Navigating the LNG Markets

A Quantitative and Sentiment-Driven Approach

Zhou Ziqi, Wang Zhiyuan, Xiao Yunhan, Li Shuai, Zhang YiQing

National University of Singapore

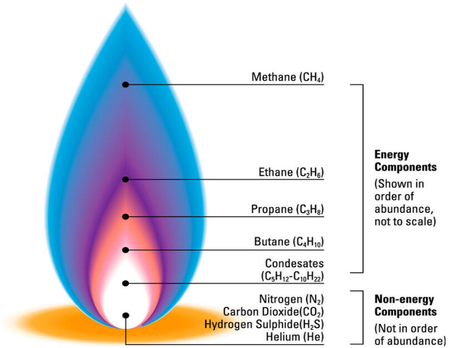
Spring Term 2024

Introduction

1. Introduction
2. Methodology
    1. Problem Statement
    2. Model Architecture
    3. Data Visualization (if time allows)
    4. Sentiment Analysis

What is LNG?

- Liquefied Natural Gas
- A Commodity product
- Liquid state through cooling



Why We Choose LNG?

1. Factors that affect LNG price
   - Weather, Fundamental Changes, Geopolitical Events, Headlines, etc.
2. Growing Market
   - Increasing demand for greener energy sources.
3. Huge Volatility
   - More volatile compared with other products.
4. Market Efficiency
   - Relatively young and less studied compared to oil.

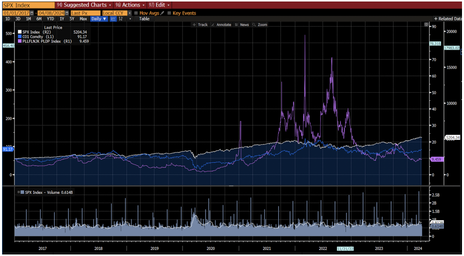


Figure: Huge Volatility in LNG Market

Our Data Source

- Bloomberg and Intercontinental Exchange, Inc (ICE)
- Platts:
  - S&P Global Commodity Insights
  - provider of energy and commodities information
  - a source of benchmark pricing in the physical commodity markets.
- Kpler:
  - provider of real-time transparency on commodity markets
  - specializing in **tracking** the global flow of commodities to deliver.

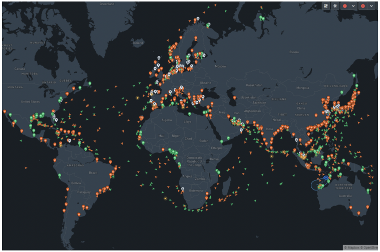


Figure: Example of Kpler Data Source

Problem Statement

- Problem Statement
  - Construct a robust quantitative trading strategy for the LNG market by designing an architechture.
  - Not to build handmade features, nor to just try different algorithms.
- Cruel Fact: Accuracy $>$ 60% can be considered as good performance in industry. So indeed "The devil is in the details! "
- But what details do we care about?

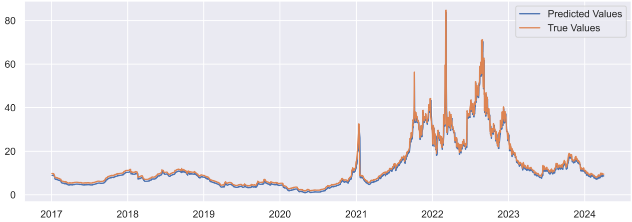


Figure: Predicting Prices = Deception!

Details that we care about

- Feature Importance and Feature Selection.
- Model Archetechture - The art of modeling.
  - Primary models and Meta modeling
  - Sentiment-based modeling using LLM
- The impact of denoising - More Robust Labeling Methods
  - Instead of using daily returns as label, we mimic how traders actually do trading in reality: holding for a period instead of daily MTM.

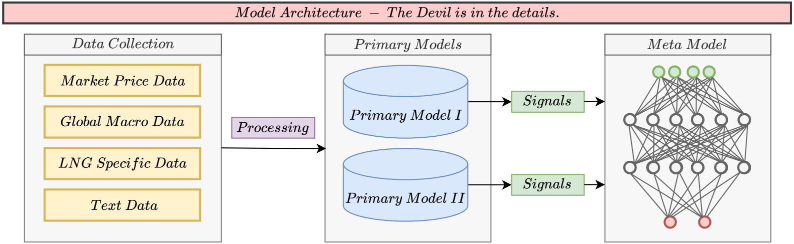


Figure: Outline and Model Architechture

Model Architechture - Primary Models

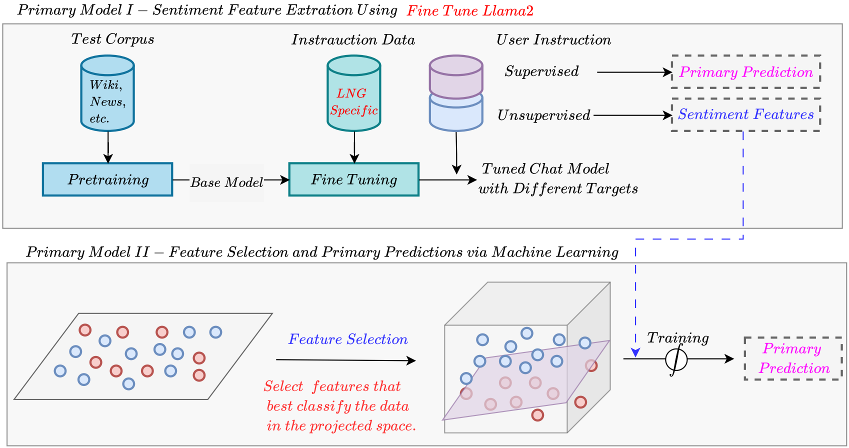


Figure: Primary Models: Information Extraction

Model Architechture - Meta Models

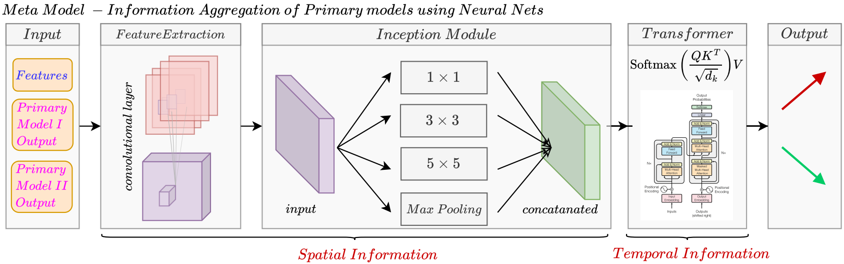


Figure: Meta Modelling using Neural Nets

Some Basic Analysis

- Feature Correlation Heatmap
- Feature Clustering
- Feature Importance

Market News Source

Potential News Source:

- Bloomberg News (Global) – No news archival
- **Wall Street Journal** (US market focus)
- **Financial Times** (EU market focus)
- Reuters – No news older than one year

(536 News from 61 Categories)

Crawler Approach

Crawler Approach:

- **Official API** or HTTP/S Request
- **Selenium** + Clean Request Headers with JS + Add Cookies
- **BeautifulSoup4** JS DOM Model for Static Web

```
options = webdriver.ChromeOptions()
browser = webdriver.Chrome(
    service=ChromeService(ChromeDriverManager().install()))
browser.implicitly_wait(5)
browser.execute_cdp_cmd("Page.addScriptToEvaluateOnNewDocument", {
  "source": """
    Object.defineProperty(navigator, 'webdriver', {
      get: () => undefined
    })
  """
})

title_element = find_with_regex(article, 'span', 'WSJTheme--headlineText--')
summary_element = find_with_regex(article, 'span', 'WSJTheme--summaryText--')
theme_element = find_with_regex(article, 'li', 'WSJTheme--type--')
timestamp_element = find_with_regex(article, 'p', 'WSJTheme--timestamp--')
data.append([timestamp, title, theme, summary])
```

Traditional Ways for Sentiment Analysis

Lexicon-based Methods

- **AFINN**: Integer-based sentiment scores.
- **SentiWordNet**: Assigns sentiment scores to synonyms
- **VADER**

Deep Learning with Word Embeddings

- **Word2Vec**: Words embeddings => Vectors
- **BERT**: Bidirectional Encoder Representations from Transformers

Disadvantages:

1. **Limited Feature Extraction**
2. **Insufficient Contextual Understanding**
3. **Parameter Efficiency**
4. **Generalization**

OpenAI Fine Tuning

Babbage-002 & Davinci-002:

- GPT3 based: smaller, lower latency
- Understand & generate natural language or code
- Completion support

GPT-35-Turbo:

- Most capable & cost effective GPT-3.5 model
- More sophisticated capabilities
- Chat support

OpenAI Fine-tuning is currently available for the following models: **'gpt-3.5-turbo-0125'**, **'gpt-3.5-turbo-1106'**, **'gpt-3.5-turbo-0613'**, **'babbage-002'**, **'davinci-002'**, and **'gpt-4-0613'** (experimental).

1. **Enhanced Generalization**
2. **Deeper Contextual Understanding**
3. **Reduced Manual Feature Extraction**
4. **Parameter Efficiency**

OpenAI Fine Tuning: Unsupervised Learning

Input Parameters	Description
System Command	Declare the purpose and requirements for the training task
Global Information	Historical index prices and LNG fundamentals.
Data List	[Index Price, Volatility, News Headline, News Summary]
Relative Weight	Weights for each training data point.

Output Parameters	Description
Output List	[Direction, Magnitude, Impact Duration, Volatility, Comment]

Specifications:

- Number of Conversation Cases: ~50
- Value Range for Index Price, Annual Volatility, Weekly Volatility: [-10, 10]

OpenAI Fine Tuning: Supervised Learning

Input Parameters	Description
System Command	Declare the purpose and requirements for the training task
Global Information	Historical index prices and LNG fundamentals, essential for trend analysis and prediction.
Data List	[Index Price, Volatility, News Headline, News Summary]
Relative Weight	Weights assigned to each training data case.

Output Parameters	Description
Output List	[Return_T+n, Volatility for period n, Comment]

Specifications:

- Number of Cases: 300-500
- Real values for returns and volatility

Sentiment Analysis with Local LLM (LLaMA2)

Disadvantage of OpenAI Fine Tuning

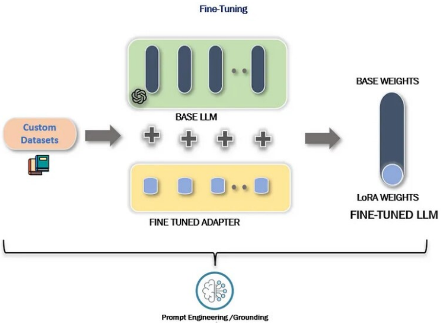
- Poor long term memory
- Expensive for training with whole historical data and background information

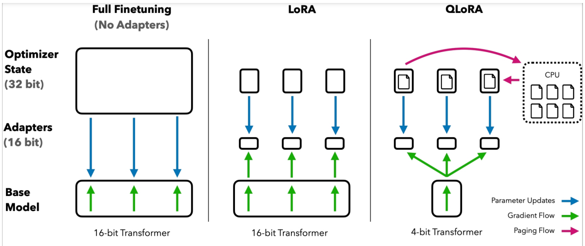
LLaMA 2 Deployment & Fine Tuning Process

- Preparing environments for accelerate, peft, bitsandbytes, transformers, and trl
- Model Configuration & Loading Dataset (for base model)
- 4-bit Quantization Configuration (trainable Low-Rank Adapter layers)
- Set PEFT Parameters and start training
- Model Fine Tuning

```
# Model Configuration
base_model = "NousResearch/Llama-2-7b-chat-hf"
guanaco_dataset = "mlabonne/guanaco-llama2-1k"
new_model = "llama-2-7b-chat-guanaco"
# Loading Dataset
dataset = load_dataset(guanaco_dataset, split="train")
# 4-bit Quantization Configuration
compute_dtype = getattr(torch, "float16")
quant_config = BitsAndBytesConfig(load_in_4bit=True, bnb_4bit_quant_type="nf4", bnb_4bit_compute_dtype=compute_dtype, bnb_4bit_use_double_quant=False)
# Loading Tokenizer
tokenizer = AutoTokenizer.from_pretrained(base_model, trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
# Set PEFT Parameters
peft_params = LoraConfig(lora_alpha=16, lora_dropout=0.1, r=64, bias="none", task_type="CAUSAL_LM")
# Model Fine Tuning
trainer = SFTTrainer(model=model, train_dataset=dataset, peft_config=peft_params, dataset_text_field="text", max_seq_length=None, tokenizer=tokenizer,
```

Sentiment Analysis with Local LLaMA2: LoRA





Thank You for your attention!