Ciencia de Datos

Práctico N°7: Árboles de Decisión

Problema 1: Al ingresar al bar el protagonista de este problema, encuentra que sus cinco amigos con quienes se reune los viernes al salir de trabajar, ya tienen sus bebidas sobre la mesa. María que tiene un cargo de gerente paga la ronda la mitad de las veces. Pablo paga la ronda la cuarta parte de las veces, mientras que Sara y Carlos, que son becarios, pagan indistintamente entre ambos la octava parte de las veces. Juan nunca sacó la billetera desde que se reunen los viernes.
a) ¿Qué fracción de las veces el protagonista paga la ronda?
b) Calcular la entropía de la distribución de probabilidad con la que cada uno paga la ronda. ¿Cuál es el número medio de preguntas (de respuesta binaria) que necesitan hacerse en promedio para saber quién paga la ronda?

c) Poco después de arribar al bar se suman dos antiguos amigos quienes no vivieron en la ciudad el último año. Ellos deciden que la próxima ronda la debe pagar el protagonista y conociendo que cursa esta materia, lo desafían a predecir que bebida tomará cada uno.
Con la información de las tres variables binarias: sexo, si es o no estudiante y si le gusta o no bailar, y recordando la elección de bebidas de la noche anterior, puede construirse la siguiente tabla:

Bebida	Sexo	Estudiante	Baile
cerveza	M	T	T
cerveza	M	F	T
vodka	M	F	F
vodka	M	F	F
vodka	F	T	T
vodka	F	F	F
vodka	F	T	T
vodka	F	T	T

Usando entropía de información entrenar un árbol de decisión *con lápiz y papel* a partir de la tabla anterior. Registrar todos los valores calculados para elegir las variables en cada nodo del árbol.
d) Proponer una poda posible del árbol y calcular la nueva *accuracy* resultante.

Problema 2: Estudiar la implementación de Árboles de Decisión para clasificación provistas por scikit-learn.
a) ¿Qué algoritmo de árboles de decisión está implementado en scikit-learn? ¿Qué nombre tiene el modelo? ¿Cuál es su principal limitación?
b) Identificar cuál opción del parámetro *criterion* se corresponde con la entropía de Shannon y cuál con Gini como medidas de impureza. ¿Qué parámetros controlan el tamaño del árbol y el *prunning*?

Problema 3: Entrenar un árbol de decisión usando el iris dataset (usar todos los datos y variables). Emplear entropía de Shannon y luego Gini.
a) Graficar los arboles resultantes usando la información disponible en la entrada de scikit-learn.
b) Identificar la variable utilizada como raíz y el valor de corte resultante en cada caso. Interpretar a partir del conocimineto adquirido sobre este dataset.

c) Utilizando la información disponible en Plot the decision surface of decision trees trained on the iris dataset, estudiar las fronteras de decisión bidimensionales generadas por el árbol de decisión aplicado sólo sobre las variables de sépalo del iris dataset.

Problema 4: Para evitar que un árbol se ajuste en exceso (overfitting), `DecisionTreeClassifier` proporciona los parámetros `min_samples_leaf` y `max_depth`. La poda usando una función de costo de complejidad proporciona otra opción para controlar el tamaño de un árbol. Esta técnica de poda está regulada por el parámetro `ccp_alpha`. Los valores más altos de `ccp_alpha` aumentan el número de nodos podados.
Con la información provista en Post pruning decision trees with cost complexity pruning, usar el Breast cancer Wisconsin dataset provisto por scikit-learn para:
a) graficar la impureza total de las hojas vs. `ccp_alpha`.
b) graficar el número de nodos y profundidad del árbol vs. `ccp_alpha`.
c) graficar la *accuracy* de clasificación sobre los conjuntos de training y testing vs. `ccp_alpha`.

Métodos de Ensemble

Problema 5: Para la realización de esta sección 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 ensemble del test del ensemble, 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 6: Votación de la mayoría
a) Construir tres árboles de decisión con diferentes profundidades, tomando los valores de parámetro `max_depth = 1,2,3`. Luego construir un ensemble 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 ensemble.

Problema 7: 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 8: Adaptive Boosting (Adaboost)
Definir el clasificador AdaBoost usando todas las variables siguiendo los pasos

```
Adaboost instructions
```

y aplicarlo al *Breast Cancer Wisconsin* dataset.

Problema 9: 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 10: Random Forests

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

Problema 11: Genere una tabla donde presente las métricas estudiadas para clasificación (clase VIII) para cada uno de los modelos desarrollados en los ejercicios 6, 7, 8 y 9.