Detailed Breakdown of the Machine Learning Model Workflow

1. Sentiment Analysis (NLP Model)

Overview:

- Input: News articles, social media posts, or any text data related to TSLA.
- Process: The NLP (Natural Language Processing) model processes this text to classify whether the sentiment toward TSLA is positive, neutral, or negative.
- Output: The result is a sentiment score (e.g., +0.8 for positive sentiment, -0.3 for negative sentiment) that quantifies the overall market sentiment.

How the Algorithm Works:

- Text Preprocessing: First, the text data is cleaned and preprocessed. This includes:
    - Tokenization: Splitting the text into words.
    - Removing Stopwords: Removing common words like “the” or “is.”
    - Lemmatization/Stemming: Reducing words to their base forms (e.g., "running" becomes "run").
- Feature Extraction: The preprocessed text is then converted into numerical features using methods such as:
    - TF-IDF (Term Frequency-Inverse Document Frequency): Weighs the importance of a word in the document relative to its frequency across all documents.
    - Word Embeddings (e.g., Word2Vec, BERT): Maps each word to a high-dimensional vector based on its context.
- Classification: The features are then passed to a classification model (e.g., Naive Bayes, Support Vector Machines, or a neural network). The classifier predicts whether the sentiment in the text is positive, negative, or neutral.
- Output: The final output is a sentiment score, which is used as an input for both the Boosting Algorithm and the Reinforcement Learning Agent.

2. Boosting Algorithm (XGBoost)

Overview:

- Input:
    - Technical indicators: Moving averages (50-day, 200-day), RSI, Bollinger Bands, etc. (This will come from Odi and Yagiz)
    - Fundamental data: Financial metrics like P/E ratio, earnings, balance sheet data (Odi and Yagiz)
    - Sentiment score: The sentiment score from the NLP model is used as an additional feature.

- Process: The Boosting algorithm combines these inputs to predict the probability of TSLA's stock trending upward or downward over the short term (e.g., the next 5 days).
- Output:
    - Trend probability (e.g., a 75% probability that TSLA will rise).
    - Feature importance: Which inputs (e.g., sentiment score, moving averages) were most important in predicting the trend.

How the Algorithm Works:

- Weak Learners: Boosting builds many weak learners, typically small decision trees, where each learner focuses on correcting the errors of the previous ones.
- Training Process:
    - The first weak learner makes predictions based on the input features. Since it's a weak learner, it likely makes a lot of errors.
    - The algorithm then calculates residuals (errors) and trains the next weak learner to focus on correcting these errors.
    - This process is repeated, with each learner improving the model's accuracy by reducing errors.
- Final Prediction:
    - The final model aggregates the predictions from all weak learners. Each learner's prediction is weighted, and the combined output is a trend probability (e.g., 75% chance of TSLA trending upward).
- Feature Importance:
    - Boosting models like XGBoost or LightGBM can rank features by their importance in predicting the trend. For example, if the sentiment score is one of the key drivers of a price increase, it will be ranked higher than other features.

3. LSTM Model (Long Short-Term Memory)

Overview:

- Input:
    - Historical price data: A sequence of past stock prices (e.g., over the last 60 days).
    - Volume data: The trading volume for each day, which can give insight into market activity.
    - Sentiment score: The current sentiment score from the NLP model (might include not sure yet)
- Process: The LSTM model processes this sequential data to predict the next day's stock price or short-term future prices (e.g., the next 1–5 days).
- Output:
    - Next day's predicted price (e.g., TSLA's price will rise from $750 to $765 tomorrow).

How the Algorithm Works:

- Sequential Data: LSTM models are designed to handle time-series data, where the order of data points is crucial (e.g., stock prices over time).
- Memory Mechanism:
    - LSTM stands out because of its ability to remember long-term dependencies. It has a cell state that carries information from previous time steps, allowing the model to capture longer trends in the data.
    - The Forget Gate decides what information from the previous time steps should be kept or discarded.
    - The Input Gate updates the cell state with new information (e.g., today's stock price).
    - The Output Gate generates the next hidden state, which informs the next LSTM cell.
- Training:
    - The LSTM model is trained using backpropagation through time to adjust its internal weights. The model is trained to minimize the difference between its predicted price and the actual price.
- Final Prediction:
    - After training, the LSTM model takes in recent historical data (e.g., the last 60 days of stock prices) and outputs a predicted price for the next day.

4. Reinforcement Learning (RL Agent)

Overview:

- Input:
    - Sentiment score: From the NLP model.
    - Trend probability: From the Boosting algorithm.
    - Next day's predicted price: From the LSTM model.
    - Portfolio state: Information about the current portfolio (e.g., cash available, how many TSLA shares are held).
- Process: The RL agent uses these inputs to decide the optimal trading action: buy, sell, or hold TSLA shares.
- Output:
    - Trading action (e.g., buy 10 shares of TSLA at $750).

How the Algorithm Works:

- Environment and Agent:
    - In reinforcement learning, an agent (the RL model) interacts with an environment (the stock market) by taking actions (buy, sell, hold) based on the current state (inputs from the LSTM, Boosting, and Sentiment models).
    - The agent's goal is to maximize long-term rewards, which in this case is the profit from trading TSLA.
- Reward Function:
    - The agent receives a reward based on the outcome of its actions:

        - Positive reward for profitable trades (e.g., if the agent buys TSLA and the price rises).
        - Negative reward for losing trades.
- Policy Learning:
    - The RL agent learns a policy (a strategy for selecting actions) through trial and error. Over time, it learns which actions (buy, sell, hold) lead to the best rewards.
    - It uses techniques like Deep Q-Learning (DQN) or Policy Gradient methods to update its strategy and make better decisions in the future.
- Exploration vs. Exploitation:
    - The agent must balance exploration (trying new actions) and exploitation (using actions it knows are successful) to continuously improve its trading performance.
- Continuous Learning:
    - After each trade, the RL agent updates its policy based on the reward it receives (profit or loss from the trade). This helps it make more informed decisions in future trades.

5. Trading Action and Execution

Overview:

- Input: The trading signal generated by the RL agent (buy, sell, hold).
- Process: The trading signal is sent to the trading platform via an API (e.g., Alpaca, Interactive Brokers).
- Output: The system executes the trade in real-time.

Trade Execution:

- The trade execution platform takes the RL agent's decision and executes it in the live market.
    - If the RL agent decides to buy 10 shares of TSLA at $750, the system places a buy order.
    - If the agent decides to sell, the system sells the shares.

6. Feedback and Continuous Learning

Overview:

- After each trade, the RL agent receives feedback in the form of a reward (profit or loss).
- Process: The RL agent updates its policy based on this feedback to improve future trading decisions.
- Continuous Learning: Over time, the RL agent gets better at identifying profitable trades based on the inputs from the other models.

Final Summary of the Workflow:

1. Data Collection: Real-time stock prices, technical indicators, and sentiment data are collected.
2. Model Predictions:
    - Sentiment Analysis provides a sentiment score.
    - LSTM predicts short-term price movements.
    - Boosting predicts stock trends and identifies important features.
3. Reinforcement Learning: Combines the model predictions and makes optimal trading decisions (buy, sell, hold).
4. Trade Execution: RL agent's decision is executed on the trading platform.
5. Feedback Loop: The RL agent learns from the results of each trade and refines its strategy over time.