

WIDS : Market Mood & Moves : Sentiment-driven stock prediction

Week 1

Project Overview

This project focuses on understanding how market sentiment from news can influence stock price movements. I explored how data science, NLP, and finance come together to build the foundation of a sentiment-driven stock prediction system.

Natural Language Processing Fundamentals

NLP helps convert unstructured text, such as news articles and headlines, into a form that machines can understand. In this project, NLP was used to extract meaningful signals from financial news and study how language reflects market sentiment.

Text Preprocessing

Tokenization : splitting text into words

Stopword removal : removing common but uninformative words

Lemmatization : reducing words to their base form

Text preprocessing cleans raw text and removes noise so that sentiment models focus on meaningful words. This step is crucial before applying any NLP or sentiment analysis technique.

API-Based Data Collection

Tools used:

- NewsAPI for financial news headlines
- Finance for historical stock price data

APIs were used to collect real-world, live data instead of relying on static datasets. This helped simulate realistic market analysis workflows and understand how financial data pipelines work in practice.

Sentiment Analysis

VADER Sentiment Analyzer

Characteristics are Lexicon-based, Rule-driven, Suitable for short text

VADER was used as a baseline sentiment analysis tool. It works well for short texts like headlines and provides sentiment scores that are easy to interpret.

FinBERT for Financial Text

FinBERT is a Domain-specific language model with context-aware embeddings, trained on financial text.

FinBERT is better than traditional sentiment models by understanding financial terminology and context. It assigns sentiment more accurately to finance-related text compared to general-purpose NLP models.

Quantitative Finance Fundamentals

Slippage Modeling

Slippage refers to the difference between the expected price and the actual execution price. I learned that fast markets, large orders, or low liquidity can cause trades to execute at worse prices, reducing profits.

Transaction Costs Analysis (TCA)

Even small fees like brokerage, taxes, and exchange charges can significantly impact performance. This concept taught me why frequent trading strategies may look profitable on paper but fail in real life.

Liquidity & Execution Probability

Liquidity measures how easily a stock can be bought or sold without affecting its price. I learned the trade-off between **market orders** (guaranteed execution, uncertain price) and **limit orders** (fixed price, uncertain execution).

Performance Metrics

I learned that evaluating a strategy requires measuring **risk along with returns**:

- **Maximum Drawdown (MDD)**: Shows the worst loss from peak to bottom — helps understand downside risk.

- **CAGR:** Represents the average annual growth of an investment over time.
- **Win/Loss Ratio & Profit Factor:** Measure consistency and overall profitability of trades.
- **Alpha & Beta:** Explain how a strategy performs relative to the market and how risky it is.
- **Sharpe Ratio:** Measures return per unit of risk — higher means better risk-adjusted performance.
- **Sortino Ratio:** Similar to Sharpe but focuses only on harmful (downside) risk.
- **Calmar Ratio:** Compares returns with maximum drawdown, useful during volatile markets.

Week 2

Week 2 Theoretical mastery

Q) Explain why Static Embeddings fail on the word "Bank" and the importance of BERT architecture.

Ans) Static embeddings like Word2Vec maps a word like "Bank" to one fixed vector but some words including bank can have multiple meanings and there the issue arises, Word2Vec takes a weighted average of those two vectors corresponding to different meanings.

BERT uses dynamic embedding, here embedding is a function. It goes through the entire sentence and understands the context. Hence, in financial text, BERT interprets bank as financial entity while Word2Vec will confuse.

Q) Draw the 3 components of the BERT input embedding.

ans) **Final Input = Token + Segment + Position embeddings**

Token Embeddings WordPiece breaks unknown words into subwords, solving out-of-vocabulary problem

Segment embeddings these tell which sentence a word belongs to

Position embeddings they tell about BERT word order, which word is positioned where, usually transformers don't understand sequence by themselves

Q) Explain the 80-10-10 Masking Rule

ans) 80-10-10 masking rule stands for

80 % mask : replacing word with MASK

10% random word : teaches the model to trust context

10% original word : prevents model from thinking that every input is wrong

Week 2 FinBERT specifies

Q) Understand the 3-stage training pipeline.

ans) Stage 1 : FinBERT is BERT, which is specially trained for finance

So here we do general language pre training like it learns english grammar and structure

Stage 2 : Domain Adaption, Trained on finance documents, learns finance specific meanings

Stage 3 : Fine tuned for Sentiment analysis, learns to label text as positive, negative and neutral

Q) Define "Domain Adaptation" in the context of NLP.

ans) Domain Adaption is simply taking a general language model and retraining it on text from specific domain like finance

Example : FinBERT is BERT trained in finance

BERT understands bank broadly but FinBERT understands mainly as financial institution

Q) Explain why we use TRC2-Financial for pre-training and Financial PhraseBank for fine-tuning.

ans) We use TRC2-Financial for pre-training because it contains large volumes of real financial news and reports. This helps the model learn financial language, terminology, and context at a broad level.

We use Financial PhraseBank for fine-tuning because it has short financial sentences with clear sentiment labels (positive, negative, neutral). This helps the model learn how to correctly classify sentiment, which is the final task we care about.

*** I had also written some concepts in jupyter notebooks markdown cells and as comments

