Regresión logística
Logistic regression

Introducción

La regresión logística es un método estadístico que trata de modelar la probabilidad de una variable cualitativa binaria (dos posibles valores) en función de una o más variables independientes. La principal aplicación de la regresión logística es la creación de modelos de clasificación binaria.

Se llama regresión logística simple cuando solo hay una variable independiente y regresión logística múltiple cuando hay más de una. Dependiendo del contexto, a la **variable modelada** se le conoce como variable dependiente o variable respuesta, y a las **variables independientes** como regresores, predictores o features.

A lo largo de este documento, se describen de forma progresiva los fundamentos teóricos de la regresión logística, los principales aspectos prácticos a tener en cuenta y ejemplos de cómo crear este tipo de modelos en Python.

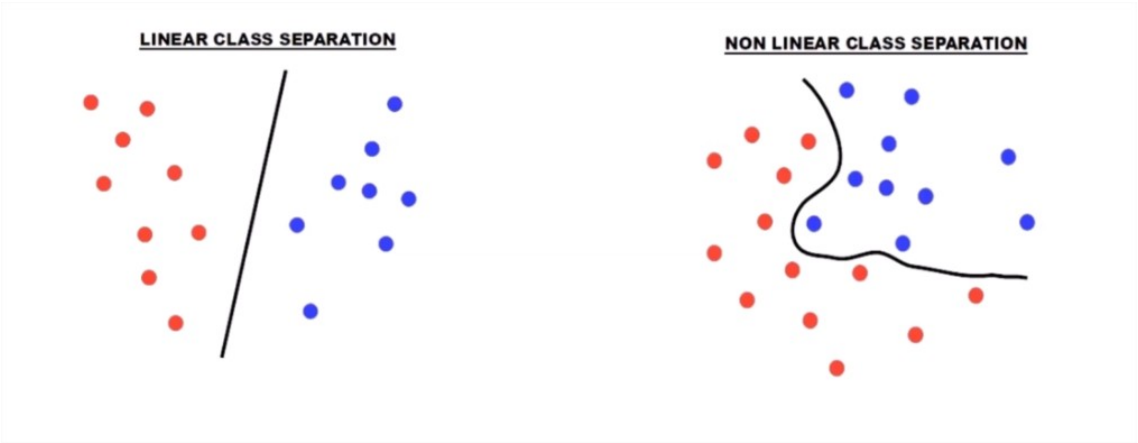
Modelos de regresión logística en Python

Dos de las implementaciones de modelos de regresión logística más utilizadas en Python son: scikit-learn y statsmodels. Aunque ambas están muy optimizadas, Scikit-learn está orientada principalmente a la predicción, por lo que no dispone de apenas funcionalidades que muestren las muchas características del modelo que se deben analizar para hacer inferencia. Statsmodels es más completa en este sentido.

Regresión logística
Logistic regression

@mihifidem
creativity is intelligence having fun

¿Cuál es la diferencia entre regresión lineal y regresión logística?



Ambos algoritmos son bastante diferentes principalmente porque uno se utiliza para proyectos relacionados a regresión mientras que el otro es para proyectos de clasificación. Veamos de manera sencilla esto.

Regresión logística
Logistic regression

La Regresión Lineal, se utiliza para proyectos de regresión, es decir cuando queremos predecir un valor numérico, por ejemplo, queremos predecir el precio de una casa o el sueldo de una persona de acuerdo a los años de experiencia.

Por su parte el algoritmo de Regresión Logística se utiliza para proyectos de clasificación, si queremos conocer si una persona vive o no el naufragio del Titanic, o si una acción de la bolsa de valores va a subir o no.

Por lo tanto, la Regresión Lineal es un algoritmo de regresión mientras que la Regresión Logística es un algoritmo de clasificación.

Pero veamos esto con un ejemplo práctico en donde implementemos ambos algoritmos, tenemos los datos históricos del tiempo de una ciudad, entonces queremos predecir lo siguiente:

Si durante el día va a llover y cuál es la probabilidad de que llueve muy fuerte

Regresión logística
Logistic regression

¿Con la explicación anterior ya has deducido cuál algoritmo debemos implementar?

Para el primer caso, que es el de predecir si va a llover, debemos utilizar el algoritmo de Regresión Logística, ya que acá vamos a predecir una de dos opciones, si llueve o no llueve, por lo que utilizamos un algoritmo de clasificación.

Por su parte para calcular la probabilidad de que llueva muy fuerte utilizamos el algoritmo de Regresión Lineal, ya que debemos calcular el porcentaje de probabilidad, por lo que es un valor numérico, por lo tanto, debemos implementar un algoritmo de regresión.

En conclusión, la Regresión Lineal es un algoritmo de regresión por lo que la utilizamos para predecir un valor numérico, mientras que la Regresión Logística es un algoritmo de clasificación por lo que la utilizamos para predecir entre dos opciones.

Regresión logística
Logistic regression

Breve Introducción a la Regresión Logística

A partir de un conjunto de datos de entrada (características), nuestra salida será discreta (y no continua) por eso utilizamos Regresión Logística (y no Regresión Lineal). La Regresión Logística es un Algoritmo **Supervisado** y se utiliza para **clasificación**.

Vamos a clasificar problemas con dos posibles estados "SI/NO": binario o un número finito de "etiquetas" o "clases": múltiple. Algunos Ejemplos de Regresión Logística son:

- Clasificar si el correo que llega es Spam o No es Spam
- Dados unos resultados clínicos de un tumor clasificar en "Benigno" o "Maligno".
- El texto de un artículo a analizar es: Entretenimiento, Deportes, Política ó Ciencia
- A partir de historial bancario conceder un crédito o no

Confiaremos en la implementación del paquete sklearn en Python para ponerlo en práctica.

Regresión logística
Logistic regression

Ejercicio de Regresión Logística en Python

Para nuestro ejercicio he creado un archivo csv con datos de entrada a modo de ejemplo para clasificar si el usuario que visita un sitio web usa como sistema operativo Windows, Macintosh o Linux.

Nuestra información de entrada son 4 características que tomé de una web que utiliza Google Analytics y son:

- Duración de la visita en Segundos
- Cantidad de Páginas Vistas durante la Sesión
- Cantidad de Acciones del usuario (click, scroll, uso de checkbox, sliders,etc)
- Suma del Valor de las acciones (cada acción lleva asociada una valoración de importancia)

Como la salida es discreta, asignaremos los siguientes valores a las etiquetas:

0 – Windows
1 – Macintosh
2 -Linux

La muestra es pequeña: son 170 registros para poder comprender el ejercicio, pero recordemos que para conseguir buenos resultados siempre es mejor contar con un número abundante de datos que darán mayor exactitud a las predicciones y evitarán problemas de overfitting u underfitting. (Por decir algo, de mil a 5 mil registros no estaría mal).

Requerimientos técnicos

Para ejecutar el código necesitas tener instalado Python -tanto la versión 2.7 o 3.6- y varios paquetes usados comúnmente en Data Science. Recomiendo tener instalada la suite Anaconda o Canopy, muy sencillas y con los paquetes para “Data Science” ya pre-instalados y funcionan en todas las plataformas.

Regresión logística
Logistic regression

Regresión Logística con SKLearn:

Identificar Sistema Operativo de los usuarios

Para comenzar hacemos los Import necesarios con los paquetes que utilizaremos en el Ejercicio.

```
import pandas as pd
import numpy as np
from sklearn import linear_model
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sb
%matplotlib inline
```

Leemos el archivo csv (por sencillez, se considera que estará en el mismo directorio que el archivo de notebook .ipynb) y lo asignamos mediante Pandas a la variable dataframe.

Mediante el método dataframe.head() vemos en pantalla los 5 primeros registros.

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

	duracion	paginas	acciones	valor	clase
0	7.0	2	4	8	2
1	21.0	2	6	6	2
2	57.0	2	4	4	2
3	101.0	3	6	12	2
4	109.0	2	6	12	2

Regresión logística
Logistic regression

A continuación llamamos al método dataframe.describe() que nos dará algo de información estadística básica de nuestro set de datos. La Media, el desvío estándar, valores mínimo y máximo de cada característica

```
dataframe.describe()
```

	duracion	paginas	acciones	valor	clase
count	170.000000	170.000000	170.000000	170.000000	170.000000
mean	111.075729	2.041176	8.723529	32.676471	0.752941
std	202.453200	1.500911	9.136054	44.751993	0.841327
min	1.000000	1.000000	1.000000	1.000000	0.000000
25%	11.000000	1.000000	3.000000	8.000000	0.000000
50%	13.000000	2.000000	6.000000	20.000000	0.000000
75%	108.000000	2.000000	10.000000	36.000000	2.000000
max	898.000000	9.000000	63.000000	378.000000	2.000000

Luego analizaremos cuantos resultados tenemos de cada tipo usando la función groupby y vemos que tenemos 86 usuarios “Clase 0”, es decir Windows, 40 usuarios Mac y 44 de Linux.

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

```
clase
0    86
1    40
2    44
dtype: int64
```

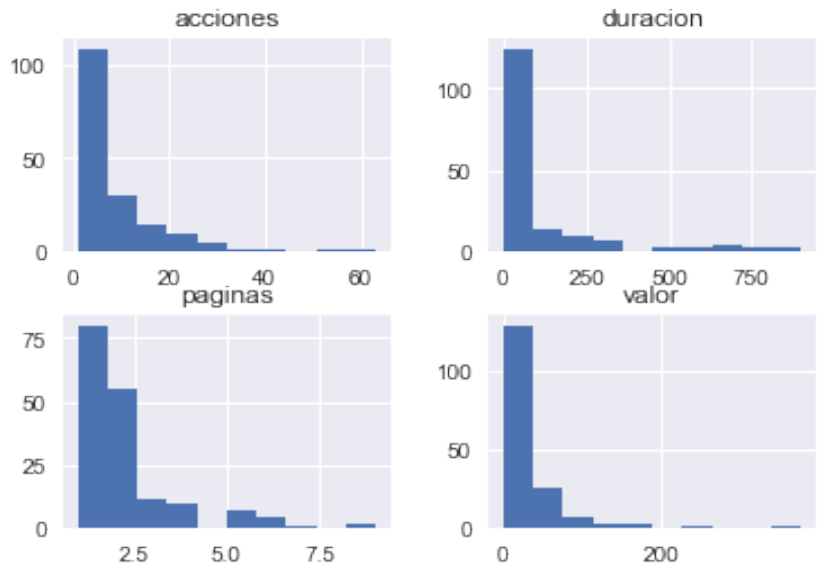
Regresión logística
Logistic regression

Visualización de Datos

Antes de empezar a procesar el conjunto de datos, vamos a hacer unas visualizaciones que muchas veces nos pueden ayudar a comprender mejor las características de la información con la que trabajamos y su correlación.

Primero visualizamos en formato de historial los cuatro Features de entrada con nombres “duración”, “páginas”,”acciones” y “valor” podemos ver gráficamente entre qué valores se comprenden sus mínimos y máximos y en qué intervalos concentran la mayor densidad de registros.

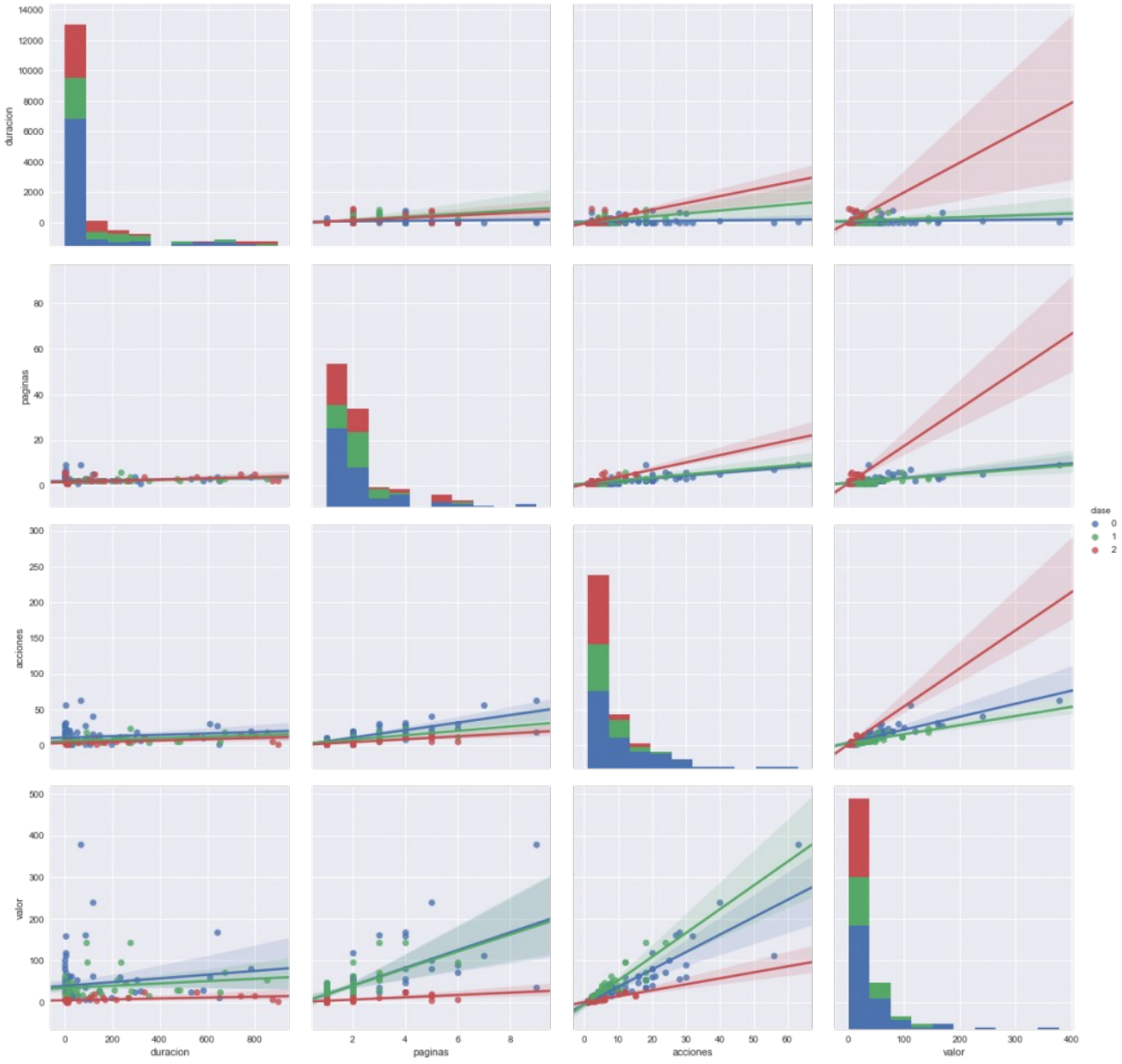
```
dataframe.drop(['clase'],1).hist()
plt.show()
```



Y también podemos interrelacionar las entradas de a pares, para ver como se concentran linealmente las salidas de usuarios por colores: Sistema Operativo Windows en azul, Macintosh en verde y Linux en rojo.

Regresión logística
Logistic regression


```
sb.pairplot(dataframe.dropna(), hue='clase',size=4,vars=["duracion",
"paginas","acciones","valor"],kind='reg')
```



Regresión logística
Logistic regression

Creamos el Modelo de Regresión Logística

Ahora cargamos las variables de las 4 columnas de entrada en X excluyendo la columna “clase” con el método drop(). En cambio agregamos la columna “clase” en la variable y.

Ejecutamos X.shape para comprobar la dimensión de nuestra matriz con datos de entrada de 170 registros por 4 columnas.

```
X = np.array(dataframe.drop(['clase'],1))
y = np.array(dataframe['clase'])
X.shape
```

(170, 4)

Y creamos nuestro modelo y hacemos que se ajuste (fit) a nuestro conjunto de entradas X y salidas ‘y’.

```
model = linear_model.LogisticRegression()
model.fit(X,y)
```

p class=””>LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class=’ovr’, n_jobs=1, penalty=’l2′, random_state=None, solver=’liblinear’, tol=0.0001, verbose=0, warm_start=False)

Una vez compilado nuestro modelo, le hacemos clasificar todo nuestro conjunto de entradas X utilizando el método “predict(X)” y revisamos algunas de sus salidas y vemos que coincide con las salidas reales de nuestro archivo csv

```
predictions = model.predict(X)
print(predictions)[0:5]
```

[2 2 2 2 2]

Regresión logística
Logistic regression

Y confirmamos cuan bueno fue nuestro modelo utilizando model.score() que nos devuelve la precisión media de las predicciones, en nuestro caso del 77%.

```
model.score(X,y)
```

0.77647058823529413

Validación de nuestro modelo

Una buena práctica en Machine Learning es la de subdividir nuestro conjunto de datos de entrada en un set de entrenamiento y otro para validar el modelo (que no se utiliza durante el entrenamiento y por lo tanto la máquina desconoce). Esto evitará problemas en los que nuestro algoritmo pueda fallar p or “sobregeneralizar” el conocimiento.

Para ello, subdividimos nuestros datos de entrada en forma aleatoria (mezclados) utilizando 80% de registros para entrenamiento y 20% para validar.

```
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation =
model_selection.train_test_split(X, y, test_size=validation_size,
random_state=seed)
```

Regresión logística
Logistic regression

Volvemos a compilar nuestro modelo de Regresión Logística pero esta vez sólo con 80% de los datos de entrada y calculamos el nuevo scoring que ahora nos da 74%.

```
name='Logistic Regression'
kfold = model_selection.KFold(n_splits=10, random_state=seed)
cv_results = model_selection.cross_val_score(model, X_train, Y_train,
cv=kfold, scoring='accuracy')
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
print(msg)
```

Logistic Regression: 0.743407 (0.115752)

Y ahora hacemos las predicciones -en realidad clasificación- utilizando nuestro “cross validation set”, es decir del subconjunto que habíamos apartado. En este caso vemos que los aciertos fueron del 85% pero hay que tener en cuenta que el tamaño de datos era pequeño.

```
predictions = model.predict(X_validation)
print(accuracy_score(Y_validation, predictions))
```

0.852941176471

Finalmente vemos en pantalla la “matriz de confusión” donde muestra cuantos resultados equivocados tuvo de cada clase (los que no están en la diagonal), por ejemplo predijo 3 usuarios que eran Mac como usuarios de Windows y predijo a 2 usuarios Linux que realmente eran de Windows.

Regresión logística
Logistic regression

Reporte de Resultados del Modelo

```
print(confusion_matrix(Y_validation, predictions))
```

```
[[16  0  2]
 [ 3  3  0]
 [ 0  0 10]]
```

También podemos ver el reporte de clasificación con nuestro conjunto de Validación.

En nuestro caso vemos que se utilizaron como “soporte” 18 registros windows, 6 de mac y 10 de Linux (total de 34 registros). Podemos ver la precisión con que se acertaron cada una de las clases y vemos que por ejemplo de Macintosh tuvo 3 aciertos y 3 fallos (0.5 recall). La valoración que de aqui nos conviene tener en cuenta es la de F1-score, que tiene en cuenta la precisión y recall. El promedio de F1 es de 84% lo cual no está nada mal.

```
print(classification_report(Y_validation, predictions))
```

```
             precision    recall  f1-score   support

          0       0.84      0.89      0.86        18
          1       1.00      0.50      0.67         6
          2       0.83      1.00      0.91        10

avg / total       0.87      0.85      0.84        34
```

Regresión logística
Logistic regression

Clasificación (o predicción) de nuevos valores

Como último ejercicio, vamos a inventar los datos de entrada de navegación de un usuario ficticio que tiene estos valores:

Tiempo Duración: 10
Paginas visitadas: 3
Acciones al navegar: 5
Valoración: 9

Lo probamos en nuestro modelo y vemos que lo clasifica como un usuario tipo 2, es decir, de Linux

```
X_new = pd.DataFrame({'duracion': [10], 'paginas': [3], 'acciones': [5], 'valor': [9]})
model.predict(X_new)
```

Conclusiones

Durante este artículo vimos cómo crear un modelo de Regresión Logística en Python para poder clasificar el Sistema Operativo de usuarios a partir de sus características de navegación en un sitio web. A partir de este ejemplo, se podrá extender a otro tipos de tareas que pueden surgir durante nuestro trabajo en el que deberemos clasificar resultados en valores discretos. Si tuviéramos que predecir valores continuos, deberemos aplicar Regresión Lineal.

Espero que puedan seguir correctamente el ejercicio y si tienen inconvenientes técnicos me escriben en los comentarios para solucionarlos o si encuentran mejoras al texto/ejercicio, también espero sus sugerencias.

Regresión logística
Logistic regression