Aprendizaje Automático 1

Práctica 2 - Regresión Lineal

Comienzo: 5 octubre (grupo jueves), 6 octubre (grupo viernes)
Entrega: a través de Moodle. 31 octubre (grupos de jueves y viernes)

Semana 1

Vamos a empezar utilizando una versión preprocesada del dataset que usamos en la práctica 1 (dataset de propiedades residenciales). El objetivo ahora es desarrollar un modelo lineal con un comportamiento lo mejor posible en términos del coeficiente de determinación R2. Para ello usaremos como datos de entrenamiento los datos del fichero housing_preprocesado_train.csv, y como test los datos del fichero housing_preprocesado_test.csv.

Para conseguir el mejor resultado hay que tener en cuenta los siguientes pasos:

Paso 1: Eliminación de variables colineales

- Si las variables de entrada al modelo son linealmente dependientes, la matriz de covarianza no es invertible. Como consecuencia, el problema de regresión por mínimos cuadrados no está bien definido.

- Para eliminar variables colineales puedes guiarte usando la matriz de correlación. ¿Qué variables eliminarías usando este método?

- Una manera de detectar cuántas variables independientes hay es calculando el rango de la matriz de covarianza:

```
import numpy as np
np.linalg.matrix_rank( X_train.cov() )
```

- Haz pruebas para ver que este método funciona: inventa un dataset con 10 filas y 2 columnas de datos donde una sea el doble que la otra. Chequea que el rango es 1. Haz el mismo análisis donde uno de los elementos de la segunda columna no sea el doble de la primera. Chequea que el rango ahora es 2. Define ahora otra tercera columna que sea la suma de las dos primeras. Chequea que el rango sigue siendo 2. Finalmente, genera una matriz aleatoria de datos de N filas y M columnas (N>M) con:

```
df = pd.DataFrame(np.random.rand(N,M))
```

  Chequea que para diferentes N y M, el rango es M. ¿Tiene sentido?

- Implementa el siguiente algoritmo para reducir automáticamente el dataset a un conjunto de variables independientes (el algoritmo está especificado con pseudocódigo):

    1. cols = nombres columnas dataset original
    2. rank = rango matriz covarianza training usando cols
    3. por cada columna col en cols:
       cols2 = cols eliminando col
       rank2 = rango matriz covarianza training usando cols2
       si rank = rank2 hacer cols = cols2 y empezar desde 3 de nuevo
    4. Simplificar training y test quedándonos solo con las columnas cols

- ¿Cuántas variables de entrada al modelo te quedan finalmente? Chequea que efectivamente son independientes (número variables = rango matriz de covarianza).

Paso 2: Construcción de un primer modelo

- Construye un modelo de regresión lineal usando sklearn y calcula su coeficiente de determinación R2 en training y en test. ¿Hay sobreajuste?

- Crea un scatter plot que enfrente el precio real de la casa con el precio estimado del modelo para los datos de training. Haz lo mismo para test. ¿Qué conclusiones extraes?

- Imprime por pantalla los coeficientes del modelo lineal ajustado. Nota: si model es el modelo ajustado, los coeficientes del modelo son model.coef_ y el término independiente es model.intercept_

- Calcula las predicciones de tu modelo usando esos coeficientes (sin usar model.predict) y chequea que coinciden exactamente con las dadas por model.predict

- ¿Se cumple la suposición de que los residuos siguen una distribución gaussiana?

- A continuación vamos a explorar el efecto de transformar de manera no lineal el target usando la transformación logarítmica. Primero haz un histograma del target sin transformar, y compáralo con el transformado. ¿Qué observas?

- A continuación crea un modelo lineal que prediga esta variable target transformada. Las predicciones del modelo tendrán que ser convertidas de nuevo a dólares usando la transformada exponencial (np.exp) ya que e**l R2 siempre hay que medirlo en la escala original de la variable target (dólares).** ¿Ha mejorado tu modelo? Nota: ahora no podrás calcular el R2 de tu modelo usando model.score ya que entonces medirías el R2 del logaritmo del precio. Lo haremos de la siguiente forma: después de aplicar la transformada exponencial a las predicciones, usa la función r2_score: from sklearn.metrics import r2_score.

Semana 2

Paso 1: Construcción de un modelo lineal con librería statsmodels

- Construye un modelo de regresión para el dataset anterior usando la librería statsmodels. ¿Obtiene los mismos coeficientes que sklearn? ¿Los coeficientes de determinación son iguales? ¿Es el QQ-plot el esperado para un modelo de regresión lineal?
- Saca por pantalla la información estadística del modelo: desviación estándar de los coeficientes, valores p, intervalos de confianza de los coeficientes. Con una significancia estadística del 95%, ¿qué variables parece que no aportan información?

Paso 2: Selección de variables con statsmodels

- Implementa el siguiente algoritmo de eliminación de variables usando statsmodels:
  1- Construye modelo con variables cols
  2- Detecta qué variables parece que no aportan información de acuerdo al criterio de los valores P. Si no encuentras ninguna, fin del algoritmo. Si encuentras una o varias, toma la que tenga un valor P mayor y quítala de cols. Vuelve a 1

- ¿Con cuántas variables te quedas usando este algoritmo? ¿Qué R2 tiene en training y en test el nuevo modelo? ¿Es mejor que el modelo sin seleccionar variables?

Paso 2: Selección de variables con regularización

- Crea un modelo lineal usando regresión Lasso usando la clase LassoCV del módulo linear_model de sklearn. ¿Qué variables tienen peso 0 en el modelo construido? ¿es esto consistente con lo que has visto en el paso 1? ¿Qué R2 en training y test tiene tu modelo? ¿es mejor respecto a los anteriores?
  Importante: para usar regresión Lasso o Ridge, las variables de entrada al modelo y la variable target se deben estandarizar.

- Repite el punto anterior usando regresión Ridge

Semana3

Ahora vamos a utilizar todo lo aprendido en la práctica 1 y en la parte 1 de esta práctica para realizar un modelo lineal para predecir de manera lo más precisa posible la afluencia de público a un evento deportivo (partidos de béisbol). Para ello nos proporcionan una base de datos, **baseball_construccion.csv**, que contiene los datos registrados a lo largo de la primera parte de una temporada.

Las columnas del fichero son:

- **year** : año del evento
- **home_team** : equipo local
- **month, day** : mes y día del mes del evento
- **attend** : número de personas que acudió al evento
- **day_of_week** : día de la semana
- **temp** : temperatura ese día
- **skies** : clear/cloudy/rainy (cielo despejado / nuboso / lluvioso)
- **day_night** : si el evento se produce por el día o por la noche
- **cap** : promoción gorras
- **shirt** : promoción camisetas
- **fireworks** : si ese día hay fuegos artificiales en el evento
- **bobblehead** : si ese día regalan a la entrada muñequitos de los jugadores
- **hum>average** : si la humedad está por encima de la media
- **attend_inc** : incremento de espectadores respecto al último partido del equipo local en ese estadio
- **attend_thousands** : número de espectadores en millares

Se nos proporciona el conjunto baseball**_explotacion.csv** con los datos de los 8 últimos partidos de cada equipo local, y sobre los que tenemos que predecir el número de personas que acudirán al estadio.

Nos piden que construyamos un fichero con las predicciones en las que **en cada línea haya un solo número: la predicción para la correspondiente línea del fichero de explotación.**

La calidad de nuestro modelo se va a medir por el error relativo medio, es decir:

$$Error = 100 \cdot \frac{1}{N_{test}} \cdot \sum_i \frac{abs\,(attend\_pred_i - attend_i)}{attend_i}$$

Siendo "attend_i" el número de personas que van realmente al partido i, y "attend_pred_i" nuestra estimación.

ENTREGABLES

- Fichero JupyterNotebook con el código y las respuestas a las preguntas planteadas en el dataset de casas (semanas 1 y 2). En la corrección se tendrá muy en cuenta la argumentación en las respuestas. Se subirá un solo fichero por pareja de prácticas con la nomenclatura
  AA1_P2-12_<grupo>_<pareja>.ipynb (ejemplo: AA1_P2-12_Jueves_1.ipynb)

- Fichero JupyterNotebook con el código y las respuestas a las preguntas planteadas en el dataset de béisbol (semana 3). En la corrección se tendrá muy en cuenta la argumentación en las respuestas. Se subirá un solo fichero por pareja de prácticas con la nomenclatura
  AA1_P2-3_<grupo>_<pareja>.ipynb (ejemplo: AA1_P2-3_Jueves_1.ipynb)

- Fichero predicciones.csv con las predicciones de aforo para cada uno de los partidos del fichero baseball_explotacion.csv. El fichero predicciones.csv tiene solo una columna, de título “attend”, y tantas filas como tiene baseball_explotacion.csv. Como ejemplo se deja el fichero ejemplo_predicciones.csv