UNIVERSIDAD PANAMERICANA

Facultad de Ingeniería

Materia: Datos Masivos

Proyecto Final

Predicción de valores para el Nifty

Profesor:

José María Sarmiento

Equipo:

Javier Vázquez Gurrola 0215391

Cristian Ramos Barragán 0228894

Javier Iñaki Hernández Esquivel 0223287

Índice

Introducción

Se busca utilizar los métodos de regresión logística y random forest para predecir los valores de datos financieros provenientes del conjunto de datos "Stock Market Data - Nifty 100 Stocks".
El conjunto de datos proporciona información detallada sobre diferentes características relacionadas con el mercado de valores, como el precio de cierre (close), el precio más alto (high), el precio más bajo (low) y el precio de apertura (open) de las acciones de las 100 empresas representadas en el índice Nifty.

El análisis y la predicción de datos del mercado de valores son de gran interés y relevancia para las empresas y los inversores que buscan tomar decisiones informadas en sus estrategias de inversión. El conjunto de datos "Stock Market Data - Nifty 100 Stocks" proporciona información valiosa sobre las acciones de las 100 principales empresas representadas en el índice Nifty, el cual es uno de los índices bursátiles más importantes de la India.

Justificación

Existen varias razones por las cuales es importante conocer y predecir los valores del mercado de valores utilizando este conjunto de datos:

1. Toma de decisiones de inversión: Las empresas y los inversores utilizan el mercado de valores como una forma de invertir y hacer crecer su capital. Conocer y predecir los valores de las acciones les permite tomar decisiones más fundamentadas al comprar o vender acciones. La capacidad de predecir los movimientos futuros del mercado puede mejorar significativamente la rentabilidad y minimizar los riesgos asociados con las inversiones.

2. Optimización de estrategias comerciales: Las empresas, especialmente las instituciones financieras, utilizan la información del mercado de valores para desarrollar estrategias comerciales efectivas. Al predecir los valores futuros, pueden diseñar estrategias que aprovechen los cambios en los precios de las acciones, identificar oportunidades comerciales y mitigar los riesgos asociados con las fluctuaciones del mercado.

3. Gestión del riesgo: El mercado de valores implica riesgos “naturales” y volatilidad. La capacidad de predecir los movimientos del mercado y los precios de las acciones puede ayudar a las empresas a gestionar el riesgo de manera más efectiva. Pueden implementar medidas de gestión de riesgos adecuadas, como la diversificación de la cartera, la asignación de activos y la gestión de posiciones, basadas en las predicciones obtenidas.

4. Planificación financiera: Las empresas utilizan los datos del mercado de valores para la planificación financiera a largo plazo. Al predecir los valores futuros, pueden realizar proyecciones financieras más precisas y desarrollar estrategias de crecimiento sostenible. Esto es especialmente relevante para empresas que cotizan en bolsa y necesitan comunicar de manera efectiva su desempeño y proyecciones a los inversores y accionistas.

Enfoque

Dado el tamaño masivo de los datos y su naturaleza temporal, se explorarán técnicas de predicción utilizando modelos de regresión logística y random forest. Estos modelos permiten aprovechar la estructura secuencial de los datos y los patrones temporales para predecir de manera precisa y confiable los valores futuros.

La regresión logística es un algoritmo de aprendizaje supervisado utilizado para problemas de clasificación binaria. Sin embargo, en el contexto de series de tiempo, se puede utilizar para predecir la dirección del movimiento de precios, como si el precio de cierre aumentará o disminuirá en el siguiente intervalo de tiempo.

Por otro lado, el algoritmo de random forest es una técnica de aprendizaje automático basada en árboles de decisión. En este caso, se aplicará a un problema de regresión para predecir valores numéricos continuos, como el precio de cierre. El random forest es conocido por su capacidad para manejar características altamente correlacionadas y capturar relaciones no lineales en los datos.

Los atributos seleccionados para la implementación son "date", "close", "high" y "low", esto debido a su relevancia e importancia en el análisis y predicción del mercado de valores.

1. Date (Fecha): El atributo "date" representa la marca de tiempo en la que se registró cada conjunto de valores en el data frame. En el contexto de las series de tiempo, la dimensión temporal juega un papel fundamental, ya que los precios de las acciones y otros indicadores financieros están sujetos a cambios y fluctuaciones a lo largo del tiempo. La fecha permite establecer una secuencia ordenada de observaciones y capturar patrones y tendencias temporales relevantes para la predicción.

2. Close (Precio de cierre): El atributo "close" representa el precio de cierre de una acción en un determinado intervalo de tiempo. El precio de cierre es ampliamente utilizado en el análisis técnico y fundamental para evaluar el rendimiento de una acción. Además, el precio de cierre refleja el consenso del mercado al final del período de negociación y es un indicador clave para evaluar las ganancias o pérdidas de una inversión. La predicción precisa del

precio de cierre es de gran interés para los inversores, ya que les permite tomar decisiones informadas sobre la compra, venta o retención de acciones.

3. High (Precio más alto): El atributo "high" representa el precio más alto alcanzado por una acción en un período de tiempo determinado. El precio más alto es relevante para identificar niveles de resistencia y determinar la fortaleza o debilidad de una acción. También puede proporcionar información sobre posibles oportunidades de compra o venta en el mercado. Incluir el precio más alto como atributo en el modelo de predicción permite capturar las fluctuaciones y los movimientos extremos del mercado que pueden influir en el precio de cierre.

4. Low (Precio más bajo): El atributo "low" representa el precio más bajo alcanzado por una acción en un período de tiempo determinado. El precio más bajo es relevante para identificar niveles de soporte y determinar la volatilidad de una acción. También puede ayudar a identificar posibles puntos de entrada o salida en el mercado. Incluir el precio más bajo como atributo en el modelo de predicción permite considerar la influencia de los movimientos bajistas y la presión de venta en el precio de cierre.

Estos atributos brindan información esencial para comprender los movimientos de precios en el mercado de valores. Al utilizar estos atributos en el desarrollo de modelos de predicción de series de tiempo, se busca capturar los patrones y tendencias propias del mercado de valores, permitiendo a las empresas tomar decisiones fundamentadas y optimizar sus estrategias de inversión.

Herramientas a utilizar para los datos

Se seleccionaron herramientas como DataProc de Google Cloud, Hadoop (HDFS) y PySpark por su capacidad para manejar eficientemente datos masivos para el procesamiento y análisis de estos últimos:

DataProc de Google Cloud: DataProc es un servicio administrado de Google Cloud que permite ejecutar clústeres de Apache Hadoop y Spark de manera sencilla. DataProc ofrece una infraestructura en la nube confiable, lo que facilita el procesamiento de grandes volúmenes de datos de manera eficiente. Además, DataProc es compatible con diversas herramientas de Big Data, lo que proporciona una solución completa para el manejo de datos masivos.

Hadoop: Hadoop está diseñado para el procesamiento de grandes conjuntos de datos. Su arquitectura se basa en HDFS. Hadoop es altamente escalable y tolerante a fallas, lo que lo convierte en una elección popular para el procesamiento de datos masivos.

PySpark: PySpark es la API de Python para Apache Spark. PySpark proporciona una interfaz de programación para el procesamiento de datos masivos utilizando Spark. Spark es conocido por su velocidad y eficiencia en el procesamiento en memoria y es capaz de manejar tareas de procesamiento de datos a gran escala. Utilizar PySpark permite aprovechar la potencia de Spark en un entorno de programación de Python, lo que facilita la implementación de algoritmos de machine learning y análisis de datos en el proyecto.

Visión general del desarrollo

Soluciones Actuales

La predicción de alzas o bajas en los valores de la bolsa de datos es un desafío complejo y altamente volátil. No existe una solución única y definitiva para predecir con precisión las fluctuaciones del mercado. Sin embargo, existen varias estrategias y enfoques que los inversores y analistas utilizan para intentar hacer predicciones informadas sobre los movimientos futuros de los precios de las acciones.

1. Análisis fundamental: Este enfoque se basa en el estudio de los fundamentos subyacentes de una empresa, como sus estados financieros, su modelo de negocio, su posición en el mercado y otros factores relevantes. Los inversores utilizan esta información para evaluar el valor intrínseco de una acción y determinar si está sobrevalorada o subvalorada.
2. Análisis técnico: El análisis técnico se centra en el estudio de los patrones y tendencias históricas de los precios de las acciones. Los analistas técnicos utilizan gráficos y herramientas específicas para identificar patrones repetitivos y señales de compra o venta. Esta técnica se basa en la premisa de que los precios históricos pueden proporcionar pistas sobre los movimientos futuros.
3. Modelos cuantitativos: Los modelos cuantitativos utilizan algoritmos y datos históricos para desarrollar modelos matemáticos y estadísticos que intentan predecir el comportamiento futuro del mercado. Estos modelos pueden involucrar análisis de regresión, aprendizaje automático y otras técnicas estadísticas avanzadas.
4. Análisis de sentimiento: El análisis de sentimiento implica el monitoreo de noticias, redes sociales y otras fuentes de información para evaluar el sentimiento general del mercado hacia una empresa o industria. Esta técnica se basa en la premisa de que las emociones y las opiniones pueden influir en los precios de las acciones.

Limitaciones actuales de la solución

Soluciones actuales

1. Análisis fundamental:
    - Información incompleta o incorrecta: La disponibilidad y la calidad de la información financiera y fundamental pueden variar según la empresa y la fuente. Además, los estados financieros pueden no capturar todos los

factores relevantes que afectan el valor de una acción, lo que puede limitar la precisión de las predicciones basadas en el análisis fundamental.
    - Incertidumbre macroeconómica: Los factores macroeconómicos, como las condiciones económicas generales, las políticas gubernamentales y los eventos geopolíticos, pueden tener un impacto significativo en el mercado y en las empresas. Estos factores son difíciles de predecir con precisión y pueden afectar las estimaciones basadas en el análisis fundamental.
2. Análisis técnico:
    - Dependencia de datos históricos: El análisis técnico se basa en el estudio de patrones y tendencias pasadas, lo que implica que los resultados se fundamentan en datos históricos. Sin embargo, los movimientos futuros del mercado pueden no seguir los mismos patrones y tendencias observados en el pasado.
    - Interpretación subjetiva: La interpretación de los gráficos y las herramientas técnicas utilizadas en el análisis técnico puede variar entre analistas. Esto puede dar lugar a diferentes conclusiones y, en última instancia, a decisiones de inversión divergentes.
3. Modelos cuantitativos
    - Supuestos simplificados: Los modelos cuantitativos a menudo se basan en supuestos simplificados sobre el comportamiento del mercado y las relaciones entre variables. Estos supuestos pueden no capturar completamente la complejidad y la dinámica del mercado real, lo que puede afectar la precisión de las predicciones.
    - Dependencia de datos históricos: Al igual que otros métodos, los modelos cuantitativos se basan en datos históricos para construir modelos y hacer predicciones. Sin embargo, los datos históricos pueden no capturar eventos o condiciones únicas que ocurrieron en el pasado y que pueden influir en el futuro. Además, los patrones históricos pueden no repetirse en el futuro debido a cambios en las condiciones del mercado.
4. Análisis de sentimiento:
    - Ruido y sesgo en los datos: Los datos utilizados en el análisis de sentimiento, como comentarios en redes sociales o noticias, pueden contener ruido y sesgos. Los comentarios pueden estar influenciados por factores externos, como campañas de manipulación o bots, lo que puede distorsionar los resultados del análisis.
    - Generalización limitada: Los modelos de análisis de sentimiento se entrenan en conjuntos de datos específicos y pueden tener dificultades para generalizar a nuevas situaciones o dominios. La transferencia de conocimientos de un dominio a otro puede ser desafiante, lo que limita la aplicabilidad del análisis de sentimiento en diferentes contextos.

Solución Propuesta

La solución tiene como objetivo analizar la tendencia de los datos de precios de acciones y predecir sus siguientes valores por medio de modelos de machine learning, en este caso series de tiempo, que implementen los modelos de regresión lineal y Random forest.. Para lograr esto, se utiliza Apache Spark y PySpark, que es una interfaz de programación de Python para Spark.
Se utilizarán múltiples archivos CSV que contienen datos de precios de acciones del Nifty, provenientes del dataset de Kaggle “Stock Market Data - Nifty 100 Stocks (1 min) data”.

Los indicadores relevantes para la solución son : 'close', 'open', 'high' y 'low'. Esto para proporcionar una visión general del posible comportamiento de las acciones de 100 distintas empresas, sin abrumar con demasiados indicadores e información al interesado.
A continuación, se realiza un análisis de la tendencia de estos datos mediante el cálculo de las diferencias entre los valores actuales y anteriores de diferentes columnas, como 'close', 'open', 'high' y 'low'. Esto proporciona información sobre la dirección de la tendencia en los precios de las acciones, lo que puede ser útil para análisis posteriores o la construcción de modelos predictivos.

La función timeSeriesPrediction implementada en la solución, tiene como objetivo realizar la predicción de una serie de tiempo utilizando modelos de regresión lineal y Random Forest.
El proceso general de la función es el siguiente:

1. Preparación de los datos: El DataFrame df que se pasa como entrada debe contener una columna de fechas ("date") y una columna numérica que representa los valores de la serie de tiempo que se desea predecir (columnName). Además, se realiza una conversión de tipo de datos asegurando que la columna columnName sea del tipo entero.

2. Creación de valores anteriores: Se crean columnas adicionales en el DataFrame que contienen los valores anteriores de la serie de tiempo. Estas columnas se generan utilizando una ventana deslizante y la función lag. En este caso, se crean tres columnas adicionales que representan los valores de los tres días anteriores.

3. Manejo de valores nulos: Se eliminan las filas que contienen valores nulos en las columnas de los valores anteriores. Si alguna de estas columnas es nula, se reemplaza con el valor de la columna columnName correspondiente. Esto se realiza para asegurar que no haya valores faltantes en los datos utilizados para el entrenamiento y la predicción.

4. Preparación del DataFrame para el modelo: Se utiliza la clase VectorAssembler para transformar el DataFrame en un formato adecuado para los modelos de regresión. Se seleccionan las columnas de los valores anteriores y se agrupan en una sola columna llamada "features", que contiene un vector con estas características.

5. División de los datos: El DataFrame se divide en un conjunto de datos de entrenamiento y un conjunto de datos de prueba utilizando el método randomSplit.

En este caso, se asigna el 70% de los datos para el entrenamiento y el 30% restante para la prueba.

6. Creación y ajuste de los modelos: Se crean dos modelos, un modelo de regresión lineal y un modelo de Random Forest. Estos modelos se ajustan utilizando los datos de entrenamiento.

7. Predicción y evaluación: Se realizan predicciones utilizando los modelos ajustados en los datos de prueba. Luego, se calcula el error cuadrático medio (RMSE) para evaluar el rendimiento de cada modelo en la predicción de la serie de tiempo.

8. Visualización de las predicciones: Se muestra gráficamente la comparación entre los valores reales de la serie de tiempo y las predicciones generadas por los modelos de regresión lineal y Random Forest. Esto permite visualizar el desempeño de los modelos en la predicción de la serie de tiempo y comparar las tendencias y patrones capturados por cada modelo.

Si bien la solución propuesta ofrece numerosas ventajas, también es importante tener en cuenta sus limitaciones. Es esencial comprender las restricciones y las posibles fuentes de error asociadas con la implementación de esta solución.

- Incertidumbre del mercado: Aunque el modelo puede proporcionar predicciones basadas en datos históricos, el mercado de valores es inherentemente volátil y está sujeto a factores externos impredecibles, como eventos económicos, políticos o sociales. Estos eventos pueden tener un impacto significativo en los precios de las acciones y pueden desafiar la precisión de las predicciones del modelo.

- Riesgo de sobreajuste: Si no se maneja adecuadamente, existe el riesgo de que el modelo se sobreajuste a los datos históricos y no generalice bien a nuevos datos o escenarios futuros. Esto puede llevar a predicciones inexactas o poco confiables, especialmente si los datos utilizados en el entrenamiento del modelo no representan adecuadamente las condiciones del mercado.

- Calidad y disponibilidad de los datos: Si los datos utilizados contienen errores, sesgos o falta de información relevante, esto puede afectar negativamente las predicciones del modelo. Además, puede ser difiícil encontrar conjuntos de datos históricos completos y confiables para entrenar el modelo.

- Limitaciones de los algoritmos y las técnicas utilizadas: Los algoritmos de aprendizaje automático utilizados, como Random Forest o regresión lineal, tienen sus propias limitaciones inherentes. Por ejemplo, pueden tener dificultades para capturar relaciones no lineales o pueden requerir suposiciones específicas que no se cumplen en el contexto financiero. Además, las técnicas utilizadas pueden no ser lo suficientemente sofisticadas como para capturar todos los aspectos complejos del mercado de valores.

- Costos y recursos computacionales: El procesamiento y análisis de grandes volúmenes de datos requiere recursos computacionales significativos. La

implementación de tecnologías como Hadoop, PySpark y servicios en la nube puede conllevar costos adicionales incluyendo costos de tiempo.

Propósito

El propósito de este proyecto es utilizar métodos de regresión logística y random forest para predecir los valores de datos financieros del conjunto de datos "Stock Market Data - Nifty 100 Stocks". Este conjunto de datos proporciona información detallada sobre diferentes características relacionadas con el mercado de valores de la India, como el precio de cierre, el precio más alto, el precio más bajo y el precio de apertura de las acciones de las 100 empresas representadas en el índice Nifty.

El análisis y la predicción de datos del mercado de valores son de gran interés y relevancia para las empresas y los inversores que buscan tomar decisiones informadas en sus estrategias de inversión. Al utilizar técnicas de regresión logística y random forest, se busca aprovechar la estructura secuencial de los datos y los patrones temporales para predecir de manera precisa y confiable los valores futuros.

El proyecto tiene como objetivo proporcionar a las empresas y los inversores herramientas y conocimientos para la toma de decisiones de inversión, la optimización de estrategias comerciales, la gestión del riesgo y la planificación financiera. Al predecir los valores del mercado de valores, se pueden tomar decisiones fundamentadas en la compra, venta o retención de acciones, se pueden desarrollar estrategias comerciales efectivas, se pueden implementar medidas de gestión de riesgos adecuadas y se pueden realizar proyecciones financieras más precisas.

Revisión y uso de datos

Control de datos

Nombre	Descripción	Tipo de dato	Rango de valores
date	Día y hora del registro	categórico	Mayor a 02-02-2022 10 28 am
close	Valor de cierre	numérico	0-100000
close_trend_direction	Tendencia al cierre	categórico	No change: Sin cambios low: Tiende hacia abajo high: Tiende hacia arriba
high	Valor más alto	numérico	0-100000

high_trend_direction	Tendencia del valor	categórico	No change: Sin cambios low: Tiende hacia abajo high: Tiende hacia arriba
low	Valor más bajo	numérico	0-100000
low_trend_direction	Tendencia del valor	categórico	No change: Sin cambios low: Tiende hacia abajo high: Tiende hacia arriba
open	Valor en la apertura	numérico	0-100000
open_trend_direction	Tendencia en la apertura	categórico	No change: Sin cambios low: Tiende hacia abajo high: Tiende hacia arriba

En general, los campos mencionados en el dataset de mercado de valores son importantes debido a su relevancia para el análisis y la comprensión del comportamiento del mercado y las acciones.

Estos campos proporcionan información clave sobre los precios y tendencias de las acciones en un determinado período de tiempo. El precio de cierre (Close) es un indicador fundamental que refleja el valor al que se negoció la acción al final del período. Es ampliamente utilizado para evaluar el rendimiento de una acción y tomar decisiones de compra o venta.

La dirección de la tendencia del precio de cierre (Close_trend_direction) ofrece información sobre el impulso y la fortaleza de una acción. Puede ayudar a identificar cambios en la tendencia y proporcionar señales para la toma de decisiones de inversión.

Los precios más altos (High) y más bajos (Low) alcanzados durante un período son indicadores clave de los niveles de resistencia y soporte, respectivamente. Estos niveles son utilizados por los inversores para evaluar la volatilidad y determinar los puntos de entrada o salida de una acción.

La dirección de la tendencia de los precios más altos (High_trend_direction) y más bajos (Low_trend_direction) brinda señales sobre la fortaleza de una acción y la posible continuación o reversión de la tendencia existente.

El precio de apertura (Open) representa el valor al que se negoció la acción al comienzo del período. Puede proporcionar información sobre el sentimiento inicial del mercado hacia una acción y cómo podría reaccionar ante eventos o noticias relevantes.

La dirección de la tendencia del precio de apertura (Open_trend_direction) ayuda a evaluar la continuidad o reversión de la tendencia existente y anticipar posibles movimientos de precios.

Preparación de datos

1. Obtención de los datos: Descargar al cluster el dataset "Stock Market Data - Nifty 100 Stocks (1 min)" desde Kaggle, que contiene información sobre los precios e indicadores de acciones. (imagen 1)

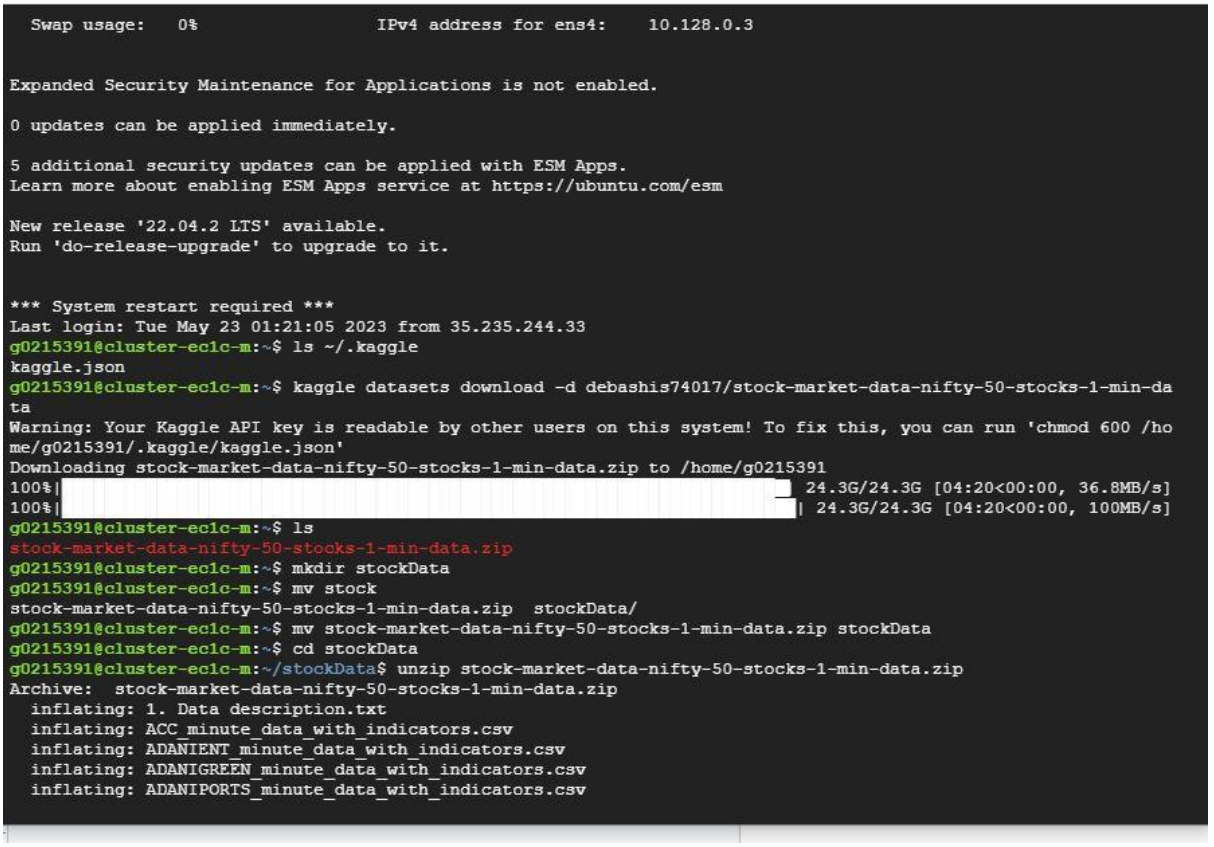


imagen 1

2. Carga de los datos en Hadoop: Utilizando Hadoop, como HDFS (Hadoop Distributed File System), para cargar los datos descargados desde Kaggle en el clúster. (imagen 2)

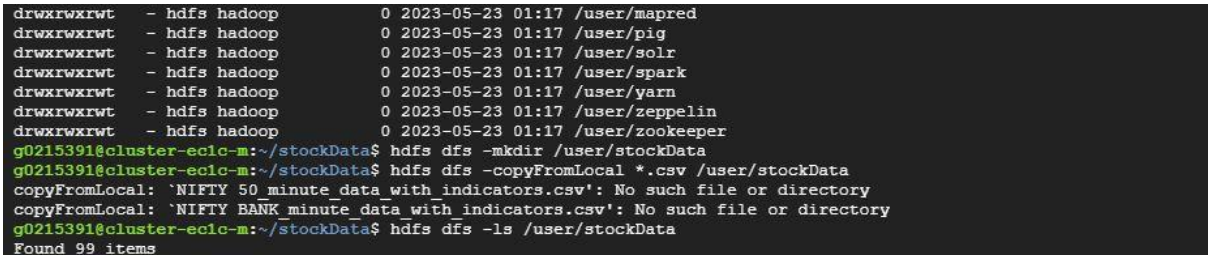


imagen 2

3. Exportación de los datos a PySpark: Una vez que los datos han sido cargados a Hadoop, se pueden exportar a PySpark, que es un entorno de Python. Los datos se convierten en data frames de PySpark, que son estructuras de datos tabulares que facilitan el análisis y la manipulación de los datos. (imagen 3)

```
#Carga de los data frames a Spark
df1 = spark.read.csv("/user/stockData/ACC_minute_data_with_indicators.csv", header = True)
df2 = spark.read.csv("/user/stockData/ADANIENT_minute_data_with_indicators.csv", header = True)
df3 = spark.read.csv("/user/stockData/ADANIGREEN_minute_data_with_indicators.csv", header = True)
df4 = spark.read.csv("/user/stockData/ADANIPORTS_minute_data_with_indicators.csv", header = True)
df5 = spark.read.csv("/user/stockData/AMBUJACEM_minute_data_with_indicators.csv", header = True)
df6 = spark.read.csv("/user/stockData/APOLLOHOSP_minute_data_with_indicators.csv", header = True)
df7 = spark.read.csv("/user/stockData/ASIANPAINT_minute_data_with_indicators.csv", header = True)
df8 = spark.read.csv("/user/stockData/AUROPHARMA_minute_data_with_indicators.csv", header = True)
df9 = spark.read.csv("/user/stockData/AXISBANK_minute_data_with_indicators.csv", header = True)
df10 = spark.read.csv("/user/stockData/BAJAJ-AUTO_minute_data_with_indicators.csv", header = True)
df11 = spark.read.csv("/user/stockData/BAJAJFINSV_minute_data_with_indicators.csv", header = True)
df12 = spark.read.csv("/user/stockData/BAJAJHLDNG_minute_data_with_indicators.csv", header = True)
df13 = spark.read.csv("/user/stockData/BAJFINANCE_minute_data_with_indicators.csv", header = True)
df14 = spark.read.csv("/user/stockData/BANDHANBNK_minute_data_with_indicators.csv", header = True)
df15 = spark.read.csv("/user/stockData/BANKBARODA_minute_data_with_indicators.csv", header = True)
df16 = spark.read.csv("/user/stockData/BERGEPAINT_minute_data_with_indicators.csv", header = True)
df17 = spark.read.csv("/user/stockData/BHARTIARTL_minute_data_with_indicators.csv", header = True)
df18 = spark.read.csv("/user/stockData/BIOCON_minute_data_with_indicators.csv", header = True)
df19 = spark.read.csv("/user/stockData/BOSCHLTD_minute_data_with_indicators.csv", header = True)
df20 = spark.read.csv("/user/stockData/BPCL_minute_data_with_indicators.csv", header = True)
df21 = spark.read.csv("/user/stockData/BRITANNIA_minute_data_with_indicators.csv", header = True)
df22 = spark.read.csv("/user/stockData/CHOLAFIN_minute_data_with_indicators.csv", header = True)
df23 = spark.read.csv("/user/stockData/CIPLA_minute_data_with_indicators.csv", header = True)
df24 = spark.read.csv("/user/stockData/COALINDIA_minute_data_with_indicators.csv", header = True)
df25 = spark.read.csv("/user/stockData/COLPAL_minute_data_with_indicators.csv", header = True)
df26 = spark.read.csv("/user/stockData/DABUR_minute_data_with_indicators.csv", header = True)
df27 = spark.read.csv("/user/stockData/DIVISLAB_minute_data_with_indicators.csv", header = True)
df28 = spark.read.csv("/user/stockData/DLF_minute_data_with_indicators.csv", header = True)
df29 = spark.read.csv("/user/stockData/DMART_minute_data_with_indicators.csv", header = True)
df30 = spark.read.csv("/user/stockData/DRREDDY_minute_data_with_indicators.csv", header = True)
df31 = spark.read.csv("/user/stockData/EICHERMOT_minute_data_with_indicators.csv", header = True)
df32 = spark.read.csv("/user/stockData/GAIL_minute_data_with_indicators.csv", header = True)
df33 = spark.read.csv("/user/stockData/GLAND_minute_data_with_indicators.csv", header = True)
df34 = spark.read.csv("/user/stockData/GODREJCP_minute_data_with_indicators.csv", header = True)
df35 = spark.read.csv("/user/stockData/GRASIM_minute_data_with_indicators.csv", header = True)
df36 = spark.read.csv("/user/stockData/HAVELLS_minute_data_with_indicators.csv", header = True)
df37 = spark.read.csv("/user/stockData/HCLTECH_minute_data_with_indicators.csv", header = True)
```

imagen 3

4. Transformación y limpieza de los datos: Aplicar operaciones de transformación y limpieza para asegurarse de que los datos sean coherentes y estén en el formato adecuado. Como ya se mencionó, esta acción se basa en la eliminación de valores nulos, la corrección de errores de formato y eliminación de datos innecesarios. (imagen 4)

```python
# Actualizar y mostrar cada dataframe individualmente
for i, dataframe in enumerate(dataframes):
    # Desechar columnas que no se van a utilizar
    columns_to_keep = ['date', 'close', 'high', 'low', 'open', 'volume']  # Lista de columnas a mantener
    updated_dataframe = dataframe.select(*columns_to_keep)

    # Asignar el dataframe actualizado a una nueva variable con el mismo nombre
    globals()[dataframes[i]] = updated_dataframe
```

imagen 4

5. Modelado de datos: Realizar análisis exploratorios, y construir los modelos predictivos. Esto puede incluye la aplicación de técnicas como regresión líneal y el Random Forest.

6. Validación y evaluación de los resultados: Evaluar los resultados obtenidos del modelado de datos para verificar su precisión y coherencia. Esto puede incluir la comparación de las predicciones del modelo con los valores reales y la evaluación de métricas de rendimiento. (imagen 5)

```
23/05/27 20:26:52 WARN WindowExec: No Partition Defined for Window operati
Mean Squared Error for Linear Regression Model is: 1.685504541193083
Mean Squared Error for Random Forest Model is: 24.869168500417967
23/05/27 20:26:55 WARN WindowExec: No Partition Defined for Window operati
```

imagen 5

Tratamiento de datos

El tratamiento de datos en el contexto de convertirlos en un formato adecuado para su uso en un modelo de aprendizaje automático implica una serie de pasos para asegurar que los datos estén en la forma y estructura correcta. A continuación se muestra el proceso que se siguio para realizar esta actividad:

1. Manejo de datos faltantes: Identificar y tratar los valores nulos en el conjunto de datos. Esto puede implicar eliminar filas o columnas con datos faltantes, imputar valores faltantes utilizando técnicas como la imputación media o mediana.

```python
# Agregar columnas con los valores anteriores
df = df.withColumn(columnName + "day1", lag(col(columnName)).over(window))
df = df.withColumn(columnName + "day2", lag(col(columnName), 2).over(window))
df = df.withColumn(columnName + "day3", lag(col(columnName), 3).over(window))



# Eliminar filas con valores nulos
df = df.withColumn(columnName + "day1", when(df[columnName + "day1"].isNull(), df[columnName]).otherwise(df[columnName + "day1"]))
df = df.withColumn(columnName + "day2", when(df[columnName + "day2"].isNull(), df[columnName]).otherwise(df[columnName + "day2"]))
df = df.withColumn(columnName + "day3", when(df[columnName + "day3"].isNull(), df[columnName]).otherwise(df[columnName + "day3"]))
```

2. Eliminación de características irrelevantes: Si hay características en el conjunto de datos que no aportan información útil para el modelo, se eliminarán.

```
# Actualizar y mostrar cada dataframe individualmente
for i, dataframe in enumerate(dataframes):
    # Desechar columnas que no se van a utilizar
    columns_to_keep = ['date', 'close', 'high', 'low', 'open', 'volume']  # Lista de columnas a mantener
    updated_dataframe = dataframe.select(*columns_to_keep)

    # Asignar el dataframe actualizado a una nueva variable con el mismo nombre
    globals()[dataframes[i]] = updated_dataframe

# Combinar los dataframes utilizando union
combined_df = dataframes[0]
for i in range(1, len(dataframes)):
    combined_df = combined_df.union(dataframes[i])

#Desechar columnas que no se van a utilizar
columns_to_keep = ['date', 'close', 'high', 'low', 'open', 'volume']  # Lista de columnas a mantener
combined_df = combined_df.select(*columns_to_keep)
combined_df.show()
```

Integridad de los datos

Con base al dataset específico utilizado en el proyecto, se pueden aplicar diversas técnicas de análisis de integridad de datos. Para el proyecto realizamos los siguientes pasos para mantener la integridad de los datos.

1. Verificación de datos faltantes: Examinar si existen valores faltantes en las variables clave del dataset, como "date", "close", "high", "low", y "open". Identificar la proporción de valores faltantes en cada variable y determinar si es necesario imputarlos o eliminar registros incompletos.

2. Validación de la consistencia de los datos: Verificar si existen discrepancias o inconsistencias en las relaciones esperadas entre las variables. Por ejemplo, asegurarse de que los valores de "close" sean coherentes con los valores de "high" y "low". También es importante revisar la dirección de la tendencia ("trend_direction") para asegurarse de que esté correctamente asociada con los valores correspondientes.

Limitaciones de los datos

Al analizar un dataset y considerar su uso en un modelo de aprendizaje automático, es importante tener en cuenta las posibles limitaciones o desafíos asociados a los datos. A continuación, se presentan algunas posibles limitaciones del dataset que podrían afectar su calidad y la precisión del modelo:

- Calidad de los datos: El dataset puede contener errores, inconsistencias o valores atípicos que afecten la calidad de los datos. Estos problemas pueden surgir debido a errores en la recopilación, el almacenamiento o la manipulación de los datos.

- Datos faltantes: El dataset puede tener valores faltantes en algunas variables. Esto puede deberse a diversas razones, como problemas de recopilación de datos, errores de ingreso o simplemente falta de disponibilidad de ciertos datos.

- Sesgo de muestra: El dataset puede no ser representativo de la población o el fenómeno que se está estudiando. Por ejemplo, puede haber una falta de diversidad en los datos en términos de tamaño de la empresa u otros factores relevantes. Esto puede resultar en un sesgo en el modelo y limitar su capacidad para generalizar a situaciones fuera del alcance del dataset.

Proceso de desarrollo

Metodología

Construcción del modelo

Para el desarrollo de una función de series de tiempo fue necesario realizar una conversión de tipo de datos de la columna seleccionada para asegurar la consistencia de los datos. Posteriormente, se agregaron nuevas columnas al DataFrame, representando los valores anteriores de la columna seleccionada para los días 1, 2 y 3. Estos valores anteriores son calculados utilizando la función lag() en combinación con la ventana deslizante.

Para garantizar la integridad de los datos, se eliminaron las filas que contenían valores nulos en las columnas de valores anteriores. Se utilizó la función when() en conjunto con el método isNull() para reemplazar los valores nulos con los valores de la columna original. De esta manera, se evita la pérdida de datos y se asegura que el modelo tenga suficiente información para realizar las predicciones.

Con el fin de preparar los datos para su uso en el modelo, se seleccionaron las columnas de características relevantes. Estas columnas corresponden a los valores de interés como ”open”, “close”, “”high”, “low”. A continuación, se utilizó el VectorAssembler para combinar 3 valores anteriores de las columnas corresposndientes en una sola columna denominada "features". Esta columna representa la entrada del modelo y contiene la información necesaria para realizar las predicciones para la serie de tiempo.

```python
def timeSeriesPrediction(df, columnName):
   # Crear una ventana deslizante para calcular los valores anteriores
    window = Window.orderBy("date")
    df = df.withColumn(columnName, df[columnName].cast(IntegerType()))

    # Agregar columnas con los valores anteriores
    df = df.withColumn(columnName + "day1", lag(col(columnName)).over(window))
    df = df.withColumn(columnName + "day2", lag(col(columnName), 2).over(window))
    df = df.withColumn(columnName + "day3", lag(col(columnName), 3).over(window))


    # Eliminar filas con valores nulos
    df = df.withColumn(columnName + "day1", when(df[columnName + "day1"].isNull(), df[columnName]).otherwise(df[columnName + "day1"]))
    df = df.withColumn(columnName + "day2", when(df[columnName + "day2"].isNull(), df[columnName]).otherwise(df[columnName + "day2"]))
    df = df.withColumn(columnName + "day3", when(df[columnName + "day3"].isNull(), df[columnName]).otherwise(df[columnName + "day3"]))

    # Convertir DataFrame a un formato adecuado para el modelo
    feature_cols = [columnName + "day1", columnName + "day2", columnName + "day3"]
    assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")
    df = assembler.transform(df)
```

Entrenamiento y validación del modelo

El conjunto de datos se dividió en conjuntos de entrenamiento y prueba utilizando la función randomSplit(). Se asignó el 70% de los datos al conjunto de entrenamiento y el 30% restante al conjunto de prueba. Esta división es crucial para evaluar el rendimiento del modelo en datos no vistos durante el entrenamiento.

Se creó un modelo de regresión lineal utilizando la clase LinearRegression de PySpark. Se especificaron la columna de etiquetas ("labelCol") y la columna de características ("featuresCol"). El modelo fue ajustado utilizando los datos de entrenamiento mediante el método fit().

Se creó un modelo de Random Forest utilizando la clase RandomForestRegressor de PySpark. Al igual que en el modelo de regresión lineal, se especificaron la columna de etiquetas y la columna de características. Se utilizó un número de árboles igual a 100. El modelo fue ajustado utilizando los datos de entrenamiento.

Se realizaron predicciones en los datos de prueba utilizando los modelos de regresión lineal y Random Forest ajustados. Para cada modelo, se utilizó el método transform() para obtener las predicciones en el conjunto de prueba. Luego, se utilizó la métrica del error cuadrático medio (RMSE) para evaluar el rendimiento de los modelos. Se utilizó la clase RegressionEvaluator de PySpark, especificando la columna de etiquetas y el nombre de la métrica ("rmse").

```
# Dividir el conjunto de datos en entrenamiento y prueba
train_data, test_data = df.randomSplit([0.7, 0.3], seed=1)

# Crear el modelo de regresión lineal
lr = LinearRegression(labelCol=columnName, featuresCol="features")
lr_model = lr.fit(train_data)

# Crear el modelo de Random Forest
rf = RandomForestRegressor(labelCol=columnName, featuresCol="features", numTrees=100)
rf_model = rf.fit(train_data)

# Realizar predicciones en los datos de prueba
lr_predictions = lr_model.transform(test_data)
rf_predictions = rf_model.transform(test_data)

# Calcular el error cuadrático medio
evaluator = RegressionEvaluator(labelCol=columnName, metricName="rmse")
rmse_lr = evaluator.evaluate(lr_predictions)
rmse_rf = evaluator.evaluate(rf_predictions)
```

Evaluación del modelo

Se imprimen los valores del error cuadrático medio (RMSE) para el modelo de regresión lineal y el modelo de Random Forest. Estos valores proporcionan una medida de la calidad de las predicciones realizadas por cada modelo. Cuanto menor sea el valor del RMSE, mejor será el ajuste del modelo a los datos de prueba.

Los datos de prueba, junto con las predicciones realizadas por los modelos, se convierten a un formato de Pandas DataFrame. Esta conversión permite una fácil manipulación y visualización de los datos.

Se muestran gráficamente las predicciones realizadas por los modelos de regresión lineal y Random Forest. Se plotea la columna de valores reales ("Actual") junto con las predicciones realizadas por cada modelo. Esto permite visualizar cómo se ajustan las predicciones a los valores reales y evaluar la capacidad del modelo para capturar las tendencias y patrones en los datos.

```
# Imprimir el error cuadrático medio
print("Mean Squared Error for Linear Regression Model is:", rmse_lr)
print("Mean Squared Error for Random Forest Model is:", rmse_rf)

# Convertir los datos de prueba a pandas DataFrame para la visualización
lr_predictions_pd = lr_predictions.select(columnName, "prediction").toPandas()
rf_predictions_pd = rf_predictions.select(columnName, "prediction").toPandas()

# Mostrar gráficamente las predicciones
plt.figure(figsize=(12, 8))
plt.plot(lr_predictions_pd[columnName], label="Actual " + columnName)
plt.plot(lr_predictions_pd["prediction"], label="Linear Regression Predictions")
plt.legend(loc="upper left")
plt.title(columnName + " - Linear Regression Predictions")
plt.show()

plt.figure(figsize=(12, 8))
plt.plot(rf_predictions_pd[columnName], label="Actual " + columnName)
plt.plot(rf_predictions_pd["prediction"], label="Random Forest Predictions")
plt.legend(loc="upper left")
plt.title(columnName + " - Random Forest Predictions")
plt.show()
```

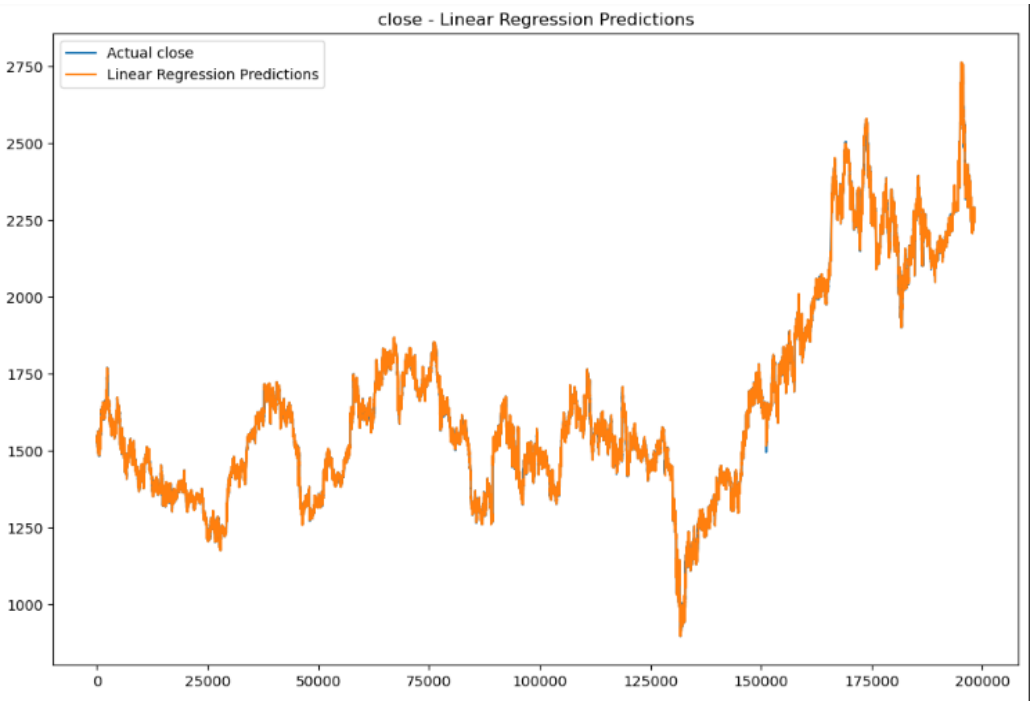
Pruebas

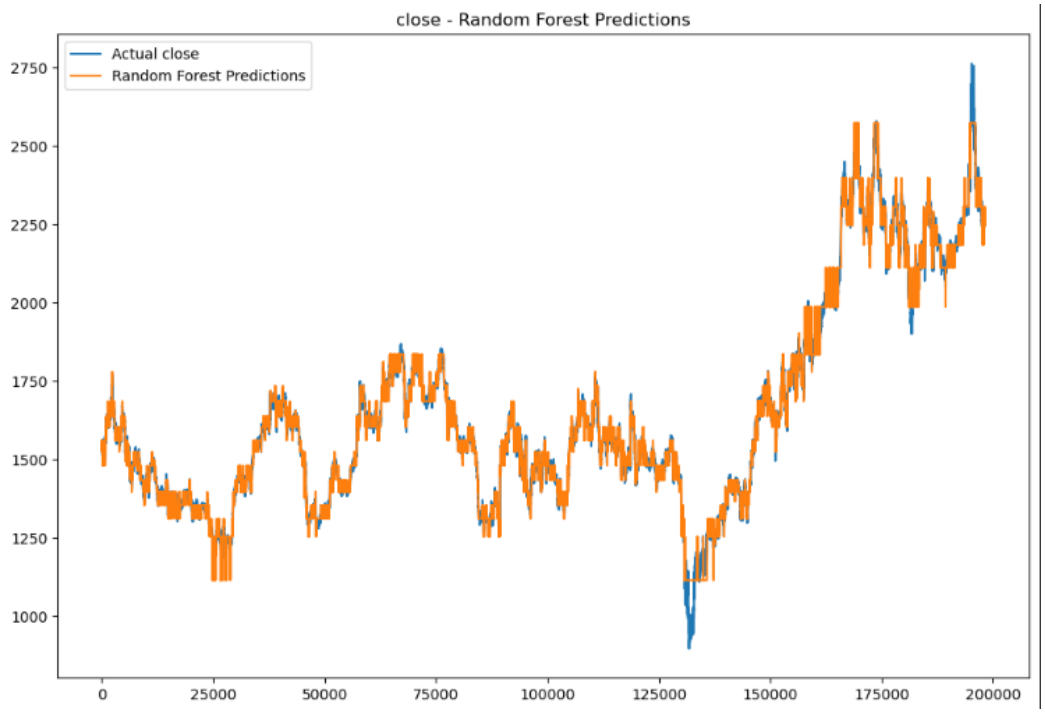
Se itera sobre los dataframes disponibles utilizando una lista de nombres de dataframes. Para cada dataframe, se realiza la implementación y el monitoreo del modelo.
Se obtiene el dataframe correspondiente utilizando el nombre del dataframe obtenido en el bucle. Esto permite trabajar con cada dataframe de manera individual con el objetivo de obtener predicciones para cada empresa dentro del Nifty.

Se itera sobre las columnas de interés, que se especifican en una lista. Para cada columna, se realiza el monitoreo del rendimiento del modelo utilizando la función timeSeriesPrediction().
Se imprimen los resultados del monitoreo para cada columna y cada dataframe. Esto permite tener una visión general del rendimiento del modelo en cada caso.

Se muestra un ejemplo de predicción por ambos modelos al dataframe 1 sobre el valor de close, con errores de:

```
23/05/27 20:26:52 WARN WindowExec: No Partition Defined for Window op
Mean Squared Error for Linear Regression Model is: 1.685504541193083
Mean Squared Error for Random Forest Model is: 24.869168500417967
23/05/27 20:26:55 WARN WindowExec: No Partition Defined for Window op
```





La capacidad de iterar sobre los dataframes y las columnas permite automatizar el proceso de monitoreo y evaluar el rendimiento del modelo de manera eficiente. Esto facilita la identificación de posibles problemas o variaciones en el rendimiento del modelo en diferentes contextos.

Conclusión

En general, el proyecto ha arrojado resultados positivos al utilizar técnicas de machine learning para predecir los estados de la bolsa de valores de la India. Estos resultados respaldan la idea de que el uso de modelos basados en datos puede brindar información valiosa para la toma de decisiones financieras. Sin embargo, es importante tener en cuenta que estos modelos son herramientas de apoyo y no deben considerarse como una fuente infalible de predicciones precisas. Los resultados obtenidos deben interpretarse con precaución y considerarse junto con otros factores relevantes, como el análisis fundamental y las condiciones económicas. La combinación de la experiencia humana y el análisis basado en datos puede proporcionar una visión más completa y respaldar la toma de decisiones informada en el mercado de valores.
De igual manera es importante mencionar que un factor que resulta de gran relevancia para modelos de este tipo es el tipo de datos y su calidad. Las predicciones obtenidas a partir de los datos utilizados muestran en algunos casos un posible sobre aprendizaje y similitudes importantes entre atributos distintos. Como un proyecto de prueba los resultados se consideran satisfactorios, sin embargo es probable que los datos utilizados no sean de la mejor calidad.

Resultados

https://drive.google.com/file/d/10YfRX7_kjLjq0G0vfyXpkjQn7HYr10Ke/view?usp=sharing