PROYECTO ARIMA

Universidad Autónoma de Querétaro

Yañez Omaña Emilio

Proyecto : Análisis de Predicción de Rendimientos de NVIDIA Usando ARIMA

Introducción al Modelo ARIMA

El modelo ARIMA (Autoregressive Integrated Moving Average) es una herramienta estadística ampliamente utilizada para modelar y predecir series temporales. Su principal ventaja radica en su capacidad para capturar patrones de dependencia temporal en datos, como tendencias, estacionalidades y fluctuaciones aleatorias. El modelo ARIMA se compone de tres componentes principales:

1. **Autoregresión (AR)**: Representa la dependencia lineal entre un valor actual y sus valores pasados.
2. **Diferenciación (I)**: Permite transformar una serie no estacionaria en estacionaria eliminando tendencias.
3. **Media Móvil (MA)**: Modela la relación entre un valor actual y el error de predicción de periodos pasados.

Metodología

Para este proyecto, se utilizó el modelo ARIMA para predecir el precio ajustado de cierre de las acciones de NVIDIA (NVDA). Los pasos principales del análisis fueron los siguientes:

1. **Descarga de Datos**: Se obtuvieron datos históricos del precio ajustado de cierre de NVIDIA desde Yahoo Finance. El periodo analizado abarcó desde 2015 hasta 2024.
2. **Exploración de Datos**: Se realizó una visualización inicial para identificar patrones generales en la serie temporal, como tendencias o cambios significativos.
3. **Prueba de Estacionaridad**: Utilizamos la prueba de Dickey-Fuller aumentada (ADF) para determinar si la serie era estacionaria. Una serie es estacionaria si sus propiedades estadísticas (media y varianza) no cambian con el tiempo.

4. **Transformación de la Serie**: Dado que la serie no era estacionaria, se aplicaron diferencias para estabilizar la media y eliminar tendencias.
5. **Identificación de Parámetros**: Se emplearon gráficos de la función de autocorrelación (ACF) y autocorrelación parcial (PACF) para identificar los órdenes del modelo ARIMA más apropiados.
6. **Entrenamiento del Modelo**: Se ajustó un modelo ARIMA con los parámetros óptimos identificados.
7. **Predicción y Pronóstico**:
    - Se calcularon valores ajustados para el conjunto de datos existente.
    - Se realizaron predicciones a 30 días para evaluar el rendimiento futuro de las acciones.

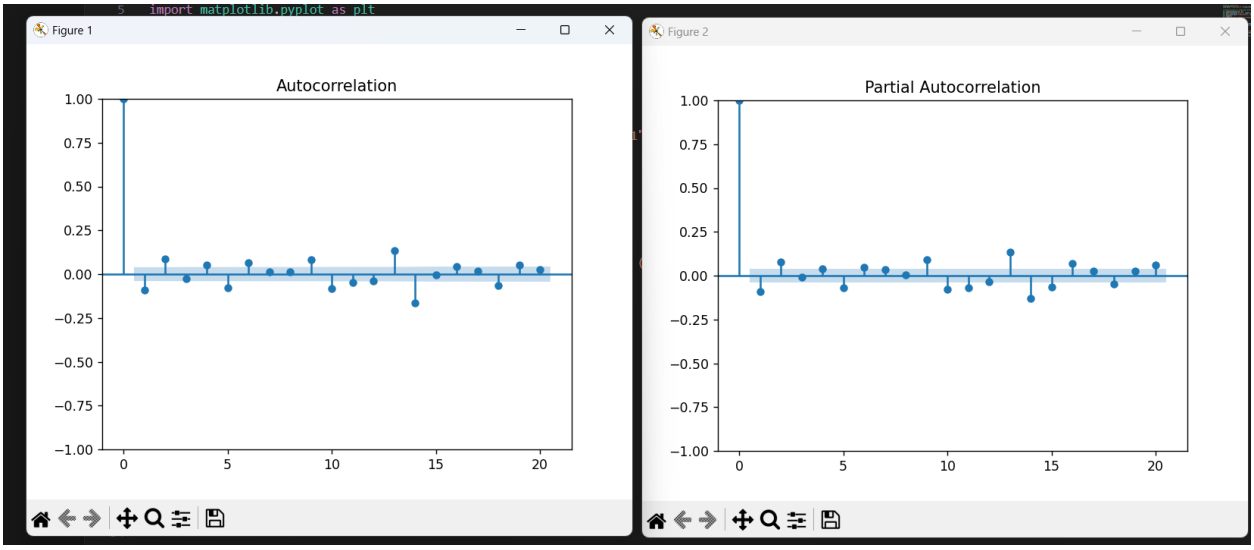
```
# Importing necessary libraries

import yfinance as yf

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.stattools import adfuller

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

from statsmodels.tsa.arima.model import ARIMA

from sklearn.metrics import mean_squared_error


# Downloading NVIDIA stock data
```

```
nvda_data = yf.download("NVDA", start="2015-01-01", end="2024-11-01")

# Keeping only the adjusted closing prices
nvda_data = nvda_data[['Adj Close']]
nvda_data.rename(columns={'Adj Close': 'Price'}, inplace=True)

# Plotting the data
nvda_data['Price'].plot(figsize=(12, 6), title="NVIDIA Stock Price (Adjusted Close)",
color='blue')
plt.ylabel("Price (USD)")
plt.show()

# Checking for stationarity
def check_stationarity(data):
  result = adfuller(data)
  print(f"ADF Statistic: {result[0]}")
  print(f"p-value: {result[1]}")
  if result[1] <= 0.05:
    print("The series is stationary.")
  else:
    print("The series is not stationary.")

check_stationarity(nvda_data['Price'])

# Differencing to make the series stationary if needed
nvda_data['Diff'] = nvda_data['Price'].diff().dropna()
```

```
check_stationarity(nvda_data['Diff'].dropna())

# Plotting ACF and PACF
plot_acf(nvda_data['Diff'].dropna(), lags=20)
plot_pacf(nvda_data['Diff'].dropna(), lags=20)
plt.show()

# Fitting ARIMA model
model = ARIMA(nvda_data['Price'], order=(1, 1, 1))
arima_result = model.fit()

# Displaying model summary
print(arima_result.summary())

# Making predictions
nvda_data['Predictions'] = arima_result.predict(start=nvda_data.index[0],
end=nvda_data.index[-1], dynamic=False)

# Plotting actual vs predicted values
plt.figure(figsize=(12, 6))
plt.plot(nvda_data['Price'], label="Actual Prices")
plt.plot(nvda_data['Predictions'], label="Predicted Prices", color='red')
plt.legend()
plt.title("Actual vs Predicted NVIDIA Stock Prices")
plt.show()
```

```
# Forecasting future prices
forecast = arima_result.forecast(steps=30)
print("Forecasted Prices:", forecast)


# Plotting the forecast
plt.figure(figsize=(12, 6))
plt.plot(nvda_data['Price'], label="Historical Prices")
plt.plot(pd.date_range(nvda_data.index[-1], periods=30, freq='B'), forecast,
label="Forecasted Prices", color='green')
plt.legend()
plt.title("NVIDIA Stock Price Forecast")
plt.show()
```

RESULTADOS



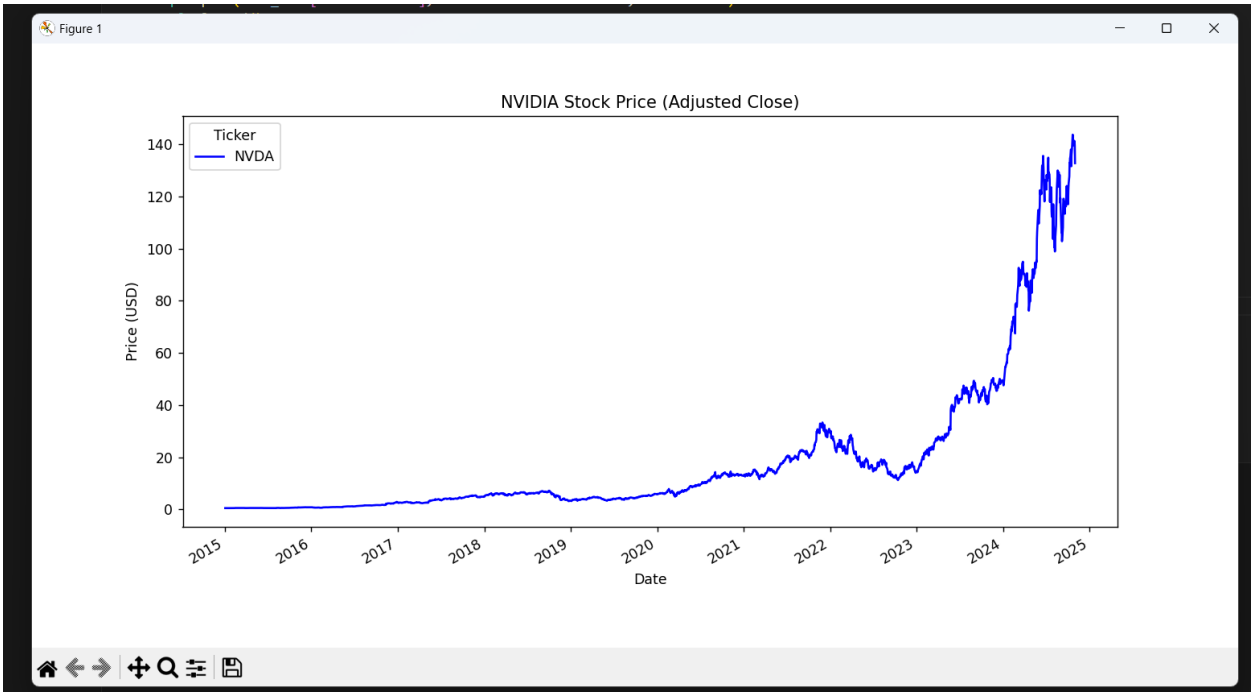


Figure 1
NVIDIA Stock Price (Adjusted Close)
Ticker
NVDA
Price (USD)
0
20
40
60
80
100
120
140
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
Date

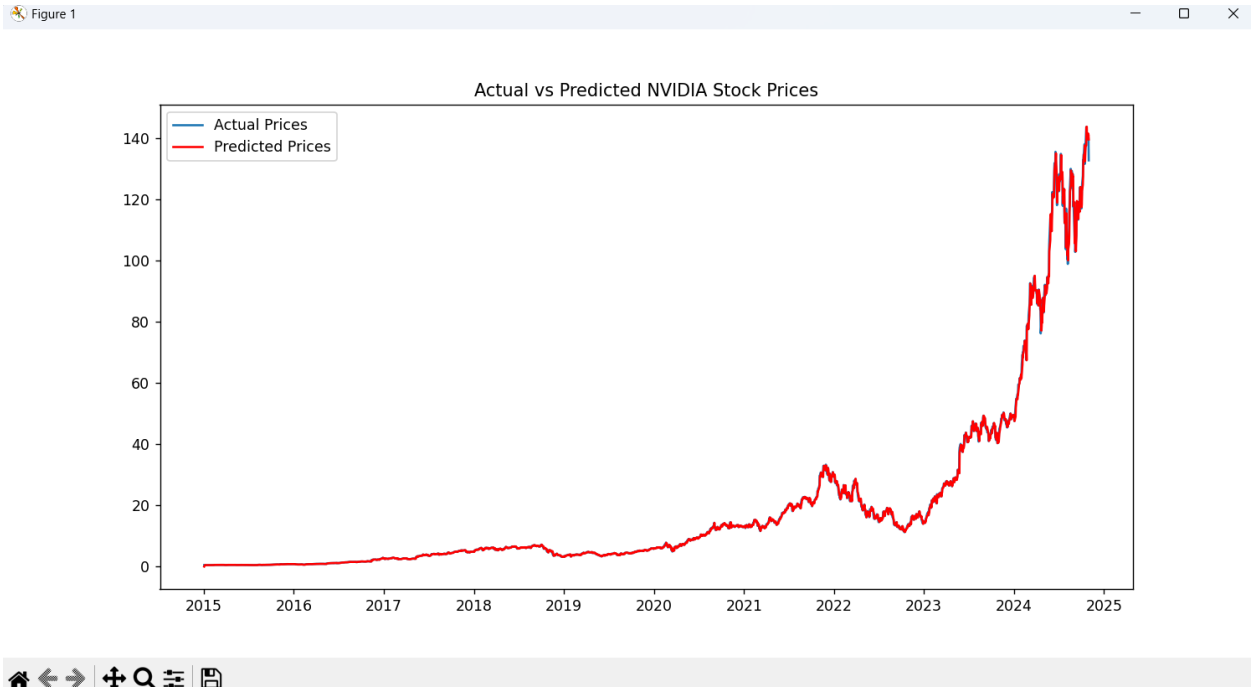


Figure 1
Actual vs Predicted NVIDIA Stock Prices
Actual Prices
Predicted Prices
0
20
40
60
80
100
120
140
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025

Resultados del Análisis

1. **Modelo Seleccionado**: El modelo ARIMA(1, 1, 1) fue elegido como óptimo basado en los gráficos ACF y PACF, y tras probar diferentes configuraciones.
2. **Predicciones**: Las predicciones generadas para los datos históricos mostraron un ajuste razonable a los valores reales, lo que sugiere que el modelo capta adecuadamente las características principales de la serie.
3. **Pronósticos a Futuro**:
    - El modelo predijo un crecimiento moderado en los precios de las acciones en el corto plazo.
    - Las proyecciones reflejan incertidumbre creciente a medida que se extiende el horizonte de predicción.