[Lastname, Firstname} Extra Credit Teslas Stock Prediction Model

October 7, 2023

[ ]:
```
import mplfinance as mpf
import pandas as pd
import numpy as np
from ta.volatility import BollingerBands
from ta.momentum import RSIIndicator
from ta.trend import SMAIndicator
from ta.trend import MACD
import yfinance as yf
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report


'''
This is only an example to demonstrates how machine learning can be applied to␣
 ↪financial data
to make predictions.

Real world trading decisions will use a combination of models, technical and␣
 ↪fundamental analysis,
as well as risk management strategies to make informed decisions in financial␣
 ↪markets.

'''

# Load Tesla stock data with an extended date range
tesla = yf.download('TSLA', start='2022-01-01', end='2023-12-31',␣
 ↪progress=False)

# Calculate Bollinger Bands
indicator_bb = BollingerBands(close=tesla['Close'], window=20, window_dev=2)
tesla['BB_upper'] = indicator_bb.bollinger_hband()
tesla['BB_middle'] = indicator_bb.bollinger_mavg()
tesla['BB_lower'] = indicator_bb.bollinger_lband()

# Calculate RSI
rsi_indicator = RSIIndicator(close=tesla['Close'], window=14)
```

```
tesla['RSI'] = rsi_indicator.rsi()

# Calculate moving averages (SMA)
sma_50 = SMAIndicator(close=tesla['Close'], window=50)
tesla['SMA_50'] = sma_50.sma_indicator()

sma_200 = SMAIndicator(close=tesla['Close'], window=200)
tesla['SMA_200'] = sma_200.sma_indicator()

# Calculate MACD
macd = MACD(tesla['Close'], window_slow=26, window_fast=12, window_sign=9)
tesla['MACD'] = macd.macd()

# Data Preprocessing
tesla.dropna(inplace=True)
tesla['Price_Up'] = np.where(tesla['Close'].shift(-1) > tesla['Close'], 1, 0)

# Features and target variable
X = tesla[['SMA_50', 'SMA_200', 'RSI', 'MACD', 'BB_upper', 'BB_middle',␣
 ↪'BB_lower']]
y = tesla['Price_Up']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Train a Decision Tree classifier
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')
print(classification_report(y_test, y_pred))

# Visualize predictions
tesla['Predicted_Up'] = clf.predict(X)
mpf.plot(tesla, type='candle', title="Tesla Stock Price Predictions",␣
 ↪addplot=[mpf.make_addplot(tesla['Predicted_Up'], panel=1, secondary_y=True)])
```

[ ]: