# STOCK MARKET PREDICTION

## USING ML

Paper: (Ishita Parmar 2020)Stock Market Prediction Using Machine Learning

# CONTENTS



# INTRODUCTION

- **Potential of Stock Prediction:** Accurate stock prediction is key to maximizing profits and minimizing risks. While financial markets are chaotic, proper analysis can uncover predictable patterns.

- **Role of Machine Learning**: ML techniques like Linear Regression, LSTM models and many others are increasingly favored for their ability to yield predictions that are close to actual market values.

- **Models Chosen:** We employ Ridge Regression for basic linear trends and LSTM networks to capture non-linear, sequential dependencies in the data.

- **Our Work**: Our work implements the methodologies of in paper in finer way, analyze the results, and draws conclutions from gained results.

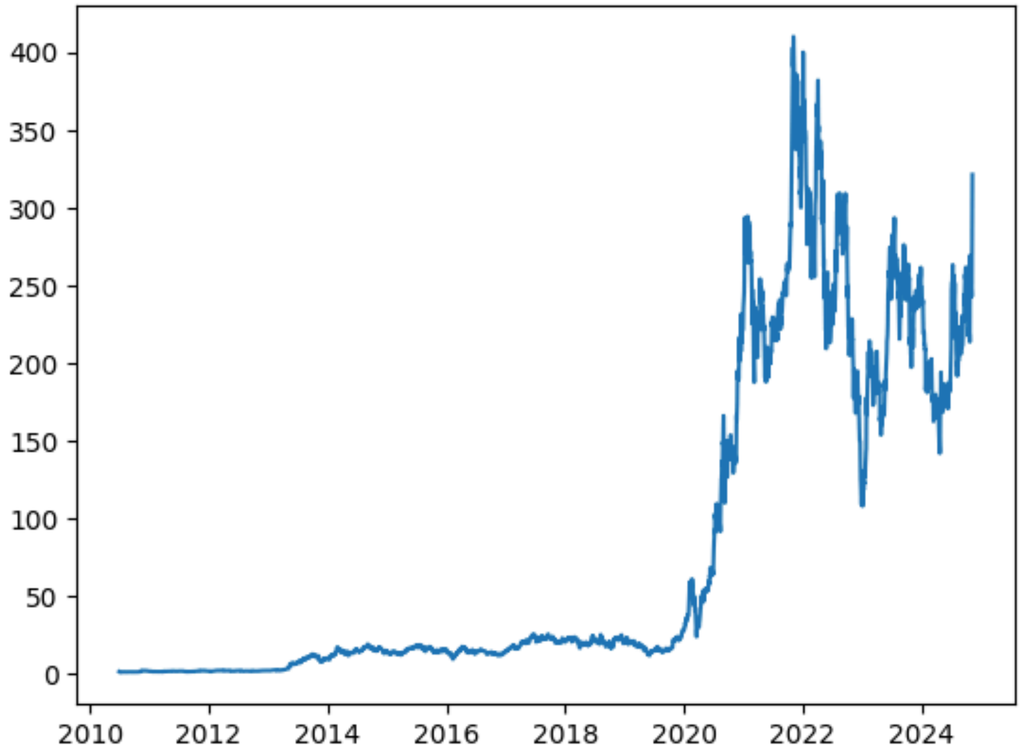
# OBJECTIVES

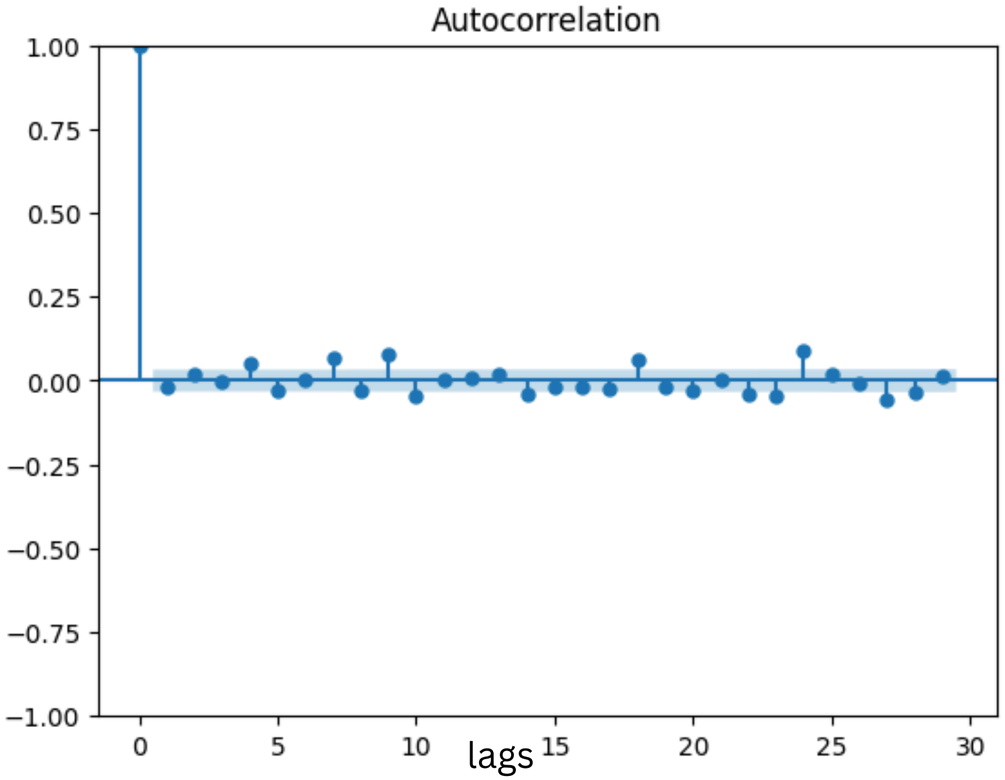
- **Develop Predictive Models:** Implement and compare machine learning models, including Ridge Regression and LSTM networks for to forecast Tesla’s stock prices.
- **Analyze Data Quality and Preprocessing:** Conduct Augmented Dickey-Fuller test to check stationarity and transform data to get features and labels by applying shift.
- **Evaluate Model Performance:** Assess models using metrics like Mean Squared Error (MSE) and R-squared to understand their predictive accuracy and reliability.
- **Utilize Time-Series Analysis**: Employ LSTM networks to capture long-term trends and trying extrapolation to predict completely new values.

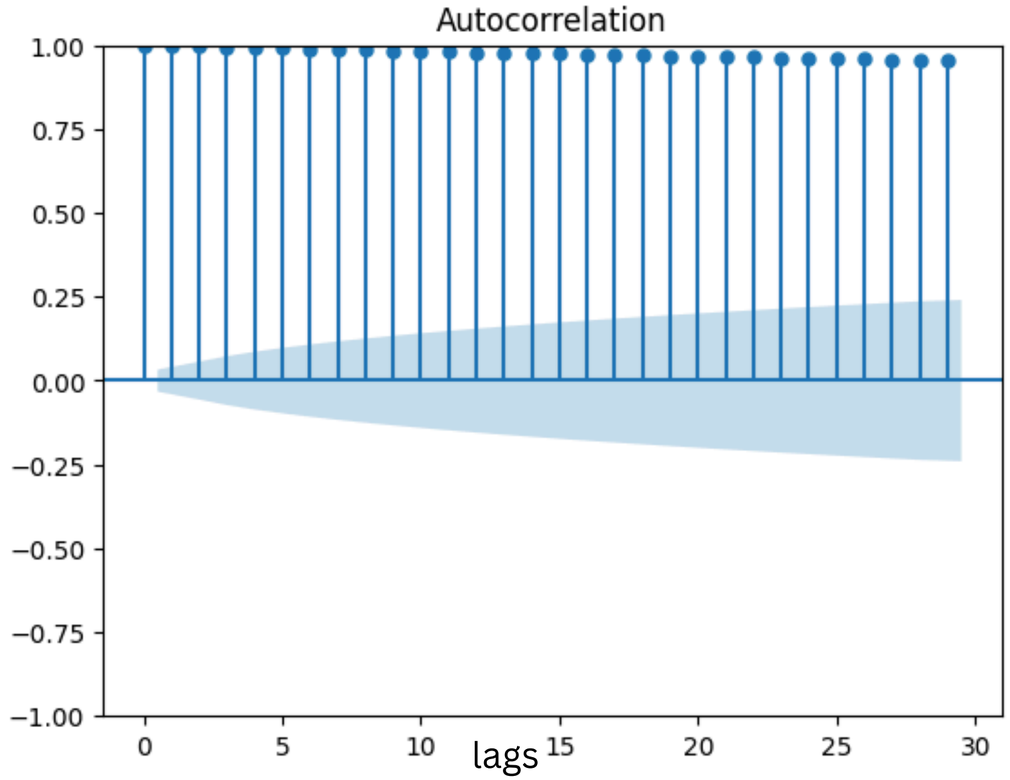
# DATA

## Source: Yahoo Finance

- Python contains a library named **yfinance** from where have imported our data. It keeps on updating it's data everyday from various renowned websites.
- Ticker used for importing data from library is Tesla's stock is **TSLA**.
- Image shows close price of Tesla stock from 2010-Till date.
- We perform Augmented Dickey-Fuller Test to check whether stationarity (trend) is present in data or not.
- If ADF-statistic is less than threshold value then we reject null and say non-stationarity is present.
- Non-stationarity is removed by differencing the dataset.






- Presence of stationarity after applying differencing.
- Image shows no correlation of stock value with previous 30 days' value.

- Presence of non-stationarity before applying differencing.
- Image shows high correlation of stock value with previous 30 days' value.

# LINEAR REGRESSION

- **Linear regression** is a supervised learning algorithm used for predicting a continuous outcome based on the relationship between independent variables (features) and the dependent variable.
- The model fits a straight line through the data points by minimizing the error between the predicted and actual values.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \varepsilon$$

  - **y** : Predicted output
  - **β0** : Intercept
  - **β1, β2, …, βn**: Coefficients (weights assigned to input features)
  - **ε** : Error term, which accounts for the unexplained variance in the data

- The model fits a straight line through the data points by finding the best coefficients β that **minimize the error between the predicted and actual values**.
- This is achieved using **Ordinary Least Squares (OLS)**, a method that minimizes the sum of squared residuals:

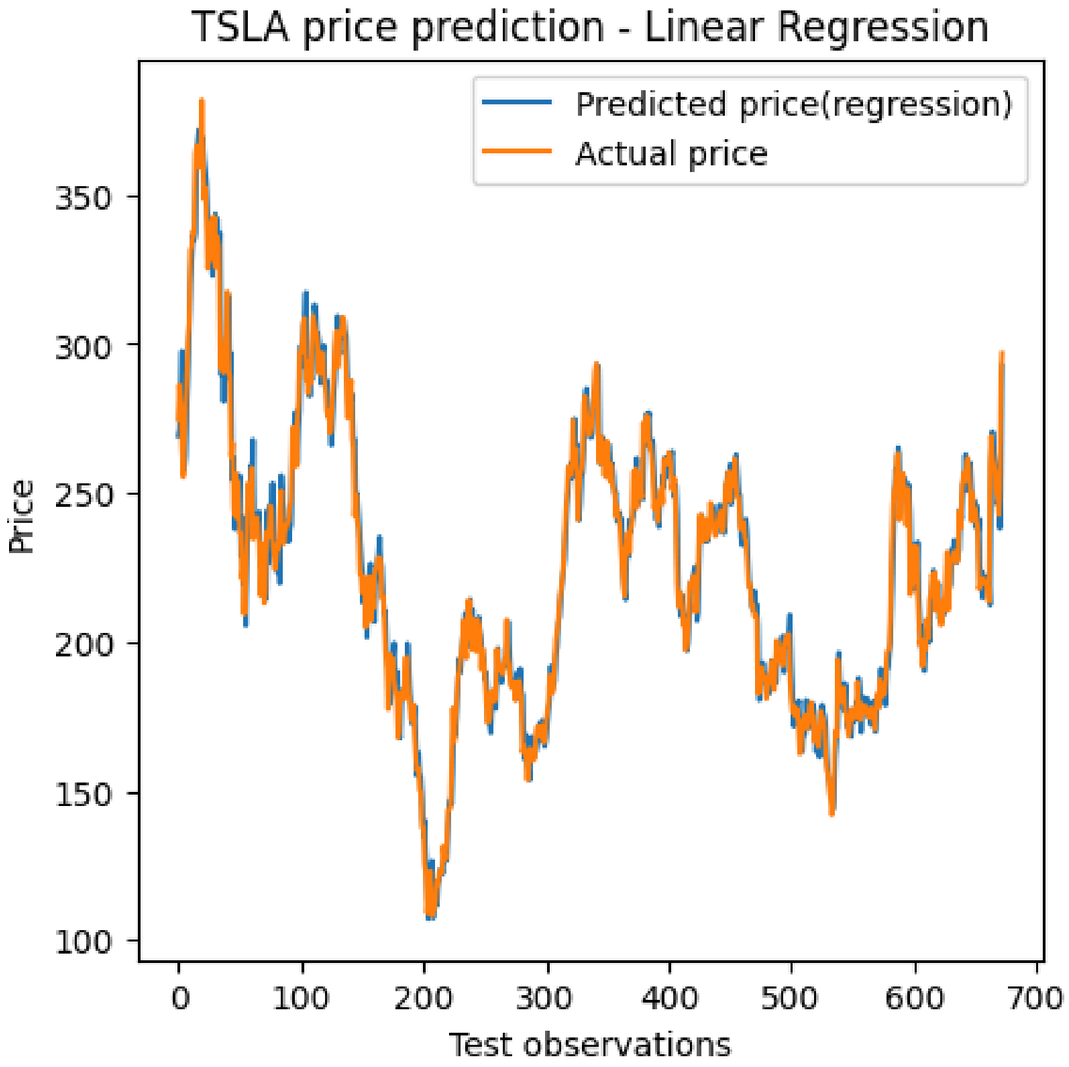
$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

# Limitations of Linear Regression

- **Overfitting** occurs when a machine learning model **learns the training data too well, including the noise and random fluctuations**. Instead of capturing the underlying patterns, the model becomes overly complex and sensitive to the specific examples in the training set.

- In regression, an overfitted model might create a **curve that follows every data point exactly** rather than a smooth trend.

- **Regularization** helps address this by adding a penalty to the model's complexity, usually measured in terms of the magnitude of its weights or parameters.



# L2 Regularization (Ridge)

- **Regularization** is a technique used in machine learning to **prevent models from overfitting** to training data, which can improve the model's performance on new, unseen data.

- Regularization modifies the objective function that the model tries to optimize by adding a regularization term. This term **penalizes larger weights**, effectively discouraging the model from relying too heavily on any single feature or becoming too complex.

- **L2 regularization**, also known as **Ridge** regularization, a penalty proportional to the square of the magnitude of the coefficients (weights) is added to the loss function. This is controlled by a hyperparameter, often called λ or alpha:

$$y = w_1x_1 + w_2x_2 + \cdots + w_nx_n + \varepsilon$$

$$\text{Loss} = \text{MSE} + \lambda * \Sigma w^2$$

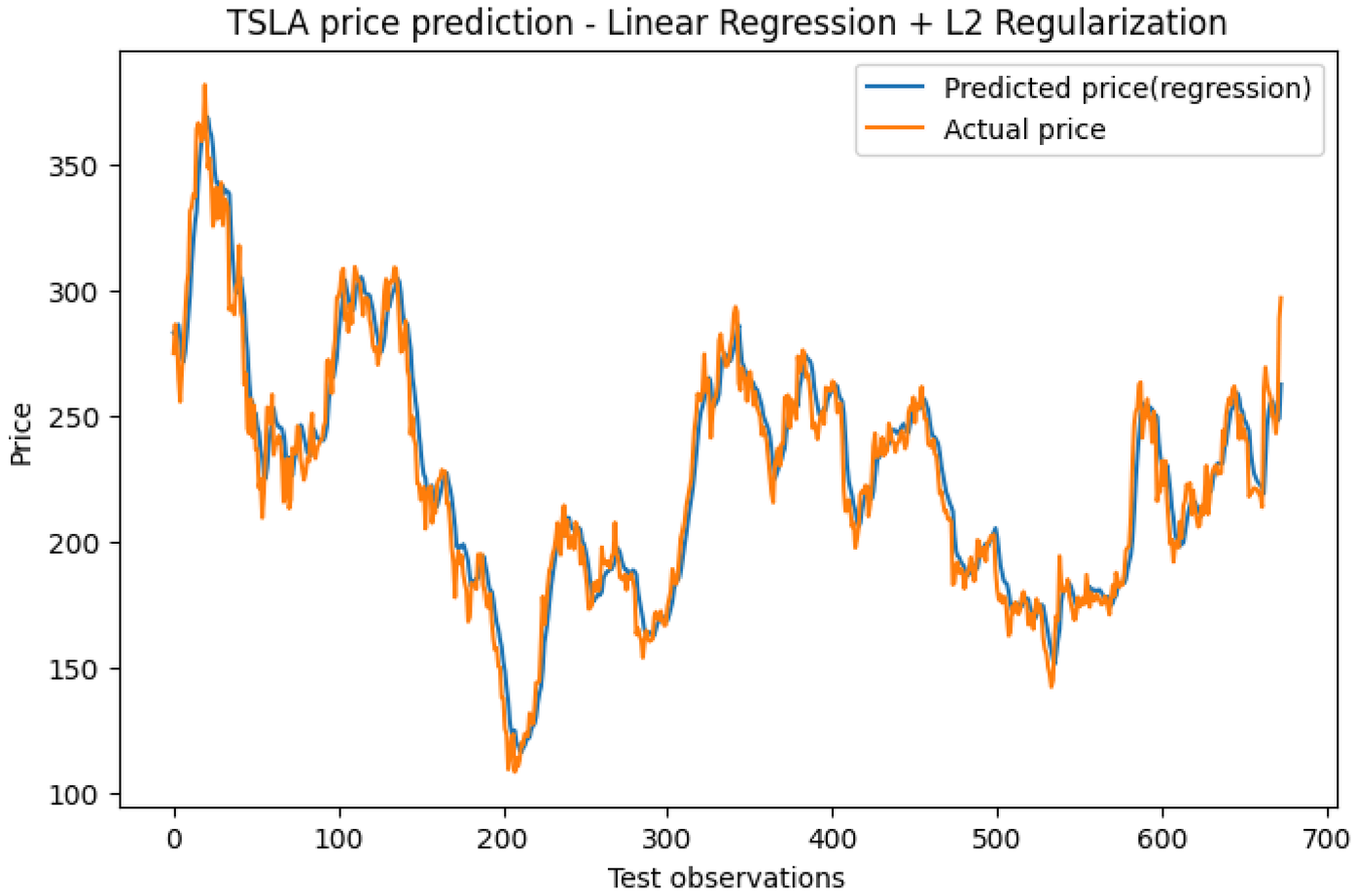
- L2 regularization encourages weights to be small but doesn't necessarily zero them out. This makes it useful for **reducing the impact of less important features** without removing them entirely.

# Ridge Regression (L2 - Regularization)

- Here is the graph comparing the actual stock prices against the predicted prices using the Linear Regression model with L2 regularization.

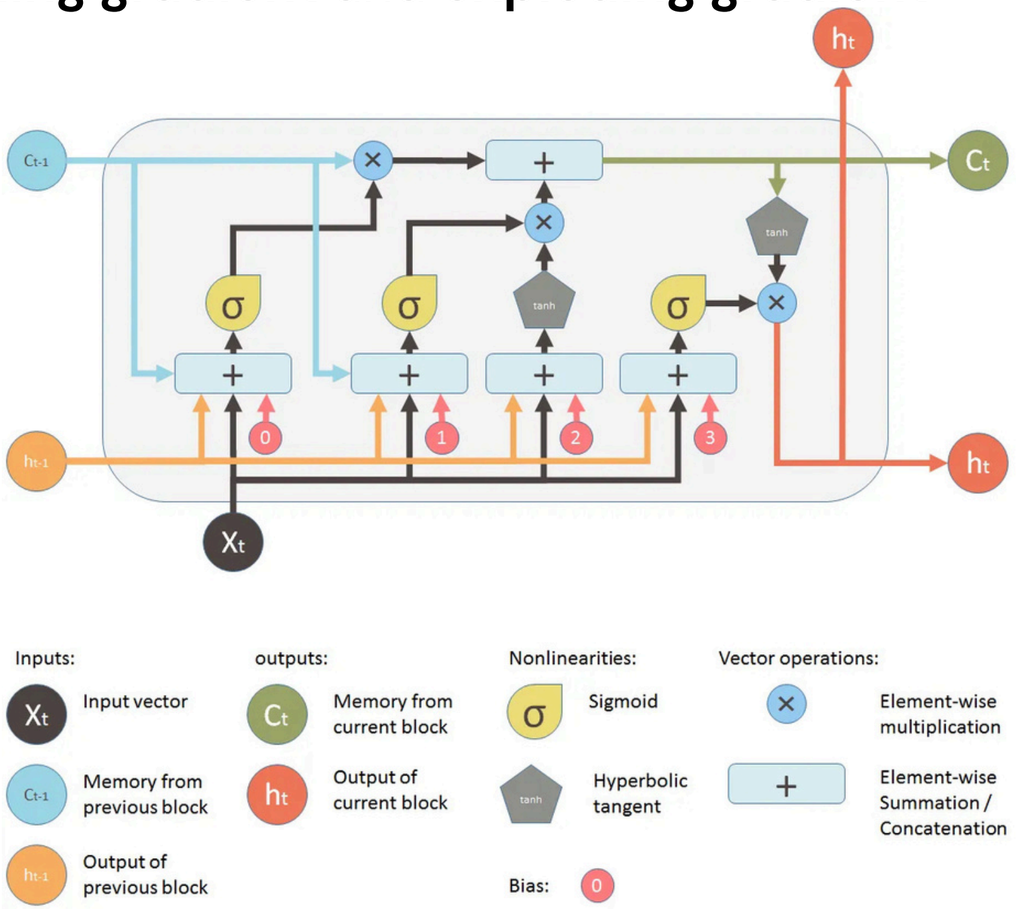
- The model's performance is quantified with an **MSE of 135.07** and **R-squared value of 0.94** indicates that 94% of the variance in the data is captured by our model.

- Linear regression, even with regularization, **may struggle with more complex, nonlinear relationships** and temporal dependencies in the data—factors that are common in time-series data.



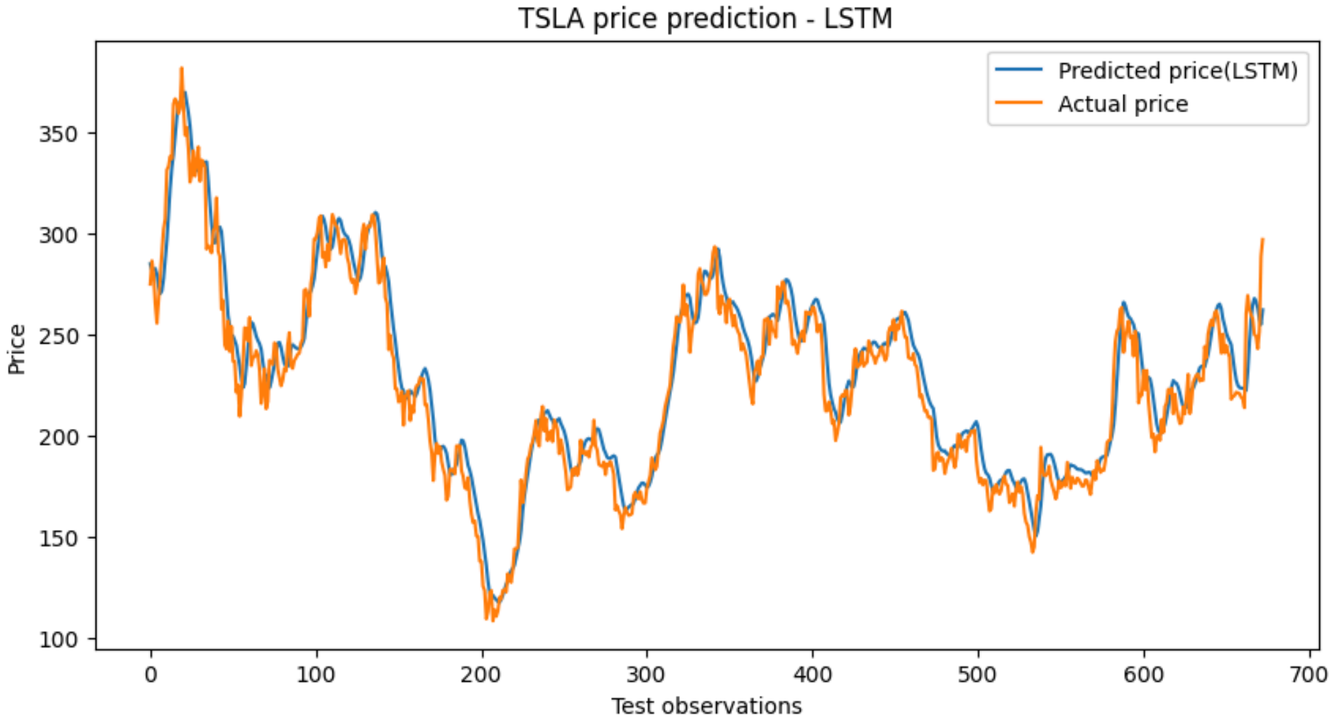
# LSTM MODEL

- **LSTMs** are a specialized type of RNN architecture designed to tackle a specific challenge of remembering information over extended periods.
- Two significant issues with **RNNs** are the **vanishing gradient** and **exploding gradient** problems.
- This issue arises due to the reuse of the same parameters in RNN blocks at each step.
- LSTM solves this problem by using **gated mechanism** which makes it a bit complex too.
- Since we need LSTM network to learn the pattern or trend in stock data, we have used data as it is without applying differencing.
- Image on the right shows a LSTM cell with input and output.



# Approach

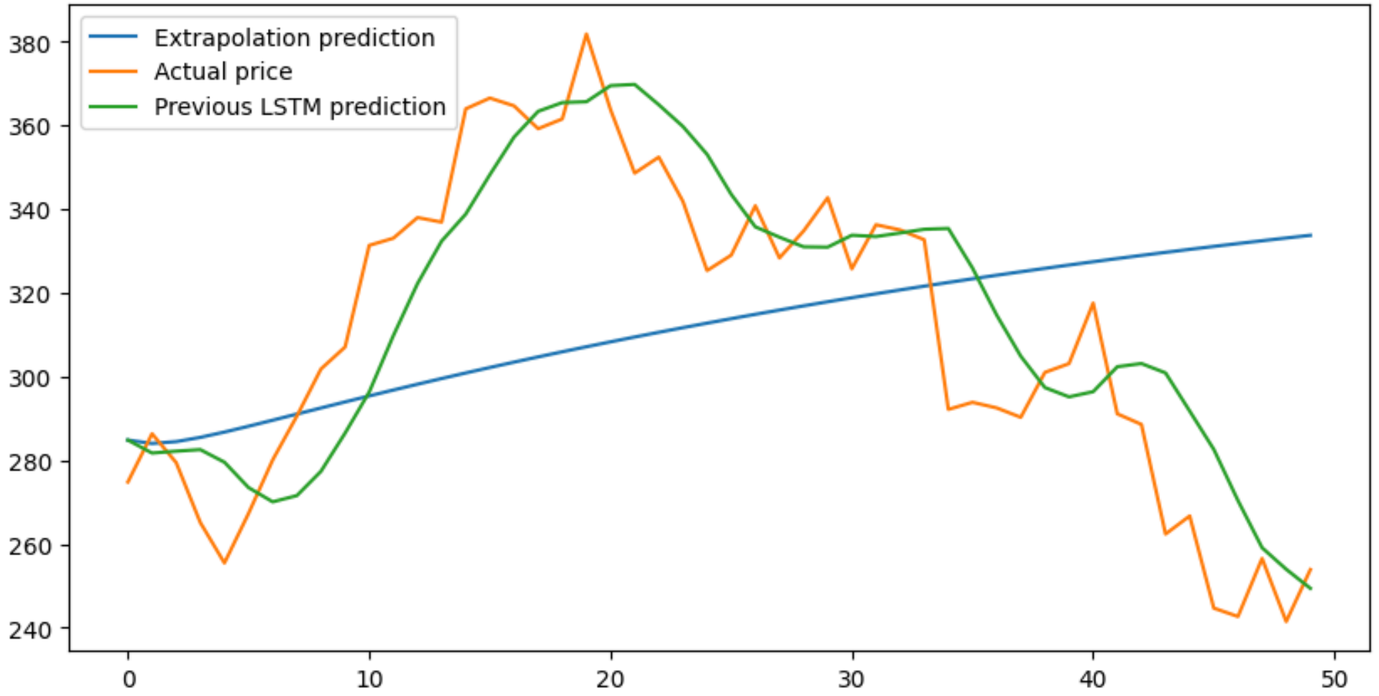
- The model uses the last 50 days of stock prices as input features and predicts 51st day's price as output label.
- Overfit is prevented by using Dropout of 20%.
- The model's performance is quantified with an **MSE of 153.6** and **R-squared value of 0.93.**




- Getting a lesser R-square than LR does not mean that model performs bad as R-squared should never be taken as a metric to judge a model **(High R-sq. == Overfit)**.
- Our model thus makes better generalization and also learns non-linear patterns in stock data.
- **Let's test our model further.....**

# Extrapolation using LSTM

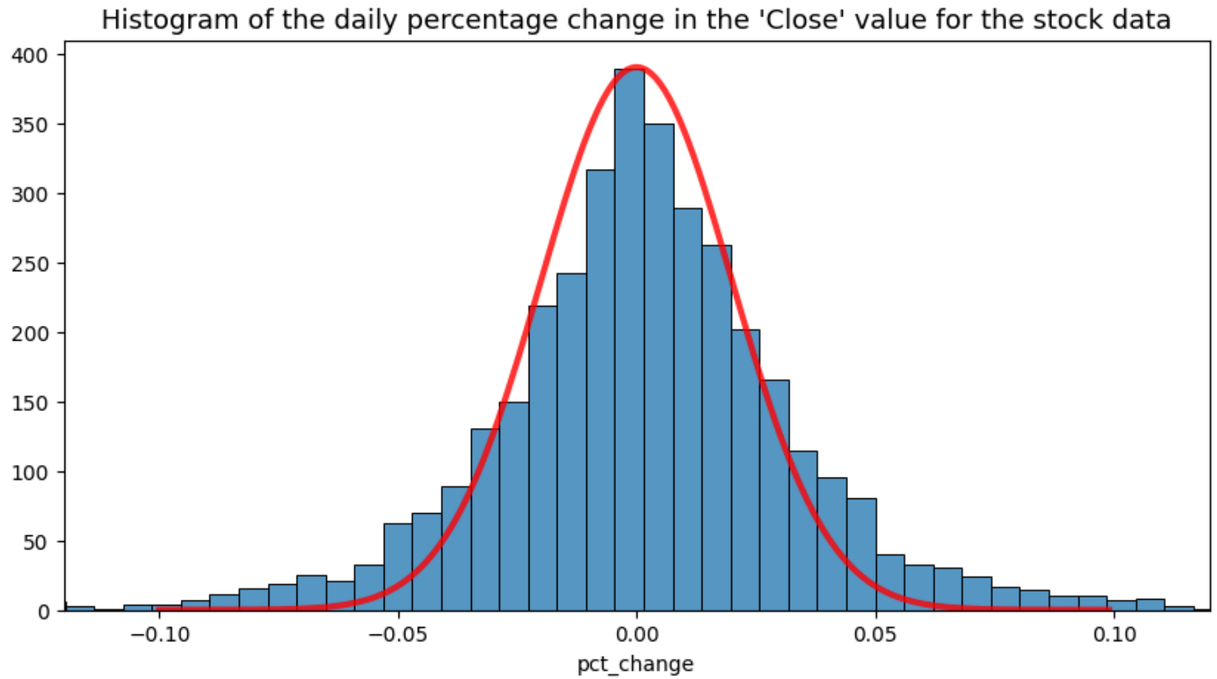
- In this section we would like to experiment with out LSTM model to **predict future values using it's own predictions**.
- That is, we won't use true past data to predict next days' value, rather, we will use just a single observation (of 50 lags).
- We will then drop the 1st element of input and append the predicted output in input. (sliding window kind of)
- Repeat this for 50 predicitions.




- **Notice:** one important thing that the **trend of actual price** is **captured afterwards** by **previous LSTM predictions**.
- **Extrapolation output** $\neq$ **LSTM output**

# WHAT WENT WRONG IN EXTRAPOLATION??

- Let us subtract yesterday's value from today's value and convert it into percentage. {prop. to Y(t)-Y(t-1)}
- Plot this values (say error) in form of histogram as shown in right image.
- We can conclude that tomorrow's prediction is equal to today's value plus some error drawn from a Gaussian curve, i.e. **Y(t) = Y(t-1) + e(t)**




- The above equation is exactly equation for a **RANDOM WALK PROCESS** (a non-stationary process).
- Also we worked on non-stationary data (shown by ADF Test), which can be analyzed here as well.
- So, we can say that **LSTM predictions** are nothing, it's just **previous day's value + error**.

# CONCLUSION & FINDINGS

- This paper was an attempt to determine the future prices of the stocks with greater accuracy and reliability using machine learning techniques.
- Both the techniques have shown an improvement in the accuracy of predictions, thereby yielding positive results with the LSTM model proving to be more efficient.
- The results are quite promising and has led to the conclusion that it is possible to predict stock market with more accuracy and efficiency using machine learning techniques.
- We observe that we can predict stock price correctly for 1-2 days in future as shown in extrapolation part as there might be various random things that may happen in market which can't be predicted by ML techniques.
- In the future, the accuracy of the stock market prediction system can be further improved by utilizing a much bigger dataset than the one being utilized currently.
- Furthermore, other emerging models of Machine Learning could also be studied to check for the accuracy rate resulted by them.
- Sentiment analysis through Machine Learning on how news affects the stock prices of a company is also a very promising area.

# THANK YOU

Patel Rhut Dipakbhai (220756)
Daxal Prajapati (220782)
Pragati Rathi (220780)
Nitish Saroj (220735)
Priyanka Choudhary (220821)

**Link**: https://github.com/hector756/Stock-Price-Prediction