Diabetes india

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

Clasificación class [0,1] (supervisado)

Target:

class

0 500

1 268

No muy desbalanceado 2:1

Qué nos interesa?

El recall: que no se nos escape ningún 1. FN-> 0.

768 filas

Aún así, para tener una idea general de los algoritmos vamos a utilizar el accuracy. Sabiendo que no es lo adecuado. Con fines didácticos.

Por ello vamos a buscar primero por accuracy y después por recall

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

Features

Pairplot:

preg, plas, mass,age

preg con age = 0.54

plas con test = 033

Las obviamos, la primera por trabajar con Ensembles tipo RandomForest (hablar colinealidades) y la segunda por no ser muy alta.

Nos fijamos en la diagonal pero ahora sí nos fijamos también en los cruces (porque DecisionTree puede hacer cortes en el x1, x2)

Matriz de correlación:

Correlación alta -> bueno, pero nos habla de relación lineal. Acordarse del ejemplo de los meses ayer. Un DecisionTree puede capturar una relación no lineal (una variable con 0 de correlación nos puede servir.

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

1

cross\_validation 10 con todo el dataset -> Buena estimación del error.

Bagging con DecisionTree max\_depth=5

0.7850820232399179

RandomForest max\_depth=5, estimators=100

0.7720437457279563

AdaBoost max\_depth=5, estimators=100

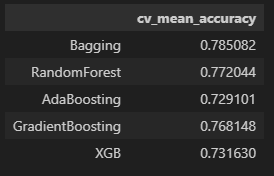
0.7291011619958988

GradientBoosting, estimators=100

0.7681476418318524

XGBoost, estimators=100

0.7316302118933699



Para saber si hay overfitting separar acc\_train, acc\_test

La desviación estándar no es un buen indicativo de overfitting. En cualquier caso nos interesa una buena estimación de la métrica, que es lo que conseguimos con cross\_val y nos permite compararlos.

2 Hiperparametrización de los mejores RandomForest y y GradientBoosting

(problema, podría haber otro algoritmo que mejor parametrizado tuviera mejor performance)

Procedimiento

GridSearch (accuracy por practicar, no sería una buena métrica)

BestEstimator

* best\_params que maximizan la métrica (accuracy)
* métrica con 10 folds en train -> buena estimación

RetrainModel on Train

Eval on Test -> mejor estimación

Cross\_val en full dataset del modelo reentrenado.

Train test split 0.2: train = 614,test =154

En este ejercicio no comparamos las métricas de train con las de test para identificar overfitting. Simplemente buscamos el mejor de ellos buscando un poco a ciegas (no malo necesariamente) o automáticamente (que suena mucho mejor)

RandomForest

{'max\_depth': 4, 'max\_features': 3, 'max\_leaf\_nodes': 12, 'n\_estimators': 80}

Acc: 0.7785552445688392 -> No mejora la anterior.

Retrain, eval on test

Acc: 0.7727272727272727 -> Idéntica como debería ser

Recall: 0.6545454545454545 -> Recall bajo, deberíamos hacer un GridSearch por recall

Acc cross\_val en X: 0.7695146958304854 -> Como debería ser. Sin sorpresas.

{'max\_depth': 5, 'max\_features': 3, 'n\_estimators': 100}

Acc: 0.7467532467532467 -> **baja el accuracy**

Recall: 0.7090909090909091 -> **sube el recall**

Acc cross\_val en X:

0.7720437457279563

0.07675482452844708

GradientBoosting

{'learning\_rate': 0.05, 'max\_depth': 2, 'n\_estimators': 50}

Acc: 0.7785419165667067

Retrain, eval on test

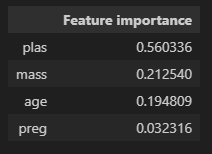
Acc 0.7662337662337663

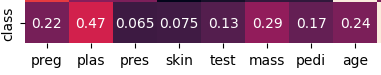
Recall: 0.5818181818181818

GridSearch por recall

Acc cross\_val en X: 0.7694805194805194

Elegimos RandomForest porque a pesar de tener la misma métrica en acc: 0.7695, tiene mejor recall (0.65 vs 0.58) es más rápido. Como hemos dicho que queríamos accuracy (a pesar de no ser lo adecuado). Además tenemos unas feature importances más solidas.





Preg, a pesar de tener un coeficiente de correlación equivalente a age, no contribuye.

DISCUSIÓN: Métrica? Método?