Facultad de Matemática, Astronomía y Física y Computación
Universidad Nacional de Córdoba

Ciencia de Datos

Práctico N°9: Métodos de Ensemble

Problema 1: Para la realización de este práctico resulta útil consultar la ipynb de Sebastian Raschka, pero se propone cambiar de dataset y usar en su lugar el *Breast Cancer Wisconsin* dataset. Con el fin de evaluar por separado los modelos que integran el emsemble del test del emsemble, separar el 25 % de los datos para test y a su vez dividir el conjunto de entrenamiento, separando de este un 25 % para validación de cada modelo.

Problema 2: Votación de la mayoría

a) Construir tres árboles de decisión con diferentes profundidades usando el parámetro `max_depth = 1,2,3`. Luego construir un emsemble con igual peso para esos árboles usando `EnsembleVoteClassifier()` de la librería `mlxtend.classifier`; o bien, `VotingClassifier` de `scikit-learn`.

b) Reportar la `accuracy` en la validación de cada modelo por separado y en el test del emsemble.

Problema 3: Bagging: Bootstrap Aggregating

Usando `BaggingClassifier()`, construir una *bolsa* de 500 árboles de decisión, usando todas las características, *bootstrap* con reemplazo y *out-of-bag samples* para estimar el error de generalización. Reportar *accuracy out-of-bag* (OOB) y sobre el set de test.

Problema 4: Adaptive Boosting (Adaboost)

Utilizar 500 *tree stumps* (`max_depth=1`) para implementar `AdaBoostClassifier()` con el algoritmo `SAMME.R`, usando todas las variables. Reportar *accuracy* sobre el set de test.

Problema 5: Gradient Boosting

Implementar `GradientBoostingClassifier()` con los valores default de los parámetros y estudiar el impacto de los parámetros `n_estimators, learning_rate, max_depth` en *accuracy*. ¿En qué casos se recomienda usar `HistGradientBoostingClassifier()`?

Problema 6: Random Forests

Implementar `RandomForestClassifier()` y estudiar el significado de los parámetros `max_depth`, `max_features`, `min_samples_leaf` y `min_samples_split`.