Predictor Temperatura

Grados Farenheit en actual

Todas las temperaturas del anio 2016

week\_day en string

Nota de color: friend

348 filas -> Falta algún día

describe-> Escalas distintas -> Pero DecisionTree no afecta

DataPreparation

week\_int

week\_int -> LabelEncoder asigna un número sin orden

OneHotEncoder -> Demasiado sparse? Sería lo adecuado.

Convertimos Lunes -> 1, … Sun->7

Pero eso querría decir que esperamos una relación “creciente”, “decreciente” porque del 1 al 7 tienen un orden

Creamos las estaciones como one hot encoder

month -> Obviamente tenemos una relación. Problema -> correlación 0 por la forma

Modificamos month artificialmente -> Ahora tenemos una relación creciente con correlación 0.68, pero esto no es ideal ni de cerca

Lo mejor es hacer lo siguiente: la temperatura media del mes Sí tiene una relación con la temperatura. ASÍ DESCARTAMOS EL NÚMERO DEL MES.

Se podría hacer más facil con groupby transform -> Correlación de 0.88

SELECCIÓN DE VARIABLES

temp\_2, temp\_1 (2 días antes y un día antes) -> SerieTemporal con shift.

self\_month con actual correlación del 0.68, pero es un tanto artificial

avg\_temp\_month -> Ésta promete

average -> Pero correlación alta con avg\_temp\_month. 0.95 Probablemente sea la misma variable calculada de forma un poco distinta.

Dejamos fuera

forecast\_noaa

forecast\_acc

Correlación del 0.99 misma variable con average

Obviamos la alta correlación entre average y avg\_temp\_month

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

En cualquier caso, se puede hacer el fit con distintos sets de variables y comprobar cúal nos da las mejores predicciones

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Discusión: que pasa si tenemos una clasificación como la del dataset de Iris. Es relevante la correlación? Setosa va antes que virgínica? La matriz de correlación no es lo ideal para seleccionar varia

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Train Test Split

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Fit unsconstrained**

R2 train= 1.0 -> Overfitting

R2 test = 0.72 -> Overfitting

MAE test= 4.8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**max\_depth = 3**

R2 train 0.8397458755018086

R2 test 0.7786776796296856

MAE 4.267242249556827

Buen modelo y además representable white\_box

Feature\_importances:

temp\_1 = 0.88

average = 0.11

El modelo parece haber elegido average en lugar de avg\_temp\_month

(recordad la sensibilidad a pequeños cambios: podría haber sido al revés?)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

GridSearch en Train

parameters = {

"max\_depth" : [2,3,4,5],

"min\_samples\_leaf": [3,5,7],

}

Best params

{'max\_depth': 4, 'min\_samples\_leaf': 3}

MAE cv en train: np.float64(-4.108918381420885)

MAE test : 4.050098189195934

No hay overfitting! MAEtrain = MAEtest!!!!

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Probamos con min\_samples\_leaf = 5 y lo entrenamos en set de Train completo.

Nos da un MAE prácticamente igual en test 4.05. Parece no afectar este cambio.

El MAE en FULL dataset sube ligeramente con cross\_validation a 4.6. Mayor que el cv en train. Extraño, pero válido.

***GridSearchCV*** *no entrena automáticamente el best\_estimator\_ en el dataset completo (train en este caso) después de encontrar los mejores hiperparámetros. Hay que hacerlo manualmente.*

Nos quedamos con este modelo porque generalizará mejor (más constrained)

temp\_1 = 0.86

average = 0.13

CUALQUIERA DE LOS DOS MODELOS NOS SERVIRÍA COMO MODELO FINAL.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DecisionTreeClassifier

Crear el target 59F->15C y además categórica.

Target balanceado. 60,40. Qué nos interesa? Los 0s, los 1s. Uno diría que ambos por igual pero eso no es cierto…

ANECDOTA 60% Probabilidad para que la gente no se enfade.

Ya conocemos nuestras variables predictoras así que directamente train\_test split.

parameters = {

"max\_depth" : [2,3,4,5],

"min\_samples\_leaf": [3,5,7],

}

Best params: {'max\_depth': 3, 'min\_samples\_leaf': 5}

Best score: np.float64(0.892077922077922)

Reentrenamos en Train con esos parámetros.

Predecimos en test:

**Accuracy 0.9142857142857143 -> No hay overfitting.**

AUC 0.8988095238095237

Recall 0.9761904761904762

Precision 0.8913043478260869

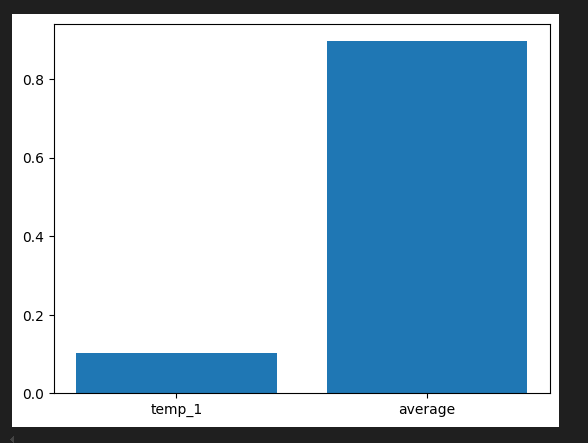
F1 Score 0.9318181818181818

Confusion matrix

[[23 5]

[ 1 41]]

Y sorpresa:



INESTABILIDAS A PEQUENIOS CAMBIOS!!!!! PARA EL Regressor era al revés.

Miramos la matriz de correlación y vemos que average tiene una correlación del 0.82 con temp\_1.

**CONCLUSIÓN. No tenemos un mal modelo pero tiene los problemas intrínsecos de los que hablamos ayer.**