Checkpoint 3 - Grupo 05

Introducción

En la tercera parte del trabajo práctico, creamos varios clasificadores utilizando diferentes modelos, ajustando hiperparámetros y seleccionando las métricas adecuadas. Finalmente, elegimos los modelos que consideramos los más apropiados para realizar predicciones.

Construcción del modelo

- Hiperparámetros optimizados para KNN

{'weights': 'distance', 'n_neighbors': 19, 'metric': 'manhattan', 'algorithm': 'brute'}

- Hiperparámetros optimizados para SVM

No logramos ejecutar exitosamente la optimización de hiperparametros para SVM ya que tardó muchísimo tiempo.

- Hiperparámetros optimizados para RF

{criterion: 'entropy', ‘max_features’: 'log2', ‘n_estimators’: 300, ‘n_jobs’: -1, ‘random_state’: 1}

- Hiperparámetros optimizados para XGBoost

{'learning_rate': 0.20630590886467293, 'max_depth': 4, 'n_estimators': 242}

- Modelos usados para el ensamble tipo voting

Tanto para el hard voting como para el soft usamos knn, random forest y xgboost.

- Modelos usados para el ensamble tipo stacking y meta modelo (meta learner) del ensamble stacking

Para stacking usamos KNN, Random Forest y XGBoost. El meta modelo del ensamble fue Random Forest.

Cuadro de Resultados

Modelo	F1-Test	Presicion Test	Recall Test	Accuracy	Kaggle
KNN	0,60	0,61	0,62	0,61	0,61
SVM*	0,77	0,78	0,79	0,77	0,76
Random Forest	0,88	0,88	0,88	0,88	0,87
XGBoost	0,87	0,88	0,88	0,86	0,86

Hard Voting	0,87	0,87	0,88	0,87	0,860
Soft Voting	0,87	0,88	0,88	0,87	0,862
Stacking	**0,88**	**0,89**	**0,89**	**0,88**	**0,87**

K-Nearest Neighbors (KNN):
En el caso de KNN, realizamos una búsqueda de hiperparámetros utilizando RandomizedSearchCV con 10 folds y 10 iteraciones. El mejor modelo de KNN se ajustó con los parámetros óptimos y se utilizó para predecir los datos de prueba. Calculamos la precisión del modelo y generamos un informe de clasificación, que proporciona métricas detalladas.

Support Vector Machine (SVM):
Para SVM, aplicamos una transformación de escala min-max a los datos de entrenamiento y luego utilizamos PCA para reducir la dimensionalidad. Después de la transformación, creamos un modelo SVM y lo entrenamos.

Random Forest:
En Random Forest, realizamos una búsqueda de hiperparámetros utilizando RandomizedSearchCV con criterios como el número de árboles, la profundidad máxima, etc. Luego, entrenamos el modelo con los mejores hiperparámetros y evaluamos su rendimiento en los datos de prueba. El modelo final se guardó en un archivo.

XGBoost:
Para XGBoost, realizamos una búsqueda aleatoria de hiperparámetros utilizando RandomizedSearchCV y la métrica de área bajo la curva ROC. Después de encontrar los mejores hiperparámetros, entrenamos el modelo XGBoost y evaluamos su rendimiento. También mostramos una curva de características para evaluar la capacidad de clasificación del modelo.

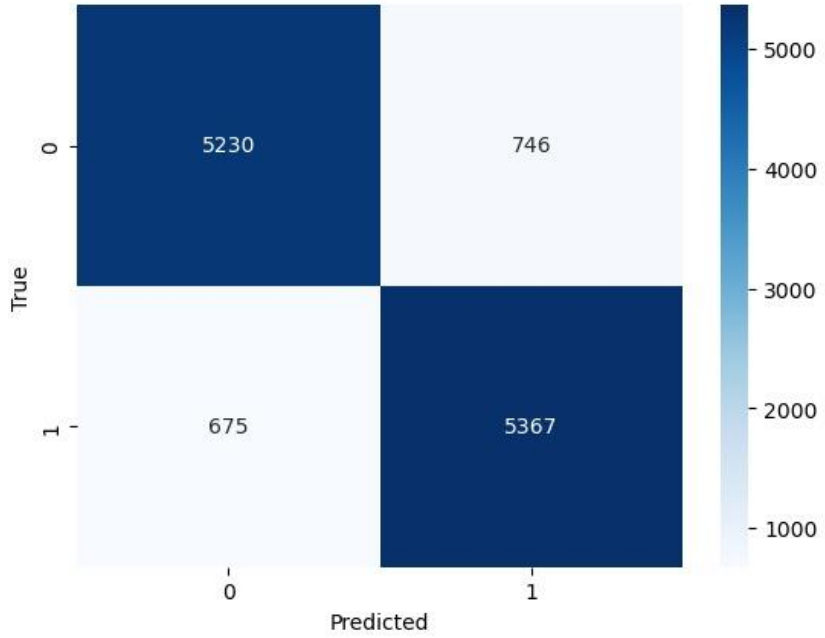
Voting:
Implementamos dos tipos de votación: Hard Voting y Soft Voting. Hard Voting toma las decisiones mayoritarias de varios modelos, mientras que Soft Voting considera las probabilidades de predicción. Utilizamos los mejores modelos de KNN, Random Forest y XGBoost para crear un clasificador de votación. Evaluamos el rendimiento de ambos clasificadores y guardamos los modelos en archivos separados.

Stacking:
El modelo Stacking demostró ser el más efectivo, combinando Random Forest, XGBoost y KNN como modelos base y utilizando un Random Forest Classifier como meta-modelo. Tras evaluar el rendimiento de los modelos base mediante validación cruzada, construimos el modelo de Stacking. Al combinar las predicciones de los modelos base y utilizar el

meta-modelo, Stacking logró un rendimiento sólido en la clasificación de los datos de prueba, destacándose como la opción preferida para la tarea de clasificación.

Matriz de Confusion

Matriz de confusión de stacking:



En el siguiente gráfico se puede observar los resultados de evaluar X_testeo en el modelo generado a través de stacking. Hay 5367 casos donde la predicción dio 1, y era lo verdadero. 5230 donde la predicción dio 0, y también era lo verdadero. Y luego, 675 casos donde la predicción dió 0 y el valor de verdad era 1, y 746 casos donde ocurrió lo contrario.

Tareas Realizadas

Integrante	Tarea
Iara Jolodovsky	KNN, Random Forest, SVM, Reporte
Martín Abramovich	XGBoost, Voting, Reporte
Tomás Vainstein Aranguren	Stacking, Reporte