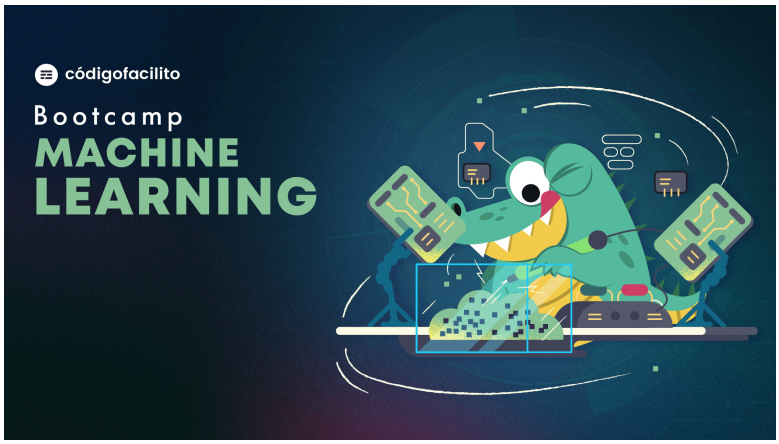
Bootcamp de Machine Learning - Código Facilito

Predicción del Estado de Pacientes con Cirrosis Hepática

18 características clínicas para predecir el estado de la cirrosis hepática

Víctor Alberto Martinez Hernandez



12 de febrero de 2024

Índice general

Descripción del Proyecto

Objetivos

- **Mejorar la precisión del diagnóstico:** Identificar con precisión el estado del paciente, lo que puede ayudar a los médicos a tomar mejores decisiones sobre el tratamiento y la gestión de la enfermedad.
- **Identificar factores predictivos clave:** Revelar qué características o factores clínicos tienen mayor influencia en la progresión de la cirrosis y en los resultados de los pacientes.

Contexto

La cirrosis es una enfermedad hepática crónica que puede tener consecuencias graves para la salud de los pacientes. Es importante desarrollar modelos de aprendizaje automático que puedan predecir los resultados de la cirrosis y ayudar a los médicos en el manejo clínico de la enfermedad.

Impacto

El desarrollo de un modelo preciso de predicción de cirrosis puede tener un impacto significativo en la práctica clínica y en los resultados de los pacientes. Al mejorar la capacidad de diagnosticar y predecir la progresión de la enfermedad, se pueden tomar medidas preventivas y de tratamiento de manera más efectiva, lo que puede mejorar la calidad de vida y la supervivencia de los pacientes.

Alcance

Desarrollo y la evaluación de un modelo de aprendizaje automático para predecir los resultados de la cirrosis utilizando datos clínicos y biomédicos. Se explorarán diversas técnicas de modelado y se evaluará la precisión y la generalización del modelo utilizando conjuntos de datos independientes.

Métricas del negocio a considerar

- Precisión diagnóstica del modelo.
- Sensibilidad y especificidad en la detección de casos de cirrosis.
- Impacto en la toma de decisiones clínicas y en el manejo de la enfermedad.
- Mejora en los resultados de los pacientes, como la supervivencia y la calidad de vida.

Stakeholders

- Médicos y profesionales de la salud involucrados en el manejo clínico de la cirrosis.
- Pacientes con cirrosis y sus familias.
- Investigadores y científicos en campos de hepatología y bioinformática.

Estructura del Repositorio

```
/
├── docs/
│   ├── figures/
│   │   ├── cf-ml.png
│   │   └── cf.png
│   ├── docs.pdf
│   ├── docs.tex
│   ├── Makefile
│   ├── preamble.sty
│   └── refs.bib
├── models/
│   └── inference-pipeline.joblib
├── notebooks/
│   ├── images/
│   │   └── cirrhosis.png
│   ├── 1_data_preparation.ipynb
│   ├── 2_model_training.ipynb
│   └── service_tester.ipynb
├── service/
│   ├── bentofile.yaml
│   ├── cirrhosis_status_runner.py
│   ├── download_bentoml_model.py
│   └── service.py
├── src/
│   ├── Makefile
│   ├── status_prediction_flow.py
│   └── utils/
│       ├── training_pipeline.py
│       └── validate_model.py
├── data/
│   ├── cirrhosis.csv
│   └── cirrhosis-profile.html
├── models/
├── README.md
└── requirements.txt
```

- **data**: Archivos .csv que contienen los datos de entrenamiento y prueba, además de un html con un análisis exploratorio básico obtenido por medio de la biblioteca ydata-profiling.
- **docs**: Documentación en formato .tex y .pdf.
- **notebooks**: Jupyter notebooks con el procesamiento de datos y la construcción de los modelos.

- **service**: Servicio de despliegue con BentoML.
- **src**: Construcción de pipelines de Metaflow.
- **models**: Modelos de machine learning.
- **README.md**: Archivo de markdown con indicaciones para la instalación, ejecución y despliegue del modelo.
- **requirements.txt**: Archivo para el listado e instalación de dependencias.

Dependencias

El proyecto utiliza Python 3.11 y las siguientes dependencias:

- numpy
- pandas
- ydata-profiling
- matlplotlib
- seaborn
- scikit-learn
- xgboost
- metaflow
- mlflow
- bentoml

En el directorio de raíz del proyecto hay un archivo requirements.txt, donde se encuentran listadas todas dependencias, se instalan ejecutando del siguiente comando:

```
pip install -r requirements.txt
```

Procesamiento de Datos

En el notebook `1-data-preparation.ipynb` se encuentran todos los procesos relacionadas a la preparación de los datos.

- Imputación de datos faltantes duplicados.
- Imputación de datos faltantes vacíos.
- Generación de nuevas características:
  - **DiagnosisDate**: La característica 'Age' representa la edad de los pacientes en días. Restando 'NDays' a 'Age' obtenemos la fecha del diagnóstico.
  - **AgeYears**: Convertir la edad del paciente (dada en días) en años.
- División de features entre numéricas y categóricas.
- Pipeline de transformación para características categóricas:
  - **OneHot-Encoder**: Se aplica a la feature 'Edema' ya que presenta tres valores posibles.
  - **Ordinal-Encoder**: Se aplica a la feature 'Stage' ya que sus valores tienen un orden establecido.
  - **Binary-Encoder**: Se aplica a las features ['Drug', 'Sex', 'Ascites', 'Hepatomegaly', 'Spiders'] ya que presentan dos valores posibles.
- Label-Encoder a la variable objetivo, ya que sus valores son categóricos (C, CL, D).

Modelado

Para abordar este problema de clasificación multiclase, se han seleccionado dos modelos de aprendizaje automático ampliamente utilizados:

- **XGBoost Classifier**: Algoritmo de gradient boosting que destaca por su eficiencia y capacidad para manejar conjuntos de datos complejos.
- **Random Forest Classifier**: Algoritmo de ensemble que combina múltiples árboles de decisión para obtener predicciones más precisas y robustas.

Evaluación del Rendimiento

El rendimiento de cada modelo se evalúa utilizando la métrica de pérdida logarítmica (log loss) a través de la validación cruzada estratificada repetida (RepeatedStratifiedKFold). Esta técnica asegura una evaluación rigurosa y confiable del rendimiento de cada modelo.

Implementación del Proceso de Modelado

El proceso de modelado se implementa utilizando la función validate_models, que acepta una lista de modelos y el conjunto de features (X) junto con las etiquetas (y). Se entrena cada modelo utilizando la estrategia de validación cruzada especificada y calcula el rendimiento promedio en los conjuntos de entrenamiento y validación.

El rendimiento de cada modelo se presenta en un DataFrame que incluye el nombre del modelo, la puntuación promedio en el conjunto de entrenamiento y la puntuación promedio en el conjunto de validación.

XGBoost Classifier demostró ser más sólido y consistente para nuestro caso de uso.

Evaluación

Para evaluar el rendimiento de los modelos, se utilizaron varias métricas y técnicas de evaluación.

Log Loss

El log loss cuantifica la diferencia entre las predicciones del modelo y las etiquetas reales. Un valor más bajo indica un mejor rendimiento del modelo, donde valores cercanos a cero representan una predicción perfecta.

Accuracy

Proporción de predicciones correctas realizadas por el modelo sobre el total de predicciones. Una precisión más alta indica un mejor rendimiento del modelo.

Matriz de Confusión

La matriz de confusión es una tabla que muestra el número de predicciones correctas e incorrectas realizadas por el modelo para cada clase. Permite visualizar el rendimiento del modelo en términos de falsos positivos, falsos negativos, verdaderos positivos y verdaderos negativos para cada clase.

Despliegue

Para el despliegue se utiliza una combinación de Metaflow, MLflow y BentoML. Estas herramientas facilitan la construcción un pipeline de entrenamiento y despliegue de modelos de manera eficiente y escalable.

Construcción del Pipeline con Metaflow y MLflow

Se define un flujo de trabajo que abarca desde la carga de datos hasta la evaluación del modelo entrenado. MLflow se utiliza para el seguimiento y la gestión de experimentos, lo que permite registrar métricas, parámetros y artefactos asociados con cada ejecución del modelo.

Despliegue del Modelo con BentoML

BentoML empaqueta el modelo entrenado como un servicio web de Python. Este facilita la creación de una API REST para el modelo, lo que permite realizar inferencias en tiempo real sobre nuevos datos a través de solicitudes HTTP.

- Descarga del modelo: `download_bento_model.py`
- Creación de servicio para la entrega de predicciones
  - `cirrhosis_status_runner.py`
  - `service.py`
- Bundle bentoml: `bentoml build`. Este construirá el bundle y le asignará una id.
- Crear imagen de docker: `bentoml containerize <bundle-id>`
- Correr imagen de docker: `docker run -rm -p 3000:3000 <bundle-id>`

Interacción con el Modelo a través de la API

Los usuarios pueden realizar llamadas a la API con nuevos datos y recibir predicciones del modelo en respuesta, lo que permite una integración sencilla y flexible del modelo en diversos sistemas y aplicaciones.

Puede ver una prueba en el notebook `service_tester.ipynb`

Bibliografía

[1] Walter Reade and Ashley Chow. Multi-class prediction of cirrhosis outcomes. https://kaggle.com/competitions/playground-series-s3e26, 2023.

[1]