Documentación del Modelo de Machine Learning para la Predicción de Inventarios de Productos de la Ferretería TODO REPUESTOS

1. Descripción del Proyecto

Este proyecto pretende predecir los valores mínimos y máximos de inventario para la ferretería TODO REPUESTOS, utilizando el algoritmo de Machine Learning llamado Random Forest Regressor. El objetivo principal es optimizar la gestión de inventario mediante la predicción de las cantidades de productos vendidos y, posteriormente, determinar los niveles de inventario óptimos utilizando intervalos de confianza. Esto ayudará a la ferretería TODO REPUESTOS a evitar agotamientos de stock y a optimizar sus recursos.

2. Contexto y Objetivos

Contexto

El manejo eficiente del inventario es crucial para cualquier negocio minorista, incluyendo ferreterías como TODO REPUESTOS. Un inventario mal gestionado puede resultar en pérdidas significativas debido a la falta de stock, que puede llevar a la insatisfacción del cliente y pérdida de ventas. También, el exceso de stock puede causar que se inmovilicen recursos financieros y aumentar los costos de almacenamiento. De esta forma, la predicción precisa de las cantidades mínimas y máximas de inventario se vuelve esencial para asegurar un flujo de operación óptimo y satisfacer la demanda de los clientes.

Objetivos:

- **Predecir el valor mínimo de inventario para evitar agotamientos:**
  Utilizando datos históricos de ventas y modelos de machine learning, el proyecto busca anticipar la demanda futura y determinar el nivel mínimo de inventario necesario para evitar faltantes. Esto garantiza que siempre haya suficiente stock para satisfacer la demanda sin interrupciones, mejorando la satisfacción del cliente y la eficiencia operativa.
- **Predecir el valor máximo de inventario para optimizar los recursos:**
  Además de evitar la falta de stock, es igualmente importante no sobrecargar el inventario con cantidades excesivas que podrían inmovilizar capital y aumentar los costos de almacenamiento. La predicción del valor máximo de inventario ayuda a TODO REPUESTOS a mantener niveles de stock que sean económicamente viables, asegurando una utilización óptima de los recursos disponibles.

Importancia:

Determinar los valores mínimos y máximos de inventario es vital desde la perspectiva de la cadena de suministro (supply chain) y otras áreas operativas del negocio. Una gestión efectiva del inventario no solo mejora la capacidad de respuesta del negocio ante la demanda del cliente, sino que también optimiza los costos operativos y financieros. En resumen, este proyecto no solo busca mejorar

la eficiencia del inventario, sino también contribuir al éxito general y la rentabilidad de TODO REPUESTOS mediante una planificación y gestión de inventarios más informada y precisa.

3. Datos utilizados

Fuente de Datos:

Los datos utilizados en este proyecto provienen de un archivo CSV llamado `stock_market.csv`. Este archivo contiene registros detallados de movimientos de inventario, incluyendo tanto compras como ventas de productos en la ferretería TODO REPUESTOS.

Características Relevantes:

El CSV `stock_market.csv` incluye las siguientes columnas:

- `id`: Identificación del movimiento, ya sea compra o venta.
- `product`: Código del producto.
- `effective_date`: Fecha en la que se realizó el movimiento.
- `kind`: Tipo de movimiento, ya sea "sale" (venta) o "purchase" (compra).
- `quantity`: Cantidad del producto vendido o comprado.

Proceso de Preprocesamiento:

1. **Filtrado de Datos:** Se seleccionaron solo los registros con `kind` igual a "sale".
2. **Transformación de Fechas:** A partir de la columna `effective_date`, se obtuvieron la semana del año (`week`) y el año (`year`).
3. **Construcción del DataFrame:** Se creó un DataFrame con una serie de tiempo donde cada columna representa un producto específico. Además, se añadieron dos columnas adicionales: "`week`" y "`year`", que corresponden a la semana del año y el año respectivos.

Este preprocesamiento permite estructurar los datos de manera que sean aptos para el análisis y la predicción utilizando modelos de machine learning, facilitando la identificación de patrones de venta por producto a lo largo del tiempo.

4. Modelo de Machine Learning

Algoritmo: Random Forest Regressor

El algoritmo utilizado para este proyecto es el Random Forest Regressor. Este algoritmo es un conjunto de árboles de decisión que se utilizan para la regresión. Cada árbol en el bosque se construye a partir de una muestra aleatoria del conjunto de entrenamiento, y el promedio de las predicciones de todos los árboles se toma como la predicción final. La robustez del Random Forest proviene de la combinación de múltiples árboles, lo que reduce el riesgo de sobreajuste y mejora la precisión general del modelo.

Entrenamiento:
Para entrenar el modelo, se dividió el conjunto de datos en dos partes:

- **Train (90% de los datos):** Utilizado para entrenar el modelo.
- **Test (10% de los datos):** Utilizado para evaluar el rendimiento del modelo.

El modelo fue entrenado utilizando el conjunto de entrenamiento, y se ajustaron los hiperparámetros del Random Forest para optimizar su rendimiento. La métrica utilizada para evaluar el rendimiento del modelo fue la raíz del error cuadrático medio (RMSE), que mide la diferencia promedio entre los valores predichos y los valores reales.

5. Evaluación del Modelo

Métricas Utilizadas:

- **Root Mean Squared Error (RMSE):** La RMSE se calculó para el primer producto, obteniendo un valor de 37.64. Esta métrica es útil para entender cuán cerca están las predicciones del modelo de los valores reales, en promedio.

Gráficos de Predicciones vs. Valores Reales:
Para visualizar el rendimiento del modelo, se crearon gráficos comparando las predicciones del modelo con los valores reales del conjunto de prueba. Estos gráficos ayudan a identificar cómo de bien el modelo está capturando los patrones en los datos y dónde pueden existir discrepancias.
La visualización de las predicciones vs. los valores reales proporciona una comprensión clara de la precisión del modelo y de su capacidad para generalizarse a nuevos datos.

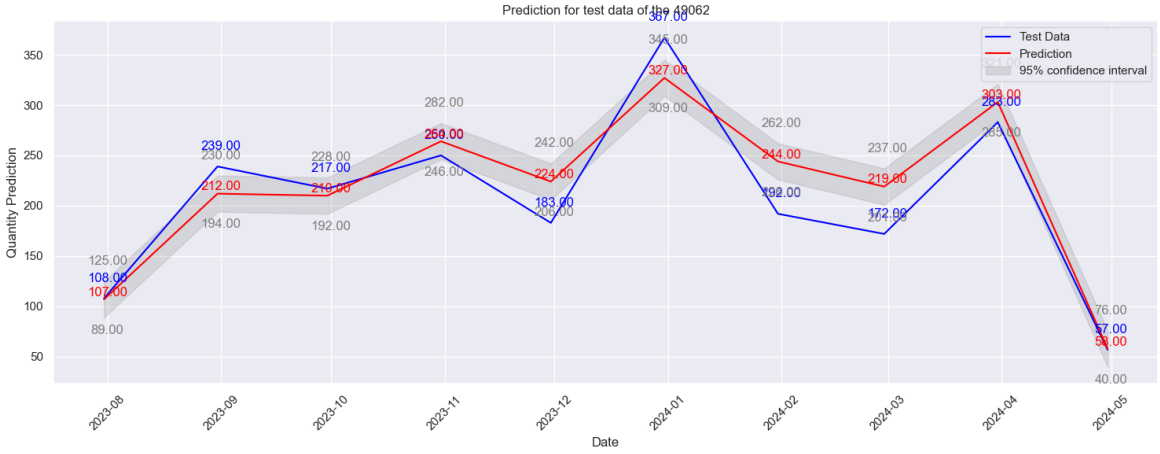

Figura 1: Gráfico de la predicción del conjunto de datos test vs el conjunto de datos reales

6. Implementación

Integración del Modelo en el Flujo de Trabajo Existente:

El modelo de predicción de inventario se ha integrado en el flujo de trabajo existente de la ferretería TODO REPUESTOS de la siguiente manera:

1. **Extracción de Datos:** Los datos de ventas se extraen periódicamente desde un archivo CSV llamado `stock_market.csv`, que se actualiza con los nuevos registros de ventas y compras.
2. **Preprocesamiento:** Se realiza un preprocesamiento de los datos para filtrar solo las ventas (`kind`='sale'), calcular las semanas y los años correspondientes, y transformar los datos en una serie de tiempo con productos como columnas.
3. **Entrenamiento del Modelo:** El modelo se entrena regularmente con el conjunto de datos de entrenamiento (90% de los datos históricos) para mantenerlo actualizado y capaz de captar nuevos patrones en los datos de ventas.
4. **Predicción:** Se generan predicciones de inventario mínimo y máximo para los próximos meses utilizando el modelo entrenado.
5. **Monitoreo y Ajustes:** Se monitorean las predicciones y se ajustan los hiperparámetros del modelo según sea necesario para mejorar su precisión.
6. **Interfaz de Usuario:** Los resultados de las predicciones se presentan a través de una interfaz de usuario que permite a los responsables de la gestión de inventario tomar decisiones informadas sobre las cantidades de stock a mantener.

Consideraciones Técnicas:

- **Escalabilidad:**
  - **Eficiencia del Modelo:** El Random Forest Regressor, siendo un conjunto de múltiples árboles de decisión, puede ser entrenado y evaluado en paralelo, lo que mejora la eficiencia y permite manejar grandes volúmenes de datos.
  - **Adaptabilidad:** El modelo puede ser reentrenado con nuevos datos conforme se registren nuevas ventas, asegurando que las predicciones se mantengan precisas a lo largo del tiempo.
- **Tiempo de Predicción:**
  - **Rendimiento:** Las predicciones del modelo son rápidas debido a la naturaleza paralela del Random Forest. Esto permite que las predicciones se generen en tiempo real o casi real, facilitando la toma de decisiones inmediata.
  - **Automatización:** El proceso de predicción está automatizado, lo que reduce la carga manual y permite que los responsables del inventario se centren en la toma de decisiones basadas en datos.

7. Ejecución del Código

La implementación del modelo de predicción de inventario se lleva a cabo en dos etapas principales: ejecución del código en Python y despliegue en una aplicación interactiva utilizando Streamlit. A continuación se detalla el proceso de ejecución.

Ejecución en Python:

- **Archivo Principal:** El código principal se ejecuta en Python mediante el archivo main.py. Este archivo contiene invoca la clase `DataAnalizer()` y sus funciones necesarias para la extracción de datos, preprocesamiento, entrenamiento del modelo y generación de predicciones.
- **Comando de Ejecución:** Para ejecutar el código en Python, se utiliza el siguiente comando en la terminal:

```
python main.py
```

Descripción del Proceso:

- El archivo main.py invoca la clase `DataAnalizer()` y sus funciones los datos que se encuentran en el archivo `ml_model.py`. La función `pre_processing.py()` se encarga del tratamiento de los datos de `stock_market.csv`, dejándolos en su forma para ser ingresados al algoritmo. La función `model_hyperparam_tuning(run=True)` hace el entrenamiento del modelo para obtener los mejores valores de los hiperparametros dado en el archivo `all_best_params_rf.json` que se encuentra en la carpeta `data`. Si se ha entrenado previmente el modelo, entonces existe el archivo `all_best_params_rf.json` y estes carga valores de los hiperparametros por producto. Por lo tanto, en este caso la función es `model_hyperparam_tuning(run=False)`. Se realiza la predicción de las fechas deseadas (máximo 6 meses) por medio de la función prediction(product, months_to_predict), donde producto es el código de un producto y months_to_predict son los meses de los datos nuevos.
- Finalmente se realizan las graficas. Una gráfica para la predicción del conjunto de test vs los datos verdaderos y otra grafica de la predicción de los meses deseados.

Despliegue en Streamlit:

Streamlit es una herramienta de código abierto que permite crear aplicaciones web interactivas para visualización de datos y machine learning. Es ideal para la implementación de dashboards y aplicaciones que requieren interacción del usuario.

1. **Introducción a Streamlit:**
   - **Interactividad:** Streamlit permite a los usuarios interactuar con el modelo de machine learning mediante una interfaz web sencilla.
   - **Visualización:** Facilita la visualización de datos y resultados de modelos en gráficos y tablas.
   - **Despliegue Rápido:** No requiere conocimientos avanzados de desarrollo web, permitiendo a los científicos de datos desplegar sus modelos rápidamente.
2. **Archivo de Aplicación:** La aplicación se ejecuta mediante el archivo `main_app.py`, que contiene el código para la interfaz de usuario y la visualización de resultados.

3. **Comando de Ejecución:** Para ejecutar la aplicación en Streamlit, se utiliza el siguiente comando en la terminal:

```
streamlit run main_app.py
```

Descripción de la Aplicación:

- **Selección del Producto:** La aplicación permite al usuario seleccionar un producto de los disponibles en el modelo entrenado.
- **Predicción de Meses:** El usuario también puede especificar la cantidad de meses para los cuales desea generar predicciones.
- **Visualización de Resultados:** La aplicación muestra dos gráficos principales:
    - **Gráfico de Conjunto de Prueba vs. Predicción:** Muestra la comparación entre los datos de prueba reales y las predicciones del modelo.
    - **Gráfico de Predicción de Meses Futuros:** Muestra las predicciones para los meses futuros, de los cuales no se tienen datos históricos.