CI-0148 Aprendizaje Máquina
Prof. Pablo Sauma Chacón

Laboratorio 3

Objetivo: Familiarizar al estudiante con la creación y el uso de regresiones lineales. Así como las diferentes métricas que existen para medir el rendimiento de los modelos implementados.

Enunciado: Lea las instrucciones del documento y resuelva el enunciado en Python. Al inicio de su documento adjunte su nombre y carnet como un comentario. Este trabajo es de carácter individual, tampoco se aceptarán códigos que no hayan sido desarrollados por su persona.

Para este laboratorio se trabajará con el set de datos: fish_perch.csv que consiste de datos estrictamente numéricos. El objetivo de este set de datos es predecir el peso `weight` de cada pez en función de sus otros atributos: `Length1`, `Length2`, `Length3`, `Width` y `Height`.

1. Función `MSE(y_true, y_predict)` que recibe dos objetos `pd.Series` que contienen los valores reales de un conjunto de datos y los valores estimados por un modelo. Calcule y retorne el error cuadrático medio de dicha predicción.

2. Función `score(y_true, y_predict)` que recibe dos objetos pd.Series que contienen los valores reales de un conjunto de datos y los valores estimados por un modelo. Calcule y retorne el coeficiente de determinación (R^2) de dicha predicción.

También deberá implementar la clase **`LinearRegression`** con los siguientes métodos:

3. Método `fit(self, x, y, max_epochs=100, threshold=0.01, learning_rate=0.001, momentum=0, decay=0, error='mse', regularization='none', lambda=0)` que recibe una variedad de parámetros:
    a. Un objeto `pandas.DataFrame` `x` que contiene los datos para los que se desea hacer la regresión lineal
    b. Un objeto `pandas.Series` `y` que contiene los valores reales de y para cada variable de `x`
    c. Una variable opcional `max_epochs` con valor por defecto `100`, que indica la cantidad de iteraciones máxima para la regresión
    d. Una variable opcional `threshold` con valor por defecto `0.01`, que indica el umbral de cambio mínimo requerido para continuar con la regresión (si el cambio del error entre dos iteraciones es menor al umbral, se finaliza)
    e. Una variable opcional `learning_rate` con valor por defecto `0.001`, que indica la tasa de aprendizaje del algoritmo de regresión
    f. Una variable opcional `momentum` con valor por defecto `0`, que indica el momentum o la inercia del algoritmo de aprendizaje
    g. Una variable opcional `decay` con valor por defecto `0`, que indica la tasa de decaimiento de la tasa de aprendizaje

h. Una variable opcional `error` con valor por defecto `'mse'`, que indica la función de error que se utilizará para calcular el error de la estimación; la otra opción válida es `'mae'`
i. Una variable opcional regularization con valor por defecto `'none'`, indica la regularización a utilizar en la regresión, las otras opciones válidas son `'l1'` o `'lasso'` (son sinónimos); y `'l2'` o `'ridge'`.
j. Una variable opcional `lambda` con valor por defecto `0`, que indica la tasa de regularización a aplicar en el algoritmo.
k. Note que este método posee muchas opciones que afectarán el resultado obtenido, sin embargo también note que muchas de estas variables no tienen ningún efecto en su valor por defecto (recalcular eta si se posee un decaimiento en 0, producirá eta). No deje que los requerimientos lo abrumen, es recomendable empezar desde la versión trivial del algoritmo e ir agregando las opciones después

4. Método `predict(self, x)` que recibe un objeto `pandas.DataFrame` `x` que contiene los datos para los que se desea hacer su estimación. El método debe retornar un objeto de tipo `pd.Series` con los valores estimados para cada uno de los valores.
    a. Este método será llamado después de `fit`, punto en el cuál ya debería contarse con los valores de C/los pesos necesarios para hacer la estimación.

5. Utilice el set de datos proveído para probar el funcionamiento de su algoritmo. Recuerde que el error debe reducirse en cada iteración del algoritmo (o llegar a un "zig-zag" producto de una tasa de aprendizaje muy elevada). Luego utilice el método `train_test_split` de la biblioteca `sklearn.model_selection` para separar un conjunto de datos en un conjunto de datos de entrenamiento y otro de prueba, utilice de semilla del split el número 21 (el método permite el parámetro opcional `random_state` para sembrar la aleatoriedad).
    a. Para cada prueba calcule e imprima el error R^2
    b. Realice la prueba con múltiples combinaciones de parámetros, intentando mejorar la estimación
    c. ¿Cuál fue la combinación de parámetros que le proveyó el mejor resultado?
    d. ¿Qué pasa si utiliza esa misma combinación pero cambia la semilla del `train_test_split`? Pruebe con varias semillas
    e. Si pasa algo inusual: ¿Por qué cree que pasa esto?