Metricas por modelo

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score, StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, roc_auc_score
from sklearn.feature_extraction.text import TfidfVectorizer
import numpy as np

# Carga de datos
data = pd.read_csv('dataset.csv')
data_clean = data.drop('sha256', axis=1).fillna('')
text_data = data_clean.drop('labels', axis=1).apply(lambda x: ' '.join(x), axis=1)
labels = data_clean['labels']
```

```
df = pd.read_csv('./dataset.csv')
df.head()
```

	sha256	labels	0	1	2
0	5c18291c481a192ed5003084dab2d8a117fd3736359218...	0	LdrUnloadDll	CoUninitialize	N
1	4683faf3da550ffb594cf5513c4cbb34f64df85f27fd1c...	0	NtOpenMutant	GetForegroundWindow	N
2	9a0aea1c7290031d7c3429d0e921f107282cc6eab854ee...	0	GetForegroundWindow	DrawTextExW	G
3	e0f3e4d5f50afd9c31e51dd9941c5a52d57c7c524f5d11...	0	NtQueryValueKey	LdrUnloadDll	GI
4	ec2b6d29992f13e74015ff0b129150b4afae15c593e4b7...	0	LdrUnloadDll	GetSystemTimeAsFileTime	N

5 rows × 177 columns

```
data = pd.read_csv('dataset.csv')
data_clean = data.drop('sha256', axis=1).fillna('')
text_data = data_clean.drop('labels', axis=1).apply(lambda x: ' '.join(x), axis=1)
labels = data_clean['labels']
```

```
# Vectorización
tfidf_vectorizer = TfidfVectorizer(max_features=1000)
X = tfidf_vectorizer.fit_transform(text_data)
y = labels

# División de datos
```

```
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42

    # Modelos
    rf_model = RandomForestClassifier(random_state=42)
    mlp_model = MLPClassifier(random_state=42, max_iter=100)

    # Validación cruzada
    kf = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
    rf_cv_scores = cross_val_score(rf_model, X_train, y_train, cv=kf, scoring='accuracy')
    mlp_cv_scores = cross_val_score(mlp_model, X_train, y_train, cv=kf, scoring='accuracy')

    # Entrenamiento
    rf_model.fit(X_train, y_train)
    mlp_model.fit(X_train, y_train)

    # Función de evaluación
    def evaluate_model(model, X_test, y_test):
        y_pred = model.predict(X_test)
        y_prob = model.predict_proba(X_test)[:, 1]  # probabilidades para la clase positiva
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred)
        recall = recall_score(y_test, y_pred)
        roc_auc = roc_auc_score(y_test, y_prob)
        return accuracy, precision, recall, roc_auc

    # Evaluación
    rf_metrics = evaluate_model(rf_model, X_test, y_test)
    mlp_metrics = evaluate_model(mlp_model, X_test, y_test)
```

```
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
```

```
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
c:\Users\aleja\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
```

```python
resultados = {
    'Métrica': ['CV Precisión Promedio', 'Precisión (Test)', 'Recall (Test)', 'Precisión
    'Random Forest': [np.mean(rf_cv_scores), rf_metrics[0], rf_metrics[1], rf_metrics[2]
    'Red Neuronal Multicapa': [np.mean(mlp_cv_scores), mlp_metrics[0], mlp_metrics[1], m
}

df_resultados = pd.DataFrame(resultados)

# Imprimir la tabla de resultados
print(df_resultados)
```

```
                 Métrica  Random Forest  Red Neuronal Multicapa
0  CV Precisión Promedio       0.953302                0.951642
1       Precisión (Test)       0.962387                0.952010
2          Recall (Test)       0.994536                0.946970
3       Precisión (Test)       0.930946                0.959079
4             AUC (Test)       0.989268                0.983050
```

Metricas por modelo

Random Forest: - CV Precisión Promedio (0.953302): El modelo generaliza bien con datos no vistos. - Precisión en el Test (0.962387): Alta exactitud en la clasificación de las muestras. - Recall en el Test (0.994536): Excelente en identificar todas las muestras de malware. - Precisión en el Test (0.930946): Alto

porcentaje de predicciones de malware correctas. - AUC en el Test (0.989268): Excelente capacidad para diferenciar entre clases.

Red Neuronal Multicapa: - CV Precisión Promedio (0.951642): Buena generalización a nuevos datos. - Precisión en el Test (0.952010): Buena exactitud en la clasificación de las muestras. - Recall en el Test (0.946970): Buena en identificar las muestras de malware, pero no tan eficiente como Random Forest. - Precisión en el Test (0.959079): Muy precisas sus predicciones de malware. - AUC en el Test (0.983050): Muy buena capacidad de diferenciación entre clases, aunque inferior a Random Forest.

Comparación de Modelos:

Random Forest es mejor para identificar malware (mayor Recall y AUC). Red Neuronal Multicapa es preferible para minimizar falsos positivos (mayor Precisión en la clasificación de malware).