex visual de los animales. Esta arquitectura es útil en varias aplicaciones, principalmente **procesamiento de imágenes**. La primera CNN fue creada por Yann LeCun[59] y estaba enfocada en el reconocimiento de letras manuscritas. La arquitectura constaba de varias capas que implementaban la **extracción de características y luego clasificación**. La imagen se divide en campos receptivos que alimentan una capa convolutional que extrae features de la imagen de entrada (Por ejemplo, detectar lineas verticales, vértices, etc). El siguiente paso es *pooling* que reduce la dimensionalidad de las features extraídas **manteniendo la información más importante**. Luego se hace una nueva convolución y otro pooling que alimenta una red feedforward multicapa. La salida final de la red es un grupo de nodos que clasifican el resultado, por ejemplo un nodo para cada número del 0 al 9 (es decir, 10 nodos, se "activan" de a uno).

Esta arquitectura usando capas profundas y la clasificación de salida abrieron un mundo nuevo de posibilidades en las redes neuronales. Las CNN se usan también en reconocimiento de video y tareas de Procesamiento del Lenguaje natural.

[57] http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/net_convolutional.png
[58] http://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/
[59] https://en.wikipedia.org/wiki/Yann_LeCun

1997 Long Short Term Memory / Recurrent Neural Network

 [60]

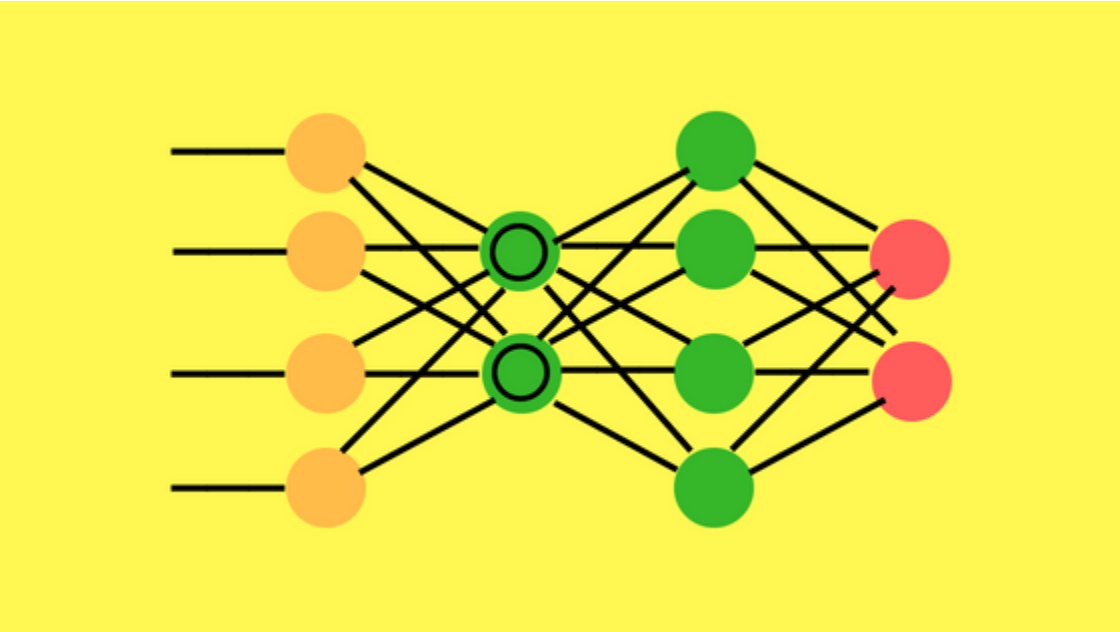
Aqui vemos que la red LSTM tiene neuronas ocultas con loops hacia atrás (en azul). Esto permite que almacene información en celdas de memoria.

Las Long short term memory son un tipo de Recurrent neural network. Esta arquitectura permite conexiones "hacia atrás" entre las capas. Esto las hace buenas para procesar datos de tipo "time series" (datos históricos). En 1997 se crearon las LSTM que consisten en unas celdas de memoria que permiten a la red recordar valores por períodos cortos o largos. Una celda de memoria contiene **compuertas que administran cómo la información fluye dentro o fuera**. La puerta de entrada controla cuando puede entran nueva información en la memoria. La puerta de *"olvido"* controla cuanto tiempo existe y se retiene esa información. La puerta de salida controla cuando la información en la celda es usada como salida de la celda. La celda contiene pesos que controlan cada compuerta. El algoritmo de entrenamiento -conocido como **backpropagation-through-time** optimiza estos pesos basado en el error de resultado. Las LSTM se han aplicado en reconocimiento de voz, de escritura, text-to-speech y otras tareas.

[60] http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/net_lstm.png

Se alcanza el Deep Learning

2006 - Deep Belief Networks (DBN)

 [61]

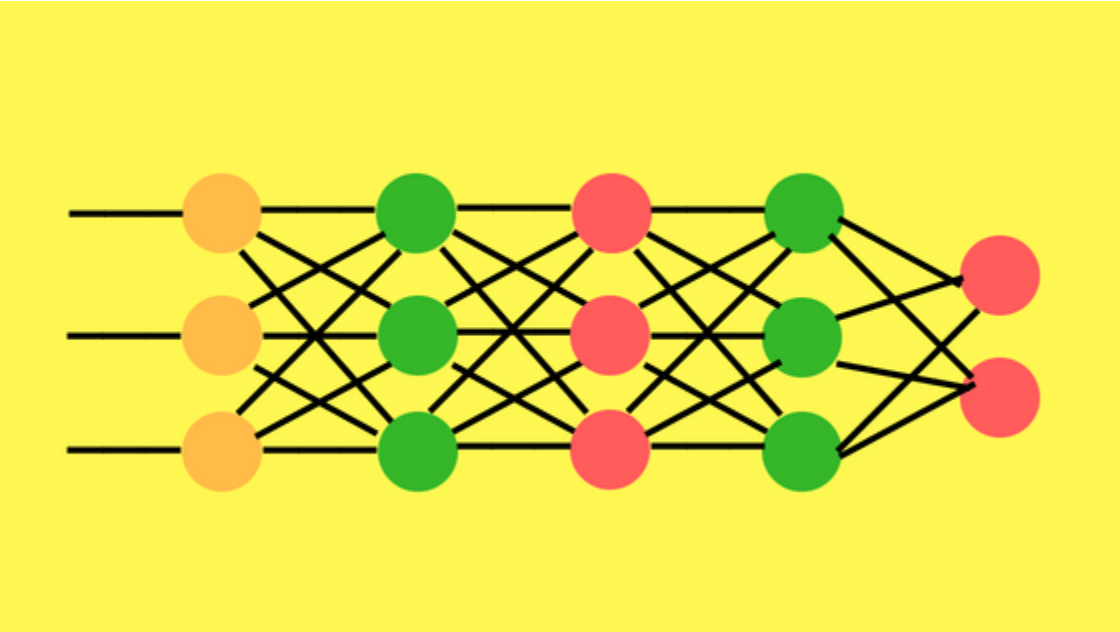
La Deep Belief Network utiliza un Autoencoder con Restricted Boltzmann Machines para preentrenar a las neuronas de la red y obtener un mejor resultado final.

Antes de las DBN en 2006 los modelos con "profundidad" (decenas o cientos de capas) eran considerados demasiado difíciles de entrenar (incluso con backpropagation) y el uso de las redes neuronales artificiales quedó estancado. Con la creación de una DBN que logro obtener un mejor resultado en el dataset MNIST[62], se devolvió el entusiasmo en poder lograr el aprendizaje profundo en redes neuronales. Hoy en día las DBN no se utilizan demasiado, pero fueron un gran hito en la historia en el desarrollo del deep learning y permitieron seguir la exploración para mejorar las redes existentes CNN, LSTM, etc. Las Deep Belief Networks, demostraron que **utilizar pesos aleatorios al inicializar las redes son una mala idea**: por ejemplo al utilizar Backpropagation con Descenso por gradiente muchas veces se caía en mínimos locales, sin lograr optimizar los pesos. Mejor será utilizar una asignación de pesos inteligente mediante un preentrenamiento de las capas de la red. Se basa en el uso de la utilización de Restricted Boltzmann Machines[63] y Autoencoders[64] para pre-entrenar la red de manera **no supervisada**[65]. Ojo! luego de pre-entrenar y asignar esos pesos iniciales, deberemos entrenar la red de forma habitual, supervisada[66] (con backpropagation). Ese preentrenamiento es una de las causas de la gran mejora en las redes neuronales permitió el deep learning: pues para asignar

[61] http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/deep_beleif.png
[62] https://tensorflow.rstudio.com/tensorflow/articles/tutorial_mnist_beginners.html
[63] https://towardsdatascience.com/deep-learning-meets-physics-restricted-boltzmann-machines-part-i-6df5c4918c15
[64] https://towardsdatascience.com/the-variational-autoencoder-as-a-two-player-game-part-i-4c3737f0987b
[65] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/#no_supervisado
[66] http://www.aprendemachinelearning.com/aplicaciones-del-machine-learning/#supervisado

los valores se evalúa capa a capa, **de a una**, y no "sufre" de cierto sesgo que causa el backpropagation, al entrenar a todas las capas en simultáneo.

2014 - Generative Adversarial Networks



](http://www.aprendemachinelearning.com/wp-content/uploads/2018/09/gan.png)
Las GAN, entrenan dos redes neuronales en simultáneo. La red de Generación y la red de Discriminación. A medida que la máquina aprende, comienza a crear muestras que son indistinguibles de los datos reales.

Estas redes pueden aprender a crear muestras, de manera similar a los datos con las que las alimentamos. La idea detrás de GAN es la de **tener dos modelos de redes neuronales compitiendo**. Uno, llamado *Generador*, toma inicialmente "datos basura" como entrada y genera muestras. El otro modelo, llamado *Discriminador*, recibe a la vez muestras del Generador y del conjunto de entrenamiento (real) y deberá ser capaz de diferenciar entre las dos fuentes. Estas dos redes juegan una partida continua donde **el Generador aprende a producir muestras más realistas y el Discriminador aprende a distinguir entre datos reales y muestras artificiales**. Estas redes son entrenadas simultáneamente para finalmente lograr que los datos generados no puedan diferenciarse de datos reales. Sus aplicaciones principales son la de generación de imágenes artificiales realistas, pero también la de mejorar imágenes ya existentes, o generar textos (captions) en imágenes, o generar textos siguiendo un estilo determinado y hasta para el desarrollo de moléculas para industria farmacéutica.

Resumen

Hemos recorrido estos primeros -casi- 80 años de avances en las redes neuronales en la historia de la inteligencia artificial. Se suele dividir en 3 etapas, **del 40 al 70** en donde se pasó del asombro de estos nuevos modelos hasta el escepticismo, el retorno de un *invierno de 10 años* cuando en **los ochentas** surgen mejoras en mecanismos y maneras de entrenar las redes (backpropagation[67]) y se alcanza una meseta en la que no se puede alcanzar la "profundidad" de aprendizaje seguramente también por falta de poder de cómputo. Y **una tercer etapa a partir de 2006** en la que se logra superar esa barrera y aprovechando el poder de las GPU y nuevas técnicas se logra entrenar cientos de capas jerárquicas que conforman y potencian el Deep Learning y *dan una capacidad casi ilimitada a estas redes*. Como último comentario, me gustaría decir que recientemente (feb 2018[68]) surgieron nuevos estudios de las neuronas humanas biológicas en las que se está redescubriendo su funcionamiento y se está produciendo una nueva revolución, pues parece que es totalmente distinto a lo que hasta hoy conocíamos. Esto puede ser el principio de una nueva etapa totalmente nueva y seguramente mejor del Aprendizaje Profundo, el Machine Learning[69] y la Inteligencia Artificial.

Más recursos

* Cheat Sheets for AI[70]
* Neural Networks and Deep Learning[71]
* From Perceptrons to deep networks[72]
* Neural Networks Architectures[73]
* A beginners guide to Machine Learning[74]
* A guide on Time Series Prediction using LSTM[75]
* Convolutional Neural Networks in Python with Keras[76]
* History of Neural Network[77]

[67] http://www.aprendemachinelearning.com/crear-una-red-neuronal-en-python-desde-cero/
[68] https://medium.com/intuitionmachine/neurons-are-more-complex-than-what-we-have-imagined-b3dd00a1dcd3
[69] http://www.aprendemachinelearning.com/que-es-machine-learning/
[70] https://becominghuman.ai/cheat-sheets-for-ai-neural-networks-machine-learning-deep-learning-big-data-678c51b4b463
[71] http://neuralnetworksanddeeplearning.com/chap1.html#perceptrons
[72] https://www.toptal.com/machine-learning/an-introduction-to-deep-learning-from-perceptrons-to-deep-networks
[73] https://ml-cheatsheet.readthedocs.io/en/latest/architectures.html
[74] https://www.ibm.com/developerworks/library/cc-beginner-guide-machine-learning-ai-cognitive/index.html
[75] https://blog.statsbot.co/time-series-prediction-using-recurrent-neural-networks-lstms-807fa6ca7f
[76] https://www.datacamp.com/community/tutorials/convolutional-neural-networks-python
[77] https://medium.com/@karthikeyana97/history-of-neural-network-d1333760f0c5

Aprendizaje Profundo: una Guía rápida

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Deep Learning y Redes Neuronales -sin código-

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona el Deep Learning? Mejor un Ejemplo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos una Red Neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo se calcula la predicción?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Entrenando Nuestra Red Neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo reducimos la función coste -y mejoramos las predicciones-?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crear una Red Neuronal en Python desde cero

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El proyecto

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Funciones Sigmoide

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Forward Propagation -ó red Feedforward-

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Backpropagation (cómputo del gradiente)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Fase 1: Propagar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Fase 2: Actualizar Pesos:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El Código de la red Neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Programa un coche Robot Arduino que conduce con IA

La Nueva Red Neuronal

El coche Arduino

Necesitaremos:

Circuito del coche

Montar el coche

Copiar la red neuronal

El código Arduino

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El Coche en Acción!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Conecta tu coche, sube el código y ¡pruébalo!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Mejoras a Futuro

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Una sencilla Red Neuronal con Keras y Tensorflow

Requerimientos para el ejercicio

Las compuertas XOR

Una Red Neuronal Artificial sencilla con Python y Keras

¿Keras y Tensorflow? What??

Analicemos la red neuronal que hicimos

Visualización de la red Neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

A Entrenar la red!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resultados del Entrenamiento

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Evaluamos y Predecimos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Afinando parámetros de la red neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Guardar la red y usarla -de verdad-

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Vale la pena una red neuronal?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Código Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Otros recursos Keras y Tensorflow

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pronóstico de Series Temporales con Redes Neuronales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es una serie temporal y qué tiene de especial?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejemplo de series temporales:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Cargar el Ejemplo con Pandas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Visualización de datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo hacer pronóstico de series temporales?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pronóstico de Ventas Diarias con Redes Neuronal

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Preparamos los datos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Creamos la Red Neuronal Artificial

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Entrenamiento y Resultados

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pronóstico de ventas futuras

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos Adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pronóstico de Ventas con Redes Neuronales (Parte 2)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Mejora del modelo de Series Temporales con Múltiples Variables y Embeddings

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Breve Repaso de lo que hicimos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Mejoras al modelo de Series Temporales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Primer Mejora: Serie Temporal de múltilples Variables

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Fecha como variable de entrada

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Segunda mejora: Embeddings en variables categóricas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué son los Embeddings?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Sobre la dimensionalidad de los Embeddings

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Conclusión de Embeddings

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Quiero Python!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparemos los Resultados de los 3 modelos:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparemos las Métricas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparemos Gráficas de Pérdida (loss)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparemos las Gráficas de Accuracy

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparamos los pronósticos y sus aciertos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos Adicionales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crea tu propio servicio de Machine Learning con Flask

Dale vida a tu IA

Ya tienes tu modelo[78], probado, funciona bien y está listo para entrar en acción. Entonces, ¿cómo lo desplegamos? Si es una solución que quieres ofrecer al público desde la nube, puedes implementar tu propio servicio online y ofrecer soluciones de Machine Learning!

Veamos cómo hacerlo!

Implementar modelos de Machine Learning

Muchas veces el modelo creado por el equipo de Machine Learning, será una "pieza más" de un sistema mayor, como por ejemplo una app, un chatbot, algún sistema de marketing, un sistema de monitoreo de seguridad. Y si bien el modelo puede correr en Python o R, es probable que interactúe con otro stack distinto de desarrollo. Por ejemplo, una app Android en Java ó Kotlin, algún sistema PHP en la nube ó hasta podría ser aplicaciones de escritorio o CRM. Entonces, nuestro modelo deberá ser capaz de interactuar y "servir" a los pedidos de esas otras herramientas.

Podríamos reescribir nuestro código en otro lenguaje (javascript, java, c...) ó si nuestro proceso ejecutara batch, podría ser una "tarea cron" del sistema operativo que ejecute automáticamente cada X tiempo y deje un archivo de salida CSV (ó entradas en una base de datos).

Pero, si nuestro modelo tiene que servir a otros sistemas en tiempo real y no podemos reescribirlo (incluso por temas de mantenimiento futuro, actualización ó imposibilidad de recrear módulos completos de Python) podemos desplegar nuestro modelo en una API accesible desde un servidor (que podría ser público ó privado) y mediante una clave secreta -si hiciera falta-.

Servir mediante una API

Una API es la manera más flexible que hay para ofrecer servicios online en la actualidad. Sin meterme en profundidad en el tema, podemos decir que lo que hacemos es publicar un punto de entrada desde donde los usuarios (clientes, apps, u otras máquinas) harán **peticiones de consulta, inserción, actualización o borrado** de los datos a los que tienen acceso. En nuestro caso, lo típico será ofrecer un servicio de Machine Learning de predicción ó clasificación. Entonces nos llegarán en

[78] https://www.aprendemachinelearning.com/principales-algoritmos-usados-en-machine-learning/

la petición GET ó POST las entradas que tendrá el modelo (nuestras features, ó lo que normalmente son las "columnas" de un csv que usamos para entrenar). Y nuestra salida podría ser el resultado de la predicción, ó una probabilidad, ó un número (por ej. "cantidad de ventas pronosticadas para ese día").

Para crear una API, podemos utilizar diversas infraestructuras ya existentes en el mercado que ofrecen Google, Amazon, Microsoft (u otros) ó podemos "levantar" nuestro propio servicio con Flask. Flask es un web framework en Python que simplifica la manera de publicar nuestra propia API (Hay otros como Django, Falcon y más).

Instalar Flask

Veamos rápidamente como instalar y dejar montado Flask.

- Instalar Anaconda[79] en el servidor ó en nuestra máquina local para desarrollo. (Para servidores también puedes usar la versión de mini-conda[80])
- Prueba ejecutar el comando "conda" en el terminal para verificar que esté todo ok.
- Crear un nuevo environment en el que trabajaremos `conda create --name mi_ambiente python=3.6`
- Activa el ambiente creado con `source activate mi_ambiente`
- Instalar los paquetes Python que utilizaremos: `pip install flask gunicorn` **NOTA**: para usuarios Windows, utilizar Waitress[81]

Hagamos un "Hello world" con Flask. Crea un archivo de texto nuevo llamado "mi_server.py[82]"

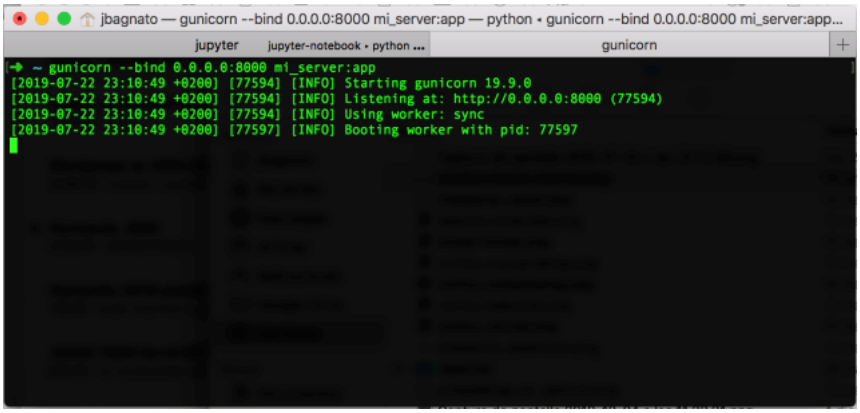
```
"""Creando un servidor Flask
"""

from flask import Flask

app = Flask(__name__)

@app.route('/users/')
def hello_world(nombre=None):

    return("Hola {}!".format(nombre))
```

Guarda el archivo y escribe en la terminal:

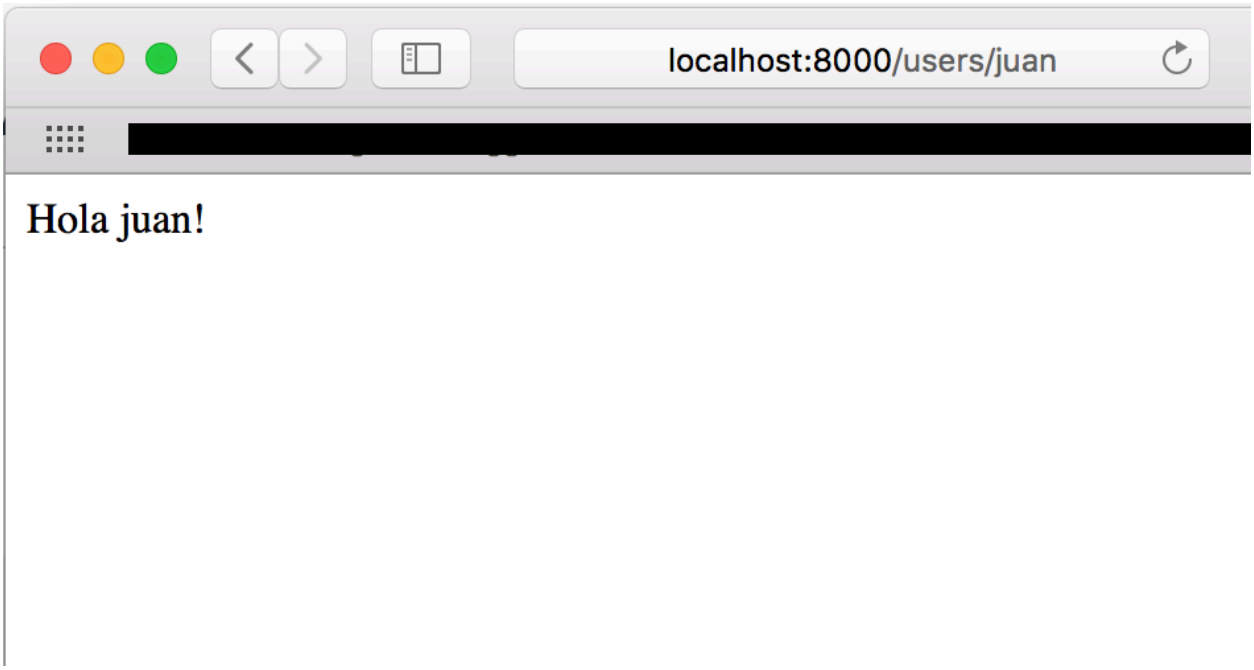
[79] https://www.aprendemachinelearning.com/instalar-ambiente-de-desarrollo-python-anaconda-para-aprendizaje-automatico/
[80] https://conda.io/miniconda.html
[81] https://docs.pylonsproject.org/projects/waitress/en/latest/
[82] https://github.com/jbagnato/machine-learning/blob/master/api_ml/mi_server.py

```
gunicorn --bind 0.0.0.0:8000 mi_server:app
```

una vez iniciado, verás algo así:



Entonces abre tu navegador web favorito y entra en la ruta `http://localhost:8000/users/juan`



Con eso ya tenemos nuestro servidor ejecutando. En breve haremos cambios para poder servir nuestro modelo de Machine Learning[83] desde Flask al mundo :)

NOTA: Usuarios Windows, seguir estas instrucciones[84] para el módulo Waitress.

[83] https://www.aprendemachinelearning.com/7-pasos-machine-learning-construir-maquina/
[84] https://stackoverflow.com/questions/51045911/serving-flask-app-with-waitress-on-windows

Crear el modelo de ML

Hagamos un ejemplo de un modelo de ML basándonos en el ejercicio de Pronóstico de Series Temporales que hace un pronóstico de ventas con redes neuronales con Embeddings. Esta vez no usaremos una notebook de Jupyter, si no, archivos de "texto plano" Python:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler

from utiles import *

df = pd.read_csv('time_series.csv',  parse_dates=[0], header=None,index_col=0, names\
=['fecha','unidades'])
df['weekday']=[x.weekday() for x in df.index]
df['month']=[x.month for x in df.index]
print(df.head())

EPOCHS=40
PASOS=7

scaler = MinMaxScaler(feature_range=(-1, 1))

reframed = transformar(df, scaler)

reordenado=reframed[ ['weekday','month','var1(t-7)','var1(t-6)','var1(t-5)','var1(t-\
4)','var1(t-3)','var1(t-2)','var1(t-1)','var1(t)'] ]
reordenado.dropna(inplace=True)

training_data = reordenado.drop('var1(t)', axis=1)
target_data=reordenado['var1(t)']
cant = len(df.index)
valid_data = training_data[cant-30:cant]
valid_target=target_data[cant-30:cant]

training_data = training_data[0:cant]
target_data=target_data[0:cant]
print(training_data.shape, target_data.shape, valid_data.shape, valid_target.shape)
print(training_data.head())

model = crear_modeloEmbeddings()

```

```
continuas = training_data[['var1(t-7)','var1(t-6)','var1(t-5)','var1(t-4)','var1(t-3\
)','var1(t-2)','var1(t-1)']]
valid_continuas = valid_data[['var1(t-7)','var1(t-6)','var1(t-5)','var1(t-4)','var1(\
t-3)','var1(t-2)','var1(t-1)']]

history = model.fit([training_data['weekday'],training_data['month'],continuas], tar\
get_data, epochs=EPOCHS,
                    validation_data=([valid_data['weekday'],valid_data['month'],vali\
d_continuas],valid_target))

results = model.predict([valid_data['weekday'],valid_data['month'],valid_continuas])

print( 'Resultados escalados',results )
inverted = scaler.inverse_transform(results)
print( 'Resultados',inverted )
```

Ya logramos entrenar un nuevo modelo del que estamos conformes. Ahora veamos cómo guardarlo para poder reutilizarlo en la API!

Guardar el modelo; Serialización de objetos en Python

El proceso de serialización consiste en poder transformar nuestro modelo ML -que es un objeto python- en ceros y unos que puedan ser almacenados en un archivo y que luego, al momento de la carga vuelva a regenerar ese mismo objeto, con sus características.

Aunque existen diversas maneras de guardar los modelos, comentemos rápidamente las que usaremos:

- **Pickle** de Python para almacenar objetos (en nuestro caso un Transformador que debemos mantener para "reconvertir los resultados escalados" al finalizar de entrenar)
- **h5py** para el modelo Keras (podemos guardar el modelo completo ó los pesos asociados a la red)

```
import pickle

#definimos funciones de guardar y cargar
def save_object(filename, object):
    with open(''+filename, 'wb') as file:
        pickle.dump(object, file)

def load_object(filename):
    with open(''+filename ,'rb') as f:
        loaded = pickle.load(f)
    return loaded

# guardamos los objetos que necesitaremos mas tarde
save_object('scaler_time_series.pkl', scaler)
model.save_weights("pesos.h5")

# cargamos cuando haga falta
loaded_scaler = load_object('scaler_time_series.pkl')
loaded_model = crear_modeloEmbeddings()
loaded_model.load_weights("pesos.h5")
```

Podemos comprobar a ver las predicciones sobre el set de validación antes y después de guardar los objetos y veremos que da los mismos resultados.

Crear una API con Flask

Ahora veamos el código con el que crearemos la API y donde incorporaremos nuestro modelo.

Utilizaremos los siguientes archivos:

- server.py[85] - El servidor Flask
- test_api.py[86] - Ejemplo de request POST para probar la API
- utiles.py[87] - las funciones comunes al proyecto
- api_train_model.py[88] - entreno y creación del modelo, una red neuronal con Embeddings (del ejercicio de TimeSeries[89]).
- time_series.csv[90] - archivo con datos para el ejercicio

Vamos a la acción:

[85] https://github.com/jbagnato/machine-learning/blob/master/api_ml/server.py
[86] https://github.com/jbagnato/machine-learning/blob/master/api_ml/test_api.py
[87] https://github.com/jbagnato/machine-learning/blob/master/api_ml/utiles.py
[88] https://github.com/jbagnato/machine-learning/blob/master/api_ml/api_train_model.py
[89] https://www.aprendemachinelearning.com/pronostico-de-ventas-redes-neuronales-python-embeddings/
[90] https://github.com/jbagnato/machine-learning/blob/master/api_ml/time_series.csv

- Crearemos un método inicial que será invocado desde la url “predict”
- Cargaremos el modelo que entrenamos previamente
- Responderemos peticiones en formato JSON

```
"""Filename: server.py
"""
import pandas as pd
from sklearn.externals import joblib
from flask import Flask, jsonify, request

from utiles import *

app = Flask(__name__)

@app.route('/predict', methods=['POST'])
def predict():
    """API request
    """
    try:
        req_json = request.get_json()
        input = pd.read_json(req_json, orient='records')
    except Exception as e:
        raise e

    if input.empty:
        return(bad_request())
    else:
        #Load the saved model
        print("Cargar el modelo...")
        loaded_model = crear_modeloEmbeddings()
        loaded_model.load_weights("pesos.h5")

        print("Hacer Pronosticos")
        continuas = input[['var1(t-7)','var1(t-6)','var1(t-5)','var1(t-4)','var1(t-3\
)','var1(t-2)','var1(t-1)']]
        predictions = loaded_model.predict([input['weekday'], input['month'], contin\
uas])

        print("Transformando datos")
        loaded_scaler = load_object('scaler_time_series.pkl')
        inverted = loaded_scaler.inverse_transform(predictions)
        inverted = inverted.astype('int32')
```

```
39
40          final_predictions = pd.DataFrame(inverted)
41          final_predictions.columns = ['ventas']
42
43          print("Enviar respuesta")
44          responses = jsonify(predictions=final_predictions.to_json(orient="records"))
45          responses.status_code = 200
46          print("Fin de Peticion")
47
48          return (responses)
```

Muy bien, podemos ejecutar nuestra API desde la terminal para testear con:

```
1  gunicorn --bind 0.0.0.0:8000 server:app
```

NOTA: Usuarios Windows, realizar la función similar de Waitress.

Y ahora hagamos una petición para probar nuestra API con un archivo Python y veamos la salida:

```
1  import json
2  import requests
3  import pandas as pd
4  import pickle
5  from utiles import *
6
7  """Setting the headers to send and accept json responses
8  """
9  header = {'Content-Type': 'application/json', \
10                    'Accept': 'application/json'}
11
12 # creamos un dataset de pruebas
13 df = pd.DataFrame({"unidades": [289,288,260,240,290,255,270,300],
14                    "weekday": [5,0,1,2,3,4,5,0],
15                    "month":   [4,4,4,4,4,4,4,4]})
16
17 loaded_scaler = load_object('scaler_time_series.pkl')
18
19 reframed = transformar(df, loaded_scaler)
20
21 reordenado=reframed[ ['weekday','month','var1(t-7)','var1(t-6)','var1(t-5)','var1(t-\
22 4)','var1(t-3)','var1(t-2)','var1(t-1)'] ]
23 reordenado.dropna(inplace=True)
24
```

```
"""Converting Pandas Dataframe to json
"""
data = reordenado.to_json(orient='records')

print('JSON para enviar en POST', data)

"""POST <url>/predict
"""
resp = requests.post("http://localhost:8000/predict", \
                     data = json.dumps(data),\
                     headers= header)

print('status',resp.status_code)


"""The final response we get is as follows:
"""
print('Respuesta de Servidor')
print(resp.json())
```

Este ejemplo nos devuelve de salida:

```
{'predictions': '[{"ventas":194}]'}
```

Actualizar el modelo (según sea necesario!)

No olvidar que si nuestro modelo es dependiente del tiempo, ó de datos históricos, ó datos nuevos que vamos recopilando (nuevas muestras) deberemos reentrenar el modelo.

Eso se podría automatizar, re-ejecutando el archivo de entrenamiento y sobreescribiendo el archivo de los pesos modelo "h5py" que habíamos generado antes cada X días ó a raíz de otro evento que en nuestro negocio sea significativo y sea el detonador del re-entreno.

Resumen

En este artículo vimos que nuestro modelo de ML puede llegar a ser una pequeña pieza de un puzzle mayor y puede ofrecer soluciones a usuarios finales ó a otros subsistemas. Para poder ofrecer sus servicios podemos contar con diversas soluciones, siendo una de ellas el despliegue de una API. Podemos crear una API fácil y rápidamente con el web framework de Flask. Ya puedes ofrecer tus modelos al mundo!

NOTAS finales: Recuerda ejecutar los archivos en el siguiente orden:

1. Copia el archivo timeseries.csv con los datos del ejercicio en tu server.
2. Entrena el modelo y crea los archivos necesarios con `python api_train_model.py`
3. Inicia el server desde una terminal con gunicorn como vimos anteriormente. (Usuarios Windows con Waitress)
4. Ejecuta el archivo de pruebas desde otra terminal con `python test_api.py`

Recursos Adicionales

Descarga los archivos creados en este artículo

- Archivos en GitHub[91]

Otros artículos relacionados en Inglés

- Flask with embedded machine learning[92]

[91] https://github.com/jbagnato/machine-learning/tree/master/api_ml
[92] https://www.bogotobogo.com/python/Flask/Python_Flask_Embedding_Machine_Learning_1.php

Clasificación de Imágenes en Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejercicio: Clasificar imágenes de deportes

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Requerimientos Técnicos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Vamos al código Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

1- Importar librerías

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

2-Cargar las imágenes

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

3- Crear etiquetas y clases

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

4-Creamos sets de Entrenamiento y Test, Validación y Preprocesar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

5 - Creamos la red (Aquí la Magia)

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

6-Entrenamos la CNN

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

7-Resultados de la clasificación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Los recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funcionan las Convolutional Neural Networks?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Muchas imágenes

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Pixeles y neuronas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

No Olvides: Pre-procesamiento

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Convoluciones

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Filtro: conjunto de kernels

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

La función de Activación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Subsampling

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Subsampling con Max-Pooling

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Ya terminamos? NO: ahora más convoluciones!!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Conectar con una red neuronal “tradicional”

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Y cómo aprendió la CNN a “ver”?: Backpropagation

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Comparativa entre una red neuronal “tradicional” y una CNN

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Arquitectura básica

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

No incluido

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Detección de Objetos con Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Agenda

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿En qué consiste la detección YOLO?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

(Algunos) Parámetros de la red

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El proyecto Propuesto: Detectar personajes de Lego

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Lo que tienes que instalar

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crea un dataset: Imágenes y Anotaciones

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recomendaciones para las imágenes:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Anotarlo Todo

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El lego dataset

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

El código Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Leer el Dataset

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Train y Validación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Data Augmentation

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crear la Red de Clasificación

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Crear la Red de Detección

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Generar las Anclas

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Entrenar la Red!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Revisar los Resultados

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Probar la Red

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Imágenes pero también Video y Cámara!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Anexo I: Webscraping

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejemplo Web Scraping en Python: IBEX35® la Bolsa de Madrid

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Contenidos:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Requerimientos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Conocimientos básicos de HTML y CSS

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Inspección Manual de la web

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Código webscraping Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Guardar CSV y ver en Excel

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Otros ejemplos útiles de Webscaping:

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Obtener los enlaces de una página web

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Anexo II: Machine Learning en la Nube

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Machine Learning en la Nube? Google Colaboratory con GPU!

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Machine Learning desde el Navegador

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

La GPU.... ¿en casa o en la nube?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es Google Colab?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Como se usa GoogleColab?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Enlazar con Google Drive

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejecutar una jupyter notebook de Github

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Descargar un recurso al cuaderno

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Descomprimir un archivo en el cuaderno

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Instalar otras librerías Python con Pip

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Otros Recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Anexo III: Principal Component Analysis

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Introducción a PCA

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Qué es Principal Component Analysis?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Cómo funciona PCA?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Selección de los Componentes Principales

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

¿Pero... porqué funciona PCA?

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Ejemplo “mínimo” en Python

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resumen

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Contras de PCA y variantes

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Resultados de PCA en el mundo real

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.

Más recursos

Este contenido no está disponible en el libro de muestra. El libro se puede comprar en Leanpub en http://leanpub.com/aprendeml.