Documentación de Pruebas: TestMainFacade

# Descripción General

El archivo de prueba TestMainFacade está diseñado para validar el correcto funcionamiento de la clase MainFacade. Esta clase actúa como una fachada principal que unifica las operaciones de:  
- Preprocesamiento de datos.  
- Análisis de mercado con el algoritmo Apriori.  
- Reducción de dimensionalidad usando PCA.  
- Segmentación de clientes usando K-Means.  
  
Las pruebas están implementadas utilizando el módulo de pruebas estándar de Python, unittest.

# Estructura de la Clase de Prueba

Ubicación del Archivo:  
El archivo de prueba se encuentra en el directorio:  
tests/test\_main.py

# Configuración Inicial

import unittest  
import pandas as pd  
from src.MainFacade import MainFacade # Asegúrate de que la ruta sea correcta

# Clase TestMainFacade

class TestMainFacade(unittest.TestCase):  
 @classmethod  
 def setUpClass(cls):  
 # Configura una instancia de MainFacade antes de ejecutar las pruebas  
 cls.facade = MainFacade()

# Casos de Prueba

1. Prueba de Entrenamiento del Modelo de Segmentación:  
  
def test\_train\_model\_segmentation(self):  
 # Verifica el entrenamiento del modelo de segmentación de clientes  
 data = pd.DataFrame({  
 'edad': [25, 30, 35, 40, 45],  
 'salario': [30000, 45000, 50000, 60000, 80000],  
 'frecuencia\_compra': [1, 2, 3, 4, 5]  
 })  
 data.to\_csv("test\_data.csv", index=False)  
 result = self.facade.train\_model("test\_data.csv", "segmentation", {})  
 self.assertIn("silhouette", result)  
  
2. Prueba de Guardado y Carga del Modelo:  
  
def test\_save\_and\_load\_model(self):  
 # Verifica el guardado y la carga del modelo entrenado  
 self.facade.model = self.facade.segmentation  
 self.facade.save\_model("test\_model.pkl")  
 self.facade.load\_model("test\_model.pkl")  
 self.assertIsNotNone(self.facade.model)  
  
3. Prueba de Predicción con el Modelo Cargado:  
  
def test\_test\_model(self):  
 # Verifica la predicción del modelo cargado  
 data = pd.DataFrame({  
 'edad': [29, 41],  
 'salario': [32000, 70000],  
 'frecuencia\_compra': [2, 5]  
 })  
 data.to\_csv("test\_data\_pred.csv", index=False)  
  
 # Entrenar el modelo antes de predecir  
 train\_data = pd.DataFrame({  
 'edad': [25, 30, 35, 40, 45],  
 'salario': [30000, 45000, 50000, 60000, 80000],  
 'frecuencia\_compra': [1, 2, 3, 4, 5]  
 })  
 self.facade.segmentation.fit\_kmeans(train\_data)  
  
 self.facade.model = self.facade.segmentation  
 predictions = self.facade.test\_model("test\_data\_pred.csv")  
 self.assertEqual(len(predictions), 2)

# Cómo Ejecutar las Pruebas

1. Ejecutar Directamente en PyCharm:  
- Abre el archivo test\_main.py.  
- Haz clic derecho en el archivo → Run 'test\_main'.  
  
2. Ejecutar desde la Terminal:  
- Abre la terminal de PyCharm.  
- Ejecuta el comando:  
 python -m unittest tests/test\_main.py  
  
3. Ejecutar con PYTHONPATH:  
 python -m unittest tests/test\_main.py

# Manejo de Errores Comunes

Error: ModuleNotFoundError: No module named 'src'  
Causa: PyCharm no reconoce el paquete src.  
Solución: Marcar la carpeta raíz como Sources Root.  
  
Error: ValueError: El modelo debe ser entrenado...  
Causa: El modelo no fue entrenado antes de predecir.  
Solución: Ejecutar fit\_kmeans() antes de predecir.  
  
Error: FileNotFoundError: No such file or directory  
Causa: El archivo CSV no se encuentra.  
Solución: Verificar las rutas de los archivos.

# Conclusión

Esta documentación cubre:  
- Cómo configurar y ejecutar las pruebas.  
- Explicación detallada de cada caso de prueba.  
- Cómo interpretar los resultados de las pruebas.  
- Solución a errores comunes.  
  
Con esta guía, deberías poder probar y validar correctamente la funcionalidad de la clase MainFacade.