

Joy Steven Castañeda Mancera - 63907

Siniestros viales

Objetivo Laboratorio

Mejorar cualquiera de las siguientes métricas de desempeño de los modelos de Árboles de decisión, Random Forest y Xgboost:

- Precisión
- Recall □ F1-score

Desarrollo Laboratorio

Modelo Árboles de decisión

Código

```
#Definición del modelo
arbol1 = DecisionTreeClassifier()
#arbol1 = GridSearchCV(dt, param_grid, cv=5, scoring='f1')
#Entrenamiento y evaluación del modelo
arbol1 = arbol1.fit(X_train,y_train)

# Calcular métricas de desempeño
y_pred = arbol1.predict(X_test)
print("\n", metrics.classification_report(y_test, y_pred, digits=2))

# Visualizar matriz de confusión
# Y_pred13 = np_utils.to_categorical(y_pred13)
# cm = matriz_confusion(Y_test, y_pred, 'si', 'Matriz de confusión clasificador AD')
```

Resultados

	precision	recall	f1-score	support
0	0.53	0.55	0.54	20310
1	0.06	0.07	0.06	972
2	0.77	0.75	0.76	38462
accuracy			0.67	59744
macro avg	0.45	0.45	0.45	59744
weighted avg	0.67	0.67	0.67	59744

Procesos Realizados

Se realiza pruebas en la definición del modelo para mejorar los valores de precisión, recall y f1-score y se establece lo siguiente:

- Se establece el parámetro de `max_depth` en 10 para limitar la profundidad máxima del árbol de decisión, para prevenir el sobreajuste y mejorar la generalización del modelo.
- Se establece el parámetro de `criterion` en `gini`, ya que se determinó que en el modelo era mejor que `entropy` ya que mide la impureza de una partición. Cuanto menor sea el valor de Gini, mejor será la calidad de la partición.
- Se establece el parámetro de `random_state` en 42 para controlar la aleatoriedad del algoritmo.

Código Modificado

```
#Definición del modelo
arbol1 = DecisionTreeClassifier(
    max_depth      = 10,
    criterion      = 'gini',
    random_state   = 42
)
#arbol1 = GridSearchCV(dt, param_grid, cv=5, scoring='f1')
#Entrenamiento y evaluación del modelo
arbol1 = arbol1.fit(X_train,y_train)

# Calcular métricas de desempeño
y_pred = arbol1.predict(X_test)
print("\n", metrics.classification_report(y_test, y_pred, digits=2))

# Visualizar matriz de confusión
# Y_pred13 = np_utils.to_categorical(y_pred13)
# cm = matriz_confusion(Y_test, y_pred, 'si', 'Matriz de confusión clasificador AD')
```

Resultados Nuevos

	precision	recall	f1-score	support
0	0.84	0.42	0.56	28310
1	0.18	0.01	0.02	972
2	0.75	0.97	0.85	38462
accuracy			0.77	59744
macro avg	0.59	0.47	0.47	59744
weighted avg	0.77	0.77	0.74	59744

Se evidencia mejora en cuanto a el macro avg y weighted avg de precisión, recall y f1-score.

Modelo Random Forest

Código

```
#Definición del modelo
rfc = RandomForestClassifier(n_estimators=200)

#Entrenamiento y evaluación del modelo
rfc.fit(X_train, y_train)

# Calcular métricas de desempeño
y_pred2 = rfc.predict(X_test)
print("\n", metrics.classification_report(y_test, y_pred2, digits=2))
```

Resultados

	precision	recall	f1-score	support
0	0.78	0.45	0.57	20310
1	0.00	0.00	0.00	972
2	0.76	0.95	0.84	38462
accuracy			0.76	59744
macro avg	0.51	0.47	0.47	59744
weighted avg	0.76	0.76	0.74	59744

Procesos Realizados

Se realiza pruebas en la definición del modelo para mejorar los valores de precisión y se establece lo siguiente:

- Se establece el parámetro de max_depth en 20 para limitar la profundidad máxima del árbol de decisión, para prevenir el sobreajuste y mejorar la generalización del modelo.


Código Modificado

```
#Definición del modelo
rfc = RandomForestClassifier(n_estimators=200, max_depth=20)

#Entrenamiento y evaluación del modelo
rfc.fit(X_train, y_train)

# Calcular métricas de desempeño
y_pred2 = rfc.predict(X_test)
print("\n", metrics.classification_report(y_test, y_pred2, digits=2))
```

Resultados Nuevos



	precision	recall	f1-score	support
0	0.86	0.41	0.56	20310
1	0.00	0.00	0.00	972
2	0.75	0.98	0.85	38462
accuracy			0.77	59744
macro avg	0.54	0.46	0.47	59744
weighted avg	0.78	0.77	0.74	59744

Se evidencia mejora en cuanto a el macro avg y weighted avg de precisión

Modelo Xgboost

Código

▼ Xgboost

```
[62] from sklearn.model_selection import train_test_split
import xgboost as xgb
```



#Definición del modelo

```
xg_class = xgb.XGBClassifier(objective='multi:softprob', colsample_bytree=1, learning_rate=0.1,
                             max_depth=5, alpha=10, n_estimators=100)
```



```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y_train = le.fit_transform(y_train)
```

```
[74] #Entrenamiento del modelo
xg_class.fit(X_train, y_train,
             eval_set=[(X_train, y_train), (X_test, y_test)],
             eval_metric='mlogloss',
             verbose=False)
```



```
/usr/local/lib/python3.10/dist-packages/xgboost/sklearn.py:889: UserWarning: `eval_metric` in `fit` method is deprecated. Use `eval_metric` in `XGBClassifier` instead.
warnings.warn(
```



XGBClassifier

```
XGBClassifier(alpha=10, base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None, colsample_bytree=1,
              device=None, early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, feature_types=None, gamma=None,
              grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.1, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=10, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=200, n_jobs=None,
              num_parallel_tree=None, ...)
```

```
#Entrenamiento y validación cruzada mediante k-fold
scores = cross_val_score(xg_class, X_train, y_train, cv=5)
print("Mean cross-validation score: %.2f" % scores.mean())

# Calcular métricas de desempeño
y_pred4 = xg_class.predict(X_test)
print("\n", metrics.classification_report(y_test, y_pred4, digits=2))
```

Resultados

```
➔ Mean cross-validation score: 0.77

      precision    recall  f1-score   support

     0       0.88      0.41      0.56      20310
     1       0.00      0.00      0.00        972
     2       0.75      0.98      0.85     38462

 accuracy          0.77      59744
 macro avg       0.54      0.46      0.47      59744
 weighted avg    0.78      0.77      0.74      59744

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_class
```

Procesos Realizados

Se realiza pruebas en la definición del modelo para mejorar los valores de precisión y se establece lo siguiente:

- Se establece el parámetro de `max_depth` en 10 para limitar la profundidad máxima del árbol de decisión, para prevenir el sobreajuste y mejorar la generalización del modelo.
- Se establece el parámetro de `n_estimators` en 200 para maximizar el numero de arboles a construir, mejorando el rendimiento del modelo, pero aumentando el tiempo de entrenamiento del modelo

Código Modificado

```
#Definición del modelo
xg_class = xgb.XGBClassifier(objective='multi:softprob', colsample_bytree = 1, learning_rate = 0.1,
                             max_depth = 10, alpha = 10, n_estimators = 200)
```

Resultados Nuevos

```
mean cross validation score: 0.77
```

	precision	recall	f1-score	support
0	0.83	0.44	0.57	20310
1	1.00	0.00	0.00	972
2	0.76	0.97	0.85	38462
accuracy			0.77	59744
macro avg	0.86	0.47	0.47	59744
weighted avg	0.78	0.77	0.74	59744

Se evidencia mejora en cuanto a el macro avg y weighted avg de precisión.

Conclusión

El proceso de mejora y ajuste de un modelo implica encontrar un equilibrio óptimo entre varios parámetros para maximizar el rendimiento y la generalización. Las mejoras iterativas y la evaluación continua permiten desarrollar un modelo más preciso y robusto, capaz de manejar diferentes conjuntos de datos de manera efectiva.