

ARISTOTLE UNIVERSITY OF THESSALONIKI

MASTER'S THESIS

Financial Machine Learning, Cryptocurrencies, and Crowdsourced Investing

A deep dive into the Numerai competition

Author:
Panagiotis
PAPAEMMANOUIL

Supervisor:
Dr. Grigorios
TSOUMAKAS

A thesis submitted in fulfillment of the requirements for the degree of Master of Data and Web Science in the

Department of Computer Science

September, 2022

Acknowledgements

First and foremost I want to thank God.

Thank you, Lord, for the blessings you have bestowed on my life. You have provided me with more than I could ever have imagined. You have surrounded me with people who always look out for me. You have given me family and friends who bless me every day with kind words and actions.

Moreover, I want to thank my significant other for all the support during this difficult process.

Also, I would like to thank the administration and the faculty members of the Master's program, for designing an engaging and fascinating, yet challenging curriculum.

Last but not least I wanna thank me.

I wanna thank me for believing in me.

I wanna thank me for doing all this hard work.

I wanna thank me for having no days off.

I wanna thank me for never quitting.

I wanna thank me for always being a giver and trying to give more than I receive.

I wanna thank me for trying to do more right than wrong.

I wanna thank me for just being me all the time.[1]

[1] Snoop Dogg - https://www.youtube.com/watch?v=wGRF3GQ4Wdk

Abstract

Technological breakthroughs are changing our lives. Recent advances in Artificial Intelligence and Cryptography enabled the development of a new generation of applications, which are set to revolutionize how we interact with technology and the world.

This thesis offers a comprehensive study of the **Numerai**[2]. Numerai is an AI-run, crowdsourced hedge fund founded in San Francisco, California, by South African technologist Richard Craib, in October 2015. Numerai is the first hedge fund in the world that combines 3 technological concepts (Artificial Intelligence, Blockchain, and Network Effect in the form of crowdsourcing), aiming to disrupt the financial industry.

The core idea of the **Numerai hedge fund** is to publish all of its data for free and let anyone around the world train AI models and submit predictions. Numerai then combines all models to form the Meta-Model, which controls the capital of the Numerai hedge fund across the global stock market. Finally, participants can get paid based on the contribution of their models to the Meta-Model's performance. This is called the *"Numerai Tournament"*, and it's described as the "hardest data science tournament on the planet" and not without reason, as stock market prediction is one of the world's most challenging and lucrative problems.

This thesis analyzes the concepts around the Numerai as a pioneering hedge fund and a Data Science competition.

In the first chapters, the operational model regarding the **Numerai Hedge Fund** is discussed, along with the Numerai-issued cryptocurrency (NMR) and the Numerai DAO.

Regarding the **Numerai Tournament Data Science competition**, a dataset analysis is conducted, as well as a presentation of the evaluation metrics. Afterward, the staking mechanism that enables the Numerai tournament rewarding strategy is defined. A significant part of the thesis studies the Machine Learning modeling methods applicable to the Numerai Tournament, providing insights into this unique competition.

[2] numer.ai

Finally, the results of the AI models developed during this work are investigated. Over 20 weeks, many models consistently contributed positively to the Numerai Meta-Model, yielding a significant Return On investment.

Contents

Acknowledgements iii

Abstract v

Περίληψη vii

Contents ix

List of Figures xiii

List of Tables xv

1 Introduction 1
- 1.1 Thesis Scope . . . 1
- 1.2 Thesis Structure . . . 3

2 Background in Blockchain & Key Terminology 7
- 2.1 Blockchain . . . 7
- 2.2 Ethereum platform . . . 8
- 2.3 Tokens & Tokenomics . . . 9
  - 2.3.1 ERC20 Tokens . . . 10
  - 2.3.2 Tokenomics . . . 10
- 2.4 Cryptocurrency wallets . . . 12
- 2.5 Decentralized Autonomous Organization (DAO) . . . 13

3 Background in Applied Machine Learning 15
- 3.1 Applied Machine Learning process . . . 15
- 3.2 Supervised Machine Learning methods . . . 17
  - 3.2.1 Regression, Classification, and Ordinal regression/-classification . . . 17
  - 3.2.2 Ensembling and Stacking . . . 18
- 3.3 Feature Exposure . . . 19
- 3.4 Regularization . . . 19
- 3.5 Evaluation frameworks . . . 20
- 3.6 Feature Selection . . . 24
- 3.7 Model hyperparameters and hyperparameters tuning . . . 25
- 3.8 Applied Machine Learning tools & frameworks . . . 26

4 Introduction to Numerai **29**
4.1 Numerai Tournament and Numerai Signals competitions 29
4.2 The Numerai Hedge Fund 29
4.3 The Numerai DAO 33
4.4 Numeraire (NMR) Tokenomics 34
4.4.1 Numerai Whitepaper 34
4.4.2 NMR as ERC20 Token 35
4.4.3 NMR Tokenomics 36

5 Analysis of the Numerai Dataset **41**
5.1 The Numerai Dataset 41
5.2 Numerai dataset evolution 45
5.2.1 About new dataset releases 46
5.3 Features and Targets 47
5.3.1 Data Obfuscation and Normalization 47
5.3.2 Correlation & Feature Groupings 47
5.3.3 Targets 49

6 Numerai Tournament Evaluation Metrics **53**
6.1 Correlation (Spearman's correlation) 53
6.1.1 Example calculations of Spearman's Correlation . 54
6.2 Meta-Model Contribution (MMC) 56
6.3 True Contribution (TC) 57
6.4 More evaluation Metrics and the Diagnostic Tool 61

7 Numerai Tournament rewards; Staking, Scoring, and Payouts **67**
7.1 The Numerai Tournament timeline 67
7.1.1 Weekly Round 67
7.1.2 Daily scores and Round resolution 68
7.2 Staking and Payouts 73
7.2.1 Staking Motivation 73
7.2.2 Staking mechanics 74
7.2.3 Payouts 75
7.2.4 Leaderboard and Medals 80

8 Competing in Numerai tournament; Machine Learning Modeling **83**
8.1 Machine Learning problem formulation 83
8.2 Models evaluation: Regression vs. Classification 85
8.3 Feature Engineering 86
8.4 Feature exposure and model performance 87
8.4.1 Reducing Feature Exposure with Regularization . 88
8.4.2 Reducing Feature Exposure with Neutralization . 88
8.5 Evaluation framework 89
8.6 Improving metrics 91
8.6.1 Feature Selection 91

8.6.2 Eras selection (hard vs. easy eras) 92
8.6.3 Regularization and Feature neutralization 93
8.6.4 Model Ensembling 93
8.6.5 Hyperparameter tuning 93
8.6.6 Model Retraining 93
8.6.7 More ideas on how to improve performance and reduce risk 94

9 Results and Conclusions **95**
9.1 Production Models evaluation and results 95
9.2 Portfolio optimization: Multipliers and Total stake value 100
9.3 Return On Investment 102
9.4 Conclusions 103

A Technical Considerations and Cloud Computing **105**

B How to purchase NMR **109**
B.1 Example workflow: How to send NMR from Binance to the Numerai wallet 110

C Risk Management and Hedging **115**
C.1 Diversification 115
C.2 Hedging against NMR crypto volatility 116

D Published articles and open source code **119**

E Contact information **121**

List of Figures

2.1 Gnosis Safe multisig wallet 13

3.1 Cross-validation methods 22
3.2 Machine Learning Frameworks 26

4.1 Numerai Hedge Fund performance 30
4.2 Numerai CoE tweet 33
4.3 Etherscan Token Numeraire 36
4.4 NMR (Numeraire) at Coinmarketcap 37

5.1 The Numerai Dataset (V2) 41
5.2 Breakdown of the Numerai Dataset (V4) 42
5.3 Total Observations per Era (Numerai Training Data) . . . 44
5.4 Total Observations per Era (Numerai Validation Data) . . 44
5.5 Feature Distribution 48
5.6 Numerai Correlation table (era=1) 49
5.7 Correlation of Feature with the Target 50
5.8 Correlation of Feature with the Target (Rolling Average (10 eras) 50
5.9 Target Distribution (target_nomi_20) 52
5.10 Numerai Targets 52

6.1 True Contribution calculation framework 60
6.2 Diagnostics Tool 62

7.1 Numerai Tournament Round 67
7.2 Numerai Tournament Overlapping scores 68
7.3 Numerai Daily Performance Scores 70
7.4 Model's Round Performance 71
7.5 Email notification with Round summary 71
7.6 Numerai Payouts App 72
7.7 Web interface to manage stake 75
7.8 Staking options 77
7.9 Compund staking 78
7.10 Payout factor 78
7.11 Stake Value growth 79
7.12 Numerai Tournament Leaderboard 80
7.13 Round summary - Medals 82

7.14 Model medals [PAPAEMMAN1] 82

8.1 The Numerai Dataset (V2) 84
8.2 ML models predictions distributions 86
8.3 Data Science field 86
8.4 Numerai Dataset (screenshot) 87
8.5 Random Forest feature importances 92

9.1 Models' correlation score (cumulative) 96
9.2 Models' total contribution score (cumulative) 96
9.3 Models' performance [PAPAEMMAN9] 98
9.4 Random model performance 100
9.5 Production Models 101
9.6 Model performance [PAPAEMMAN17] 102
9.7 Example round summary 103
9.8 Screenshot from Numerai Payouts App 104

A.1 Micrososft Azure (Cloud platform) 105
A.2 Micrososft Azure (Resource group and Services) 106

B.1 Numerai Wallet 110
B.2 Whitelist NMR wallet address 111
B.3 Binance to NMR Wallet 111
B.4 Withdraw NMR from Binance, transaction details 112
B.5 Etherscan transaction details 113

C.1 NMR to USD price 117

List of Tables

4.1 Numerai hedge fund terms 31
4.2 Numerai Fund Service Providers 31
4.3 Numerai portfolio exposures 32

5.1 Numerai Tournament Datasets 43
5.2 5-bin target distribution 51
5.3 7-bin target distribution 51

6.1 Spearman Correlation example calculations 55

7.1 Payouts Calculations for positive scores (CORR TC) . . . 79
7.2 Payouts Calculations for negative scores (CORR or TC) and capped payouts 79

8.1 Hold-out split 89
8.2 Correlation for different cross-validation methods for the same model 90
8.3 Correlation for different cross-validation methods for the same model 91
8.4 LightGBM hyperparameters 94

9.1 Production models 97
9.2 Production models medals 99
9.3 Total Contribution multipliers 101

B.1 Transaction ID details in etherscan 114

1 Introduction

1.1 Thesis Scope

Over the last decade, two major scientific-technological breakthroughs have taken place; **Artificial Intelligence** (Machine Learning, Deep Learning, Reinforcement Learning), and **Blockchain**.

Recent advances in the Artificial Intelligence field enabled the development of great performance solutions for various tasks (Pattern Recognition, Natural Language Processing, Forecasting, Computer Vision, Autonomous Vehicles, Computer Games, etc.). On the other hand, Blockchain technology (along with Cryptography) introduced a new paradigm for application development based on decentralization. The most known applications of this technology are Cryptocurrencies (Bitcoin, Ethereum, Solana, etc.), DeFi (Decentralised Finance), and NFTs (Non-Fungible Tokens).

This thesis aims to study every aspect of the Numerai (*Numerai*) data science competition.

Numerai launched in December 2015 as a single-page website that looked like a Kaggle[1] data science competition, except it was a lot more. Numerai is a hedge fund that combines

1. Artificial Intelligence,
2. Network Effect (crowdsourcing), and
3. Blockchain Technology

aiming to disrupt the Financial industry.

Network effect in Finance and the Numerai Meta-Model

A network effect is a phenomenon where a service becomes more valuable when more people use it. Network effects power all of our most incredible infrastructures, such as e-commerce, social media, delivery services, and many more. Surprisingly, there are no network effects in Finance. The financial sector is a highly competitive field, where massive hedge funds are fighting against each, so essentially, no one has the incentive to work together.

[1] https://www.kaggle.com/

Numerai aims to turn this negative space of competition into valuable collaboration, enabling the network effect in Finance by introducing a number of breakthroughs in the field.

In modern Finance, numerous hedge funds use quantitative methods to design their portfolios. What's different about Numerai is the notion of the **Meta-Model**.

One of the discoveries in the machine learning field is the power of combining many different AI models (ensembling). This method can be successfully applied in the context of stock market prediction. For example, one model can be terrific at predicting utility stocks but terrible at predicting other stocks. In contrast, another model may be competent at predicting the Bank stocks but terrible at the tech stocks. And finally, a third model might have the tech stocks just right. Numerai's core idea is creating an ensemble model from models submitted by the crowd, called "Meta-Model". The meta-model incorporates the working parts of each submodel to achieve superior performance compared to the unique models.

To be able to implement its plan, Numerai had to surpass one of the hardest problems in Finance. Predicting the Stock Market is a challenge with an extremely high barrier to entry because the hedge-fund quality data are deeply proprietary. Liberating financial data is a complicated undertaking that requires bypassing multiple restrictions.

Numerai invented an innovative type of encryption (normalization and anonymization) that enables it to share high-quality financial data with the crowd. By doing so, Numerai could share the data with the public and truly democratize participation in the stock market prediction.

Numerai believes that Wall Street hasn't the monopoly on the brightest data scientists. Therefore, they designed a mechanism to allow everyone worldwide to submit predictions to the fund and get paid for it. Participants don't have to know anything about Finance, hedge funds, and the stock market. You can be a grad student in Tokyo who doesn't even speak English, and suddenly you can share your knowledge with Numerai to make money in the stock market and pay you.

In addition, the data encryption mechanism brought another breakthrough. Having an anonymized dataset forced participants to deal with the challenge as a purely mathematical problem, blocking them from incorporating their biases about particular stocks or particular sectors or industries.

Numerai Master Plan

The idea of crowdsourcing modeling existed before numerai and the idea of using models to predict financial markets also existed before. But the idea of putting them together didn't exist before Numerai. Richard Craig (Founder and CEO of Numerai) is the first to figure out how to harness all of that. He created an incentive mechanism to capture the crowd's intelligence on the internet where anyone could participate and apply that intelligence to the stock market.

Best hedge funds in the world have 200 or 300 data scientists. In Numerai, everyone is a potential node in the network, and hundreds of thousands of people can be involved in the Numerai hedge fund. This number is far greater than any other hedge fund, so you would expect mathematically that it would do the best out of all of them.

Fundamentally, Numerai created the capital structure and the execution structure to democratize access to everyone. Numerai harnesses the intellectual power of the people in the world who don't use this as a day job but people who are good at big data handling, artificial intelligence, and statistics but don't have the resources to start a hedge fund.

Numerai's master plan (*Numerai's Master Plan*) is to become the last hedge fund in the world. And the process of doing it can be summarized in 4 steps.

1. Monopolize intelligence
2. Monopolize data
3. Monopolize money
4. Decentralize the monopoly

Do you want to get more inspired before diving into the rest of the thesis? Then the following 2 videos are what you need.

- Building The Meta Model on Numerai[2]
- Introducing Numeraire[3]

1.2 Thesis Structure

This thesis aims to study the **Numerai**[4] data science competition. Numerai combines **AI** and **Blockchain** technologies striving to disrupt the Stock Market industry. It is described as the *"hardest data science*

[2] https://www.youtube.com/watch?v=dhJnt0N497c
[3] https://vimeo.com/205032211
[4] https://numer.ai

tournament on the planet" and not without a reason because stock market prediction is one of the world's most challenging and lucrative problems.

This document aspires to serve as a comprehensive guide helping readers to understand the business model and the financial mechanisms driving the Numerai Hedge Fund, how the Numerai Tournament works as a data science challenge and the rewarding strategy based on the Numeraire (NMR) cryptocurrency token.

The author has been participating in Numerai Tournament for more than 2 years with excellent results, consistently ranking in the top 1% of the Leaderboard thus, he has an extensive experience with this topic. You can find all of his models under the username **"PAPAEMMAN*"** on the competition's leaderboard page[5].

The thesis is structured as follows.

Chapter 2 defines key terms about **cryptocurrencies and blockchain** that are helpful to understand the full spectrum of Numerai tournamnet operations.

Chapter 3 introduces core concepts of the **Applied Machine Learning** field, focusing on particular aspects of the Financial Machine Learning subfield.

Chapter 4 introduces the **Numerai Hedge Fund, the Numerai DAO, and the Numeraire (NMR) token**. This chapter clarifies the details regarding the company behind the Numerai Tournament and the community that participates in it. Also, it presents the cryptocurrency token Numeraire (NMR), which Numerai issued in 2017 to facilitate the Numerai Tournament's rewards mechanism. Along with Numeraire tokenomics, we do a comparative analysis with the tokenomics of other popular cryptocurrencies such as bitcoin and ethereum.

Chapter 5 focuses on the analysis of the **Numerai dataset**. This chapter describes the evolution of the tournament's dataset through the years and studies the features and the targets used to solve the stock market prediction problem.

Chapter 6 demonstrates the **evaluation metrics** used in the Numerai tournament. We present each metric's mathematical formulation, the calculation framework, and the intuition behind it. Numerai tournament utilizes metrics to assess the performance of the machine learning models as well as the financial characteristics of an investing strategy using the predictions of a model.

[5] https://numer.ai/tournament

Chapter 7 deep dives into the **financial** aspect of the tournament and how the rewards mechanism works. It analyses topics such as; **Staking, Scoring, and Payouts**.

Chapter 8 describes the **machine learning modeling** approaches applicable in the Numerai tournament. We are discussing different methodologies for modeling, feature engineering, cross-validation, and hyperparameter tuning.

This thesis is concluded in ***Chapter 9*** where the evaluation of the production models and the financial results are presented, along with closing remarks and future steps.

Finally, there is an appendix section containing supplementary material, helpful for interesting readers.

Appendix A discusses the **technical considerations** regarding the Numerai tournament. Due to the massive data volume, a personal computer isn't sufficient for this problem, and special hardware is required. This chapter outlines the different technologies and cloud infrastructure used to participate in the tournament successfully.

Appendix B provides a guide on purchasing NMR through a centralized exchange (Binance) and transferring it to a Numerai wallet.

Appendix C discusses the issues related to the volatility of the NMR cryptocurrency and how Numerai tournament participants can hedge against it.

Appendix D and ***E*** provide links to published articles, open source code related to this thesis, as well as the author's contact details.

2 Background in Blockchain & Key Terminology

Blockchain technology and related concepts are a core part of the Numerai ecosystem. This chapter defines key terms about cryptocurrencies and blockchain, focusing on the Ethereum decentralized blockchain platorm.

2.1 Blockchain

A blockchain is a distributed database or ledger that is shared among the nodes of a computer network. As a database, blockchain stores information electronically in digital form. Blockchain is known for its pivotal role in cryptocurrency systems like Bitcoin (Nakamoto, 2008) to keep a secure and decentralized record of transactions. The innovation of blockchain is that it guarantees the authenticity and security of data records and builds trust without the need for a trusted third party.

The main difference between a typical database and a blockchain is the structure of the data. Blockchains collect information in groups called blocks, which contain sets of information. Blocks have a specific storage capacity, and when filled, they are closed and linked to previously filled blocks, creating a chain of data called a blockchain. When a block is filled, it is set in stone and becomes a part of this timeline. This means that previous blocks are immutable and cannot be changed by anyone.

Because of their distributed and anonymized nature, blockchain systems are susceptible to attacks from network nodes. The term **Sybil attack** (Douceur, 2002) was first introduced to denote an episode where the attacker, a Sybil node, tries to forge multiple identities in peer-to-peer distributed systems. The attacker pretends to be many people simultaneously and tries to manipulate the network. From a high-level view, these different identities look like regular users, but behind the scenes, there is a single entity called an "unknown attacker" who controls all these fake entities at once.

2.2 Ethereum platform

Ethereum (Buterin, 2014) is a decentralized blockchain platform that establishes a peer-to-peer network that securely executes and verifies application code, called smart contracts. Smart contracts allow participants to transact with each other without a trusted central authority. Transaction records are immutable, verifiable, and securely distributed across the network, giving participants full ownership and visibility into transaction data. Transactions are sent from and received by user-created Ethereum accounts. A sender must sign transactions and spend Ether, Ethereum's native cryptocurrency, as a cost of processing transactions on the network.

Ethereum, like many community-driven, open-source software projects, has evolved since its initial inception. This progress made possible based on proposal submitted by core developers and community members. EIP stands for *Ethereum Improvement Proposal*. An EIP is a design document providing information to the Ethereum community or describing a new feature for Ethereum or its processes or environment. Every EIP provides a concise technical specification of the feature and a rationale for the feature. The EIP author is responsible for building consensus within the community and documenting dissenting opinions.

At the core of the ethereum platform stand smart contracts. A smart contract is a program that runs on the Ethereum blockchain. It's a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain and they run when predetermined conditions are met. They are typically used to automate an agreement's execution so that all participants can be immediately certain of the outcome without an intermediary's involvement or time loss. They can also automate a workflow, triggering the next action when conditions are met.

Smart contracts work by following simple "if/when…then…" statements that are written into code on a blockchain. A network of computers executes the actions when predetermined conditions have been met and verified. These actions could include releasing funds to the appropriate parties, registering a vehicle, sending notifications, or issuing a ticket. The blockchain is then updated when the transaction is completed. That means the transaction cannot be changed, and only parties who have been granted permission can see the results.

Benefits of smart contracts

- **Speed, efficiency, and accuracy**. Once a condition is met, the contract is executed immediately. Because smart contracts are digital and automated, there's no paperwork to process and no

time spent reconciling errors that often result from manually filling in documents.

- **Trust and transparency**. Because there's no third party involved, and because encrypted records of transactions are shared across participants, there's no need to question whether information has been altered for personal benefit.
- **Security**. Blockchain transaction records are encrypted, which makes them very hard to hack. Moreover, because each record is connected to the previous and subsequent records on a distributed ledger, hackers would have to alter the entire chain to change a single record.
- **Savings**. Smart contracts remove the need for intermediaries to handle transactions and, by extension, their associated time delays and fees.

Solidity[1] is the most popular programming language for smart contract development. It is an object-oriented, high-level language for implementing smart contracts. Solidity is a curly-bracket language designed to target the Ethereum Virtual Machine (EVM). It is influenced by C++, Python, and JavaScript. Solidity is statically typed and supports inheritance, libraries, and complex user-defined types, among other features. With Solidity, you can create contracts for uses such as voting, crowdfunding, blind auctions, and multi-signature wallets.

Another important tool of the ethereum ecosystem is **Etherscan**[2]. Eterscan is the go-to resource for users wishing to discover, verify and approve transactions that have taken place on the Ethereum blockchain. Etherscan was created to facilitate blockchain transparency and accessibility and it is one of the most known Ethereum's "block explorers". It is essentially a search engine that lets users look up, confirm and validate transactions on the Ethereum decentralised smart contracts platform. By entering an address into the search box, you can view the balance, value and all the transactions made through that address.

2.3 Tokens & Tokenomics

A **token** is a digital unit of a cryptocurrency used as a specific asset or to represent a particular use on the blockchain. Tokens have multiple use cases, but **security, utility, and governance tokens** are the most common.

[1] https://docs.soliditylang.org
[2] https://etherscan.io/

2.3.1 ERC20 Tokens

*"**ERC20**"* (*EIP-20: Token Standard*) refers to a scripting standard used within the Ethereum blockchain. This technical standard dictates a number of rules and actions that an Ethereum token or smart contract must follow and the steps to implement it. Readers can think of ERC20 as a set of basic guidelines and functions that any new token created in the Ethereum network must follow. ERC20 tokens follow the specifications defined in EIP-20 (EIP-20: Token Standard)[3].

The EIP-20 introduced a standard interface for tokens. This standard allowed for the implementation of a standard API for tokens within smart contracts. It provided basic functionality to transfer tokens and allowed tokens to be approved so another on-chain third party could spend them. A standard interface allowed any tokens on Ethereum to be re-used by other applications: from wallets to decentralized exchanges.

All ERC-20 tokens are secured by the Ethereum proof-of-stake (PoS) consensus mechanism. This means that validators and block creators are selected based on the amount of tokens they are currently staking. ERC-20 tokens are very popular as they offer higher versatility in terms of use. However, the Ethereum blockchain still provides the same level of protection as blockchains operating with other security protocols. Unlike other popular consensus mechanisms like proof-of-work (PoW), PoS does not rely on high consumption of electrical or computing power to create new tokens. This makes the consensus much more scalable and easily incorporated into different use cases.

2.3.2 Tokenomics

"Tokenomics" has become a popular term in the last few years to describe the mathematics and incentives governing crypto assets. It includes everything about the mechanics of how the asset works, as well as the psychological or behavioral forces that could affect its value long term. In other words, Tokenomics is the topic of understanding the financial characteristics of a token, such as supply and demand and all the other parts of the token.

*"**Tokenomics help researchers determine a cryptocurrency's value.**"*

As in traditional economics, the two forces we are most interested in are **Supply and Demand**. Understanding how those are baked into tokenomics gives us a good sense of how valuable a given token or cryptocurrency should be.

[3] https://eips.ethereum.org/EIPS/eip-20

Cryptocurrencies and tokens built on blockchain have pre-set, algorithmically created issuance schedules. This means that we can accurately predict how many coins will have been created by a specific date in time. Though it is possible for most crypto-assets to have this issuance schedule altered, it will generally require the agreement of many people and is very difficult to implement. This provides some comfort and security for owners, because they know the tokenomics and to what degree their asset will be created in a way that is much more predictable than governments creating fiat money.

Another critical factor affecting tokenomics is how the digital currency will be used. For example, utility tokens enable users access to products or services provided by a platform. If there is a clear link between the usage of a platform or service and the asset, this is usually a good indicator of the value of the coin.

Other important questions to answer include the following:

- How many coins or tokens currently exist?
- How many will exist in the future and when will they be created?
- Who owns the coins? Are there some set aside to be released in the future to developers?
- Is there any information to suggest that a large number of coins have been lost, burned, deleted, or are somehow unusable?

Common metrics Analysing tokens

1. **Market Capitalization**

   The total market value of a cryptocurrency's circulating supply. It is analogous to the free-float capitalization in the stock market.

   $$MarketCap = CurrentPrice * CirculatingSupply \tag{2.1}$$

2. **Fully Diluted Market Cap**

   The market cap if the max supply was in circulation.

   $$FullyDilutedMarketCap(FDMC) = price * maxsupply \tag{2.2}$$

   If max supply is null, *FDMC = price * total supply.* If max supply and total supply are infinite or not available, fully dilluted market cap can't be calculated.

3. **Volume**

   A measure of how much of a cryptocurrency was traded in the last 24 hours.

4. **Circulating supply**

   The number of coins that are circulating in the market and are in public hands. It is analogous to the flowing shares in the stock market.

5. **Total Value Locked (TVL)**

   The current value of cryptocurrencies staked in a crypto liquidity pool. The DeFi total value locked (TVL) is the total of staked assets in all the decentralized finance (DeFi) blockchains.

6. **Volume / Market Cap**

   The ratio of Volume to Market Cap.

7. **Max Supply**

   The maximum amount of coins that will ever exist in the lifetime of the cryptocurrency. It is analogous to the fully diluted shares in the stock market.

8. **Total supply**

   The number of coins that have been already created, minus any coins that have been burned. It is analogous to the outstanding shares in stock market data.

9. **Market Cap / TVL Ratio**

   The ratio of Market Cap to TVL.

2.4 Cryptocurrency wallets

A **crypto wallet** is an essential tool for buying, trading, and selling cryptocurrencies. There are 2 main types of wallets, hardware or software, also called hot and cold storage.

Multi-signature often called "multisig," is a form of technology that provides additional security for wallets and cryptocurrency transactions. Essentially, it ensures that spending cryptocurrency requires more than one approval or sign-off for the transaction to proceed. Making multiple people have to sign a transaction prevents any one person/hack from draining a wallet and making a decision that doesn't go with the majority.

Gnosis Safe[4] is a smart contract wallet running on Ethereum that requires a minimum number of people to approve a transaction before it can occur (M-of-N).

[4] https://gnosis-safe.io/

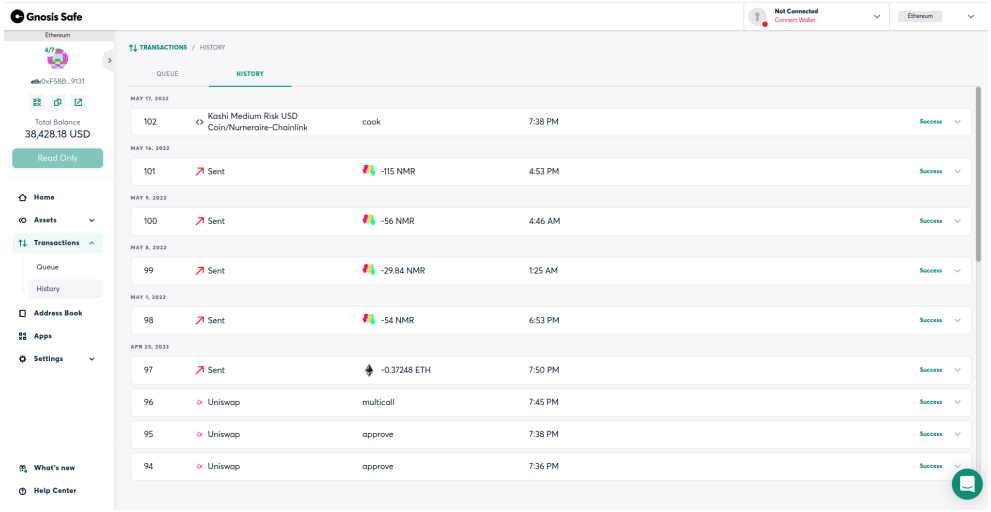


FIGURE 2.1: Gnosis Safe multisig wallet

2.5 Decentralized Autonomous Organization (DAO)

A (DAO) is a relatively new form of legal structure. In contrast with traditional companies, a DAO has no central governing body. Every member (token-holder) can and should participate in the management of the DAO; thus, DAOs follow a bottoms-up approach. DAOs are formed by members who typically share a common goal and attempt to act in the entity's best interest.

At a DAO's core is a voting mechanism that facilitates the management and decision-making process. In the context of crypto, this voting mechanism is implemented using blockchain technology. Accordingly, DAO is a group of individuals, who the community has selected, to hold keys in a multisig wallet. DAO is responsible for managing the funds in the multisig wallet, enacting any passed votes with a binding action, like funding a project, allocating resources, implementing proposals, or anything else the community might suggest and vote. In other words, a DAO is a wallet controlled by a **system of proposals and stakeholder votes**.

What's the difference between a DAO and a multisig wallet?

In many scenarios, a simple multisig wallet is the proper tool to confirm the consensus of multiple parts. For example, having a multisig wallet is likely the best option if 5 people are trying to create a shared pool of money to buy digital assets (for example, NFTs). In this setting, signing the transaction serves as voting. So, if 3 out of 5 people sign, the transaction will go through. Hence, the multisig wallet provides a layer of decentralization and organization.

But how do you scale that coordination amongst thousands of people, all with different percentages of stakes in the treasury? Or, how do you expand the previous concept to a group of total strangers to be able to collaborate trustlessly?

Here is where the DAO concept emerges. Fundamentally, both multi-sig wallets and DAOs are attempts to effectively decentralize control of an organization and coordinate human decision-making around funds quickly and fairly. But, DAOs are used to tackle the concerns mentioned above.

The **4 main components of a DAO are**:

- A community multisig wallet
- A governance system that likely includes a native governance token,
- A system of proposals,
- An execution layer, such as a smart contract or trusted group that enacts the outcome of DAO votes.

3 Background in Applied Machine Learning

3.1 Applied Machine Learning process

Artificial Intelligence (specifically Machine Learning and Deep Learning modes) lies at the core of the Numerai Tournament. Ultimately, AI models built by community members fuel the Numerai Hedge Fund and control its performance.

Machine learning (ML) is a type of Artificial Intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed. Essentially, machine learning algorithms use historical data as input to predict new output values.

From a high-level point of view, tackling a Data Science problem using ML/DL can be summarized in the following steps:

1. **Problem Definition**

   Understand and clearly describe the problem being solved.

2. **Analyze Data**

   Understand the information available that will be used to develop a model.

3. **Prepare Data**

   Discover and expose the structure of the dataset.

4. **Evaluate Algorithms**

   Develop a robust evaluation framework and a baseline model that will be used to improve and spot-check algorithms.

5. **Improve Results**

   Leverage results to develop more accurate models.

6. **Deploy Model**

   Deploy the model to production leveraging Software Engineering and MLOps best practices

7. **Present Results**

   Describe the problem and solution so third parties can understand it.

Analyzing in-depth the previous steps, we can formulate a detailed step-by-step process that can be used for any applied machine learning project. With minor modifications, we followed the method mentioned here to compete in the Numerai Tournament.

1. **Exploratory Data Analysis**

   Investigate the data carefully, perform descriptive analytics, and create visualizations.

2. **Examine the targets**

   Investigate the different values of the targets and how they are distributed. Based on this, determine the type of Machine Learning problem (Regression, Classification).

3. **Evaluation metrics**

   Choose the right metric(s) to evaluate models tailored to the specific problem goals.

4. **Design the evaluation schema**

   Split the data into training, validation, and test folds. Use them for cross-validation or hold-out-based validation.

5. **Develop a Baseline ML Model**

   Build a basic model to use as a baseline..

6. **Iterate to improve the baseline model.**

   Improve on the baseline by adding new features. To add new features, return to the EDA step and identify more insights that can be exploited.

7. **Try different models**

   After reaching a limit with feature engineering, evaluate different models and algorithms.

8. **Keep track of experiments**

   This iterative process of testing different combinations of models and features is called the experimentation phase. Keep a log of all the scores, features, models, and other related information

9. **Feature Selection**

Try feature selection when you reach a limit with different models.

10. **Conclude with feature engineering and feature selection**

    By following the above steps, you will realize that you have chosen a few best models that work well with your data.

11. **Hyperparameter Optimization**

    Conduct hyperparameter optimization to improve the best model(s) performance.

12. **Deploy Model**

    Use containers (docker) and other software engineering and MLOps best practices to make your results reproducible. Deploy the pipeline to Cloud, and automate the process.

3.2 Supervised Machine Learning methods

Supervised machine learning is suitable for problems with specific target values in the given dataset. In Supervised machine learning, the model learns a function that maps an input, called feature vector, to an output called target, based on example input-output pairs.

3.2.1 Regression, Classification, and Ordinal regression/-classification

Depending on the type of target values, there are multiple methods of Supervised Machine Learning.

Regression is a technique for investigating the relationship between independent variables or features and a dependent variable or outcome. It's used as a method for predictive modeling in machine learning, in which an algorithm is used to predict ***continuous outcomes***. In this specific setting, the 5 distinct values of the target can be considered as a sample of a continuous variable spanning from 0 to 1.

Classification is the process of categorizing a given set of data into ***distinct classes***. The classes are often referred to as targets, labels, or categories. Multiclass or multinomial classification is the problem of classifying instances into one of three or more classes while classifying instances into one of two classes is called binary classification. In this specific setting, the 5 distinct values of the target represent 5 different classes.

Ordinal Regression / Ordinal Classification is a type of regression analysis used for predicting an ***ordinal variable***, i.e., a variable whose

value exists on an arbitrary scale where only the relative ordering between different values is significant. It can be considered an intermediate problem between regression and classification.

Regarding the **multiclass classification setting**, there are 3 main types of algorithms;

1. **OneVsRestClassifier**

   Also known as one-vs-all, this strategy consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency (only *n_classes* classifiers are needed), one advantage of this approach is its interpretability. Since each class is represented by one and one classifier only, it is possible to gain knowledge about the class by inspecting its corresponding classifier.

2. **OneVSOneClassifier**

   This strategy consists in fitting one classifier per class pair. At prediction time, the class which received the most votes is selected. Since it requires to fit $\frac{n_classes(n_classes-1)}{2}$ classifiers, this method is usually slower than one-vs-the-rest, due to its $\mathcal{O}(n_classes^2)$ complexity. However, this method may be advantageous for algorithms such as kernel algorithms which don't scale well with *n_samples*. This is because each individual learning problem only involves a small subset of the data, whereas, with one-vs-the-rest, the complete dataset is used *n_classes* times.

3. **OutputCodeClassifier**

   Output-code based strategies consist in representing each class with a ***binary code*** (an array of 0s and 1s). At fitting time, one binary classifier per bit in the code book is fitted. At prediction time, the classifiers are used to project new points in the class space and the class closest to the points is chosen. The main advantage of these strategies is that the number of classifiers used can be controlled by the user, either for compressing the model ($0 < code_size < 1$) or for making the model more robust to errors ($code_size > 1$).

3.2.2 Ensembling and Stacking

Ensemble learning is a general meta-method of machine learning that strives for better predictive performance by combining predictions from multiple models.

There are 3 main classes of ensemble learning methods; Bagging, Stacking, and Boosting.

1. **Bagging** involves fitting many decision trees on different samples of the same dataset and averaging the predictions.
2. **Stacking** involves fitting many different model types on the same data and using another model to learn how to best combine the predictions.
3. **Boosting** involves adding ensemble members sequentially that correct the predictions made by prior models and output a weighted average of the predictions.

3.3 Feature Exposure

Feature Exposure is a core concept in Financial Machine Learning. This section explains the intuition behind feature exposure and why it is essential to identify and address it.

From a very high-level perspective, any supervised ML model is a function that takes an input feature vector (X) and outputs a prediction (y). The model learns a mapping between input features and the predictions at training time. However, in some problems/datasets the underlying process is non-stationary. i.e., features with great predictive power at a time frame might not have any predictive power or might even hurt the model's performance in another time frame. So, a model that attributes too much importance to a small set of features might do well in the short run but is unlikely to perform well in the long run.

Feature exposure is the correlation of predictions with each particular feature. Feature exposure measures how well balanced a model's exposure is to the features. Models with lower feature exposures tend to have more consistent performance over the long run.

3.4 Regularization

Regularization is the technique of constraining a model's complexity to make it simpler and reduce the risk of overfitting.

Overfitting is a concept in machine learning that occurs when a model fits the training data exactly. An overfitted model has learned the details and noise in the training data to the extent that this has adversely affected its performance on new unseen data. Such models have minimal training error, but are irrelevant to any other data set.

There are 2 different types of regularization, the ***L1 and L2 regularization***.

L1 regularization (Tibshirani, 1996), also known as L1 norm or Lasso, combats overfitting by shrinking the parameters towards 0. This makes some features obsolete. This method adds the absolute value of the magnitude of the coefficient as a penalty term to the loss function. Mathematically, it is expressed by extending the loss function as follows:

$$L1LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2 + \lambda \sum_{i=1}^{N} |\theta_i|$$

Essentially, when we use L1 regularization, we penalize the absolute value of the weights.

L2 regularization (Hastie, 2020), or the L2 norm, or Ridge, combats overfitting by forcing weights to be small, but not making them exactly 0. This method adds the squared value of the magnitude of the coefficient as a penalty term to the loss function.

Mathematically, it is expressed by extending the loss function as follows:

$$L2LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2 + \lambda \sum_{i=1}^{N} |\theta_i^2|$$

The regularization term that we add to the loss function when performing L2 regularization is the sum of squares of all feature weights. So, L2 regularization returns a non-sparse solution since the weights will be non-zero (although some may be close to 0). An important disadvantage of L2 regularization is that it's not robust to outliers. The squared terms will blow up the differences in the error of the outliers and the regularization would then attempt to fix this by penalizing the weights.

3.5 Evaluation frameworks

Hold-out is a process used to estimate the machine learning model's performance on a new dataset. It's based on splitting the data into training, validation, and test subsets.

1. **Training dataset**

   The sample of data used to fit the model.

2. **Validation dataset**

   The sample of data used to provide an unbiased evaluation of a model fit on the training dataset. The evaluation becomes more biased as the performance on the validation dataset is incorporated into the model configuration.

3. **Test dataset**

The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset. It is essential to highlight that this dataset must be used only once for the final assessment of the model. Integrating information based on the results of this dataset will lead to overfitting.

Cross-validation (Refaeilzadeh, Tang, and Liu, 2009) is another process used to estimate the out-of-sample machine learning model's performance. It is a resampling method that consists of multiple iterations of training and testing in different subsets of a dataset in order to approximate the generalization error of a model in an unknown dataset.

It works as follows;

1. A dataset is split into k subsets called folds.
2. A ML model is trained using k-1 folds and evaluated in the left-out fold.
3. This process is repeated k times until each fold is used as a testing set once.
4. The k evaluation scores computed in each iteration are averaged to calculate the final evaluation score.

Scikit-learn offers 8 different types of cross-validation[1], as shown in Fig. 3.1. Each method uses a different algorithm to extract the training and testing sets from the full dataset, for each iteration.

[1] https://scikit-learn.org/stable/modules/cross$_v$*alidation.html*

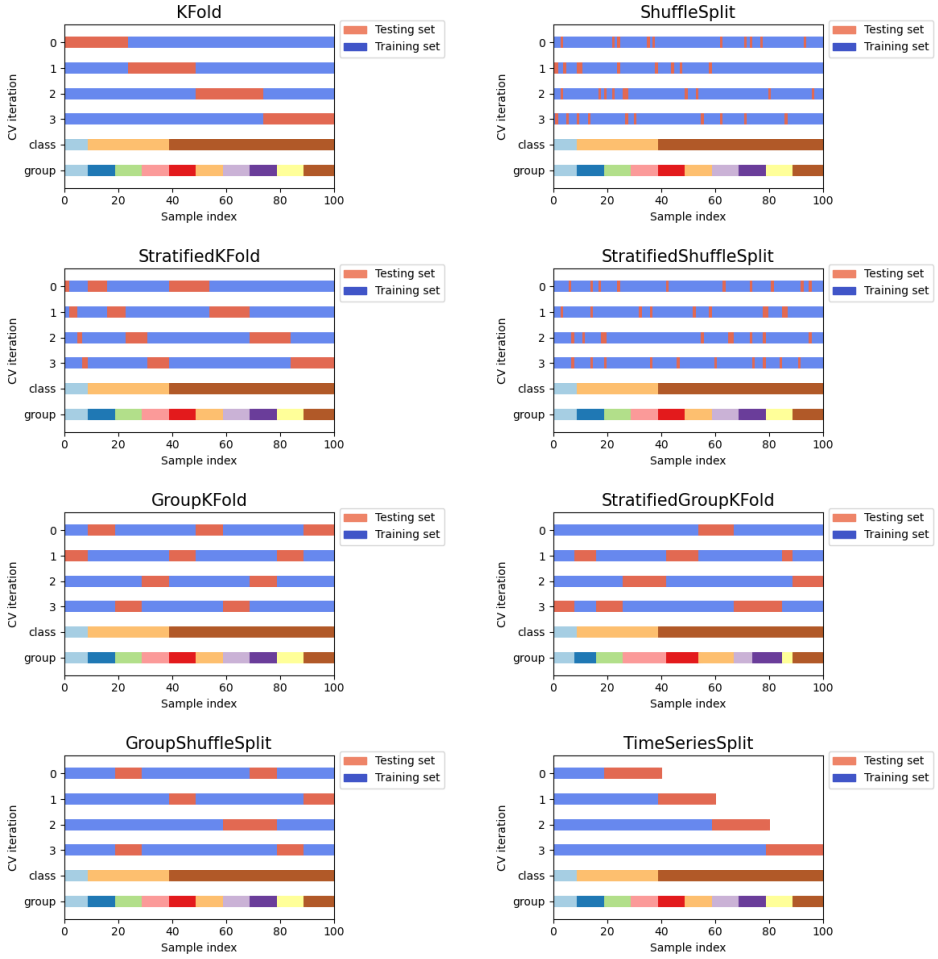


FIGURE 3.1: Cross-validation methods

1. **KFold**

   KFold cross-validation splits the dataset into k consecutive folds without shuffling by default.

2. **ShuffleSplit**

   ShuffleSplit cross-validation randomly samples the entire dataset during each iteration to generate a training set, and a test set.

3. **StratifiedKFold**

   StratifiedKFold cross-validation method splits the dataset into train and test sets in a way that preserves the same proportions of examples in each class as observed in the original dataset. The training and test sets are created without shuffling the original data set, which means that the original row order is preserved.

4. **StratifiedShuffleSplit**

StratifiedShuffleSplit cross-validation method is similar to *StratifiedKFold* because it splits the dataset into train and test sets in a way that preserves the same proportions of examples in each class as observed in the original dataset. However, it will randomly sample the entire dataset during each iteration to generate the training and test sets.

5. **GroupKFold**

   GroupKFold cross-validation is a variant of k-fold, which takes into account a third-party-provided group variable. This method ensures that the same group is not represented in both the test and training sets.

6. **GroupShuffleSplit**

   GroupShuffleSplit cross-validation provides randomized train and test sets according to a third-party-provided group variable. As with GroupKFold, this method ensures that the same group is not represented in both the test and training sets. However, this method does not preserve the row order of the original data set.

7. **StratifiedGroupKFold**

   StratifiedGroupKFold cross-validation is a variant of *StratifiedKFold* that attempts to return stratified folds with non-overlapping groups. The folds are done keeping the percentage of samples for each category. Each group will appear exactly once in the test set in all folds (the number of distinct groups must be at least equal to the number of folds). The difference between *GroupKFold* and *StratifiedGroupKFold* is that the former tries to create balanced folds so that the number of distinct groups is approximately the same in each fold, while *StratifiedGroupKFold* tries to create folds that keep the percentage of samples for each class as as much as possible given the constraint of non-overlapping groups.

8. **TimeSeriesSplit**

   TimeSeriesSplit cross-validation provides train/test sets for splitting samples of time series data observed at fixed time intervals. At each split, the test set must be higher than before, and thus shuffling is inappropriate. This cross-validation method is a variant of KFold. In the k^{th} split, it returns the first k folds as the train set and the $(k+1)$ fold as the test set. Note that, unlike standard cross-validation methods, successive training sets are supersets of previous ones.

Machine learning engineers use cross-validation to compare different machine learning pipelines and identify the one with the minimum

estimated generalization error. Therefore, the selection of the appropriate cross-validation method, is extremely important for the accurate estimation of the out-of-sample error. Using an incorrect cross-validation method can lead to data leakage[2], and thus to overoptimistic estimations.

3.6 Feature Selection

Feature selection refers to the process of eliminating features used to train ML models from the initial pool of available features. The most common benefits of a feature selection process are the simplest final model, shorter training times, and avoidance of the curse of dimensionality. Ultimately, feature selection aims to improve model performance, through the intelligent selection of training features.

There are 3 main categories of feature selection methods:

1. **Filtering methods**

   This method focused on analyzing each feature individually as well as its relationship to the target. Some examples are missing value ratio, Chi-square test, correlation coefficient, low variance filter, mutual information, etc.

2. **Embedded methods**

   This method relies on trained ML models to extract the importance of a feature. This category includes methods such as L1, L2 regularization, and Random Forest feature importance.

3. **Wrapping methods**

   This method relies on a specific ML algorithm to find the optimal set of features. The main idea is to train an ML model with different features each time and compare its performance across iterations. Some examples are forward feature selection, backward feature selection, backward feature elimination, Boruta, etc.

Of course, these methods can be combined, forming different feature selection pipelines. An approach that combines multiple feature selection methods is called ***hybrid***.

[2]Data Leakage is a term referring to scenarios is when information from outside the training dataset is used to create the model

3.7 Model hyperparameters and hyperparameters tuning

Model hyperparameters are parameters that are explicitly defined before the training process and used by the learning algorithm. These parameters define the model architecture and configuration and can significantly affect the model's performance. Therefore, selecting the best hyperparameter values for this dataset is crucial.

Searching for the ideal model architecture is referred to as hyperparameter tuning. Different sets of hyperparameters are tried during this process, and the evaluation scores are calculated based on a particular evaluation framework.

There are 5 main hyperparameter tuning methods;

1. **Exhaustive Grid Search**

   The grid search is an exhaustive search through a set of manually specified set of values of hyperparameters.

2. **Randomized Parameter Optimization**

   Random search searches the grid space randomly; instead of doing an exhaustive search, it tries randomly selected combinations of parameters.

3. **Bayesian Optimization**

   The Bayesian optimization builds a probabilistic model to map hyperparameters to the objective function. It computes the posterior predictive distribution. It iteratively evaluates a promising hyperparameter configuration, updates the priors based on the data to form the posterior distribution of the objective function, and tries to find the location of the optimum. The Bayesian optimization balances exploration and exploitation, so it is suited for functions that are expensive to evaluate.

4. **Gradient-based optimization**

   It is specially used in the case of neural networks. It computes the gradient with respect to hyperparameters and optimizes them using the gradient descent algorithm.

5. **Evolutionary optimization**

   This methodology uses evolutionary algorithms to search the hyperparameter space.

3.8 Applied Machine Learning tools & frameworks

Python[3] is a general-purpose programming language. It's the most popular programming language for data science applications. The vast majority of the machine learning frameworks are developed in python or offer a python API.

For Machine Learning development, we used 3 well-known python libraries; *Scikit-Learn (Pedregosa et al., 2011), Lightgbm (Ke, Meng, Finley, Wang, Chen, Ma, Ye, and Liu, 2017), and Xgboost (Chen and Guestrin, 2016).*



FIGURE 3.2: Machine Learning Frameworks

Scikit-learn[4] is the standard Machine Learning framework in Python. It's designed to offer the following advantages;

- Simple and efficient tools for predictive data analysis
- Accessible to everybody and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

LightGBM[5] is a gradient boosting framework that uses tree-based learning algorithms. It is distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support parallel, distributed, and GPU learning.
- Capable of large-scale data handling.

XGBoost[6] is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements

[3] https://www.python.org/
[4] https://scikit-learn.org/stable/index.html
[5] https://lightgbm.readthedocs.io/en/latest/
[6] https://xgboost.readthedocs.io/en/stable/index.html

machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solves many data science problems quickly and accurately. The same code runs on major distributed environments (such as Hadoop, SGE, and MPI) and can solve problems beyond billions of examples.

4 Introduction to Numerai

4.1 Numerai Tournament and Numerai Signals competitions

As described in Chapter 1 the **Numerai** combines **AI** and **Blockchain** technologies aiming to disrupt the Stock Market industry.

The **Numerai Tournament** is a weekly competition where data scientists can build AI models on abstract financial data to predict the stock market. Their models can be staked with **NMR cryptocurrency**[1] to earn rewards based on performance. The staked models are combined to form the Meta Model, an AI model that controls Numerai's hedge fund investments in the global stock market.

Besides the Numerai Tournament, Numerai provides the **Numerai Signals** challenge. In this competition, data scientists can upload stock market signals and determine how predictive and original they compare to all other signals on Numerai. Similarly, with Numerai Tournament, these signals can be staked with NMR cryptocurrency to earn rewards. The best and most original signals are used in managing the Numerai's hedge fund investment strategy.

This thesis focuses solely on the Numerai tournament. However, many of the topics discussed also apply to the Numerai Signals.

4.2 The Numerai Hedge Fund

Numerai (Numerai Hedge Fund) is a quant hedge fund headquartered in San Francisco, CA, built on thousands of crowdsourced machine learning models.

A quantitative or quant hedge fund (*Quantitative Hedge Fund*) is an investment fund whose securities are chosen based on numerical data compiled through quantitative analysis. These funds are considered

[1] https://coinmarketcap.com/el/currencies/numeraire/

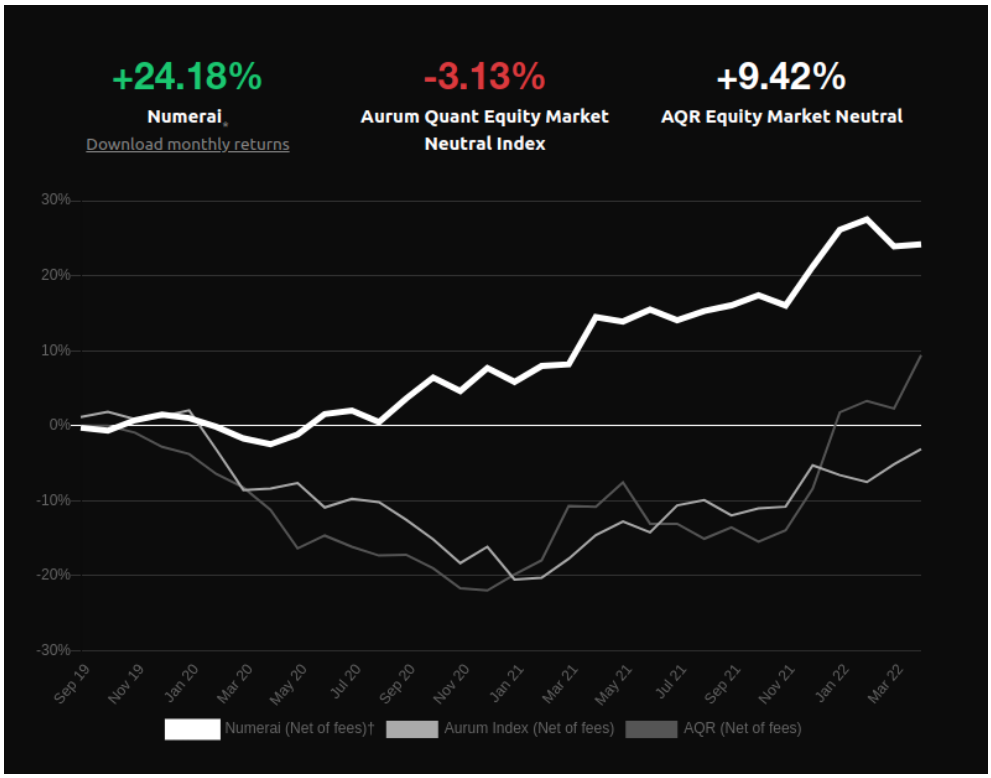


FIGURE 4.1: Numerai Hedge Fund performance

non-traditional and aggressive (non-passive). They are built with customized models using software programs and algorithms to determine investments. Advocates of quant funds believe that choosing investments using inputs and computer programs helps fund companies cut down on human fund managers' administration risks and losses.

Numerai gives away anonymised weekly stock market data so that data scientists worldwide have free, hedge-fund quality data to build their machine learning models - data they would otherwise not have access to. From inception, Numerai has believed that open participation, in the form of readily available data, is a core component in crowdsourcing solutions to significant data science problems. After Numerai receives participants' predictions, it combines them to create a Meta-Model responsible for managing the fund's investment strategy.

As shown in Fig. 4.1, Numerai Hedge Fund presents superior returns than benchmark market indices. Specifically, Numerai outperformed the *Aurum Quant Equity Market Neutral index* and the *AQR Equity Market Neutral* from September 2019 to March 2022, with a +24.18% ROI (Return On Investment) compared to -3.13% and +9.42%, respectively.

As of April 1, 2022, Numerai had roughly $71mm in Assets Under Management (AUM), targeting $100mm in assets in the coming months.

Minimum investment	$1mm
Lockup	None
Fund Capacity	$500mm
Management Fee	1% annualised
Performance Fee	25% with high-water mark (Class A) - 20% with high-water mark (Founders Class)

TABLE 4.1: Numerai hedge fund terms

Fund Administrator	SS&C
Auditor	KPMG
Legal Counsel (US)	Cole-Freiman and Mallon
Legal Counsel (Cayman)	Maples
Prime Broker	UBS
Tax Advisor	KPMG
Compliance Consultant	Titan Regulations

TABLE 4.2: Numerai Fund Service Providers

Table 4.1 presents the terms required for interesting parts in order to invest in the Numerai hedge fund. The minimum investment is $1mm with no lockup period and 1% annualised management fee. Also, 25% performance fees are applied for investors not part of the founder class. In addition, table 4.2 contains the necessary legal information about the Numerai Fund partners and service providers.

The Numerai hedge fund consists of equally-weighted long and short portfolios with approximately 500 individual equities in each portfolio. The investing strategy is characterised as mid-term because the fund holds the stocks for 3-4 months. Table 4.3 outlines the details of the investment strategy.

Gross & Net Exposure	Gross: 500% - 600% (Net: 0%)
Investment Products	100% Equities (Across market capitalizations)
Holding Period	2-4 Months (Weekly trading)
Maximum position size	< 2% AUM, monitored daily
Trading	Monitor for event risk and close or restrict security
Market Neutral	Market, country and sector-neutral
Factor Exposure	Beta of 0 with limited exposure to core style factors
Target Volatility	< 10%
Industries	< All except Biotechnology and REITS

TABLE 4.3: Numerai portfolio exposures

4.3 The Numerai DAO

> "I'm going to make a DAO called the Numerai Council of Elders, which will be a multisig wallet controlled by elite members of the Numerai Community. Numerai will give it initial $$ but have no role whatsoever in where that $$ goes."

With this tweet, in March 2021, Richard Craib[2] announced the creation of Numerai DAO, called **"Council of elders"**. Richard Craib is the founder and CEO of Numerai. Previously, Craib was a data scientist at an asset management firm with $15 billion under management. He has a degree in pure mathematics with a focus on abstract algebra at Cornell University.



FIGURE 4.2: Numerai CoE tweet

As Richard Craib had claimed, the main focus of Numerai DAO is *"advancing the interests of Numerai and the maintenance of international peace and security."*.

For that reason, an initial amount of $100k worth of NMR and ETH was sent to Numerai's DAO, on the day of its formation. The community wallet in Numerai DAO is a **"multisig" crypto wallet** in Gnosis Safe[3]. Numerai DAO requires 4 out of 7 individuals to agree and sign for any passing proposal.

Alongside with the community wallet, the Numerai team, designed a **proposal template** to allow the community members to propose their ideas to community for voting. The proposal has to justify several factors related to the project and is subject to discussion, criticism, and vote by all community members.

[2] https://twitter.com/richardcraib
[3] https://gnosis-safe.io/app/eth:0xF58B7c28DAF13926329ef0c74FA3f7258f5A9131/home

Proposal Template

"[Proposal Title]: Title"
The body of the proposal should include the following sections;

- **Proposal**: A summary of the project and who will be working on it
- **Timeline**: What's the timeline of the project?
- **Best-case outcome** : Describe the best possible scenario.
- **Worst-case outcome** : What could go wrong?
- **Success criteria** : How can the success of this project be objectively measured and evaluated?
- **Funding**: Is funding required? If yes what's the budget.

Fundamentally, every community member can vote and comment on any proposal, but the final decision and approval are granted to Numerai DAO members.

Some examples of Numerai DAO projects that have been approved and funded are

- Development of Tutorials for the community
- Open Source projects contribution
- Investing in specific cryptocurrency assets

4.4 Numeraire (NMR) Tokenomics

4.4.1 Numerai Whitepaper

In February 2017, Richard Craib, Geoffrey Bradway, Xander Dunn, and Joey Krug published the Numerai whitepaper (Craib, Bradway, Dunn, and Krug, 2017), titled ***Numeraire: A Cryptographic Token for Coordinating Machine Intelligence and Preventing Overfitting***.

A white paper, also written as "whitepaper", is an informational document usually issued by a company or not-for-profit organization to promote or highlight the features of a solution, product, or service that it offers or plans to offer. White papers are also used as a method of presenting government policies and legislation and gauging public reaction.

In this paper, the authors introduce the staking and auction mechanics at the core of the Numerai tournament. They also present a new cryptocurrency token called Numeraire (NMR), which will serve as an instrument for data scientists to express their confidence in their models' live (out-of-sample) performance and receive rewards based on the actual performance.

Abstract: "Machine learning competitions are susceptible to intentional overfitting. Numerai proposes Numeraire, a new cryptographic token that can be used in a novel auction mechanism to make overfitting economically irrational. The auction mechanism leads to equilibrium bidding behavior that reveals rational data scientists' confidence in their models' ability to perform well on new data. The auction mechanism also yields natural arguments for the economic value of a Numeraire token."

4.4.2 NMR as ERC20 Token

Numeraire (NMR) is a decentralized, open-sourced ERC-20 (Ethereum Request for Comments 20) utility token, running on the Ethereum blockchain, developed by the Numerai team. Ethereum tokens are represented as smart contracts that are executed on the Ethereum blockchain. The source code to Numeraire's smart contract is publicly available.[4]

Numerai and the Numeraire token are unique in terms of the idea behind their creation. NMR is the first cryptocurrency to be created and released by a hedge fund. One of the main benefits of the NMR token is that it is awarded to data scientists whose models perform well in the Numerai tournament. This means that the token becomes more valuable as more people enter the tournament and start competing.

Fig. 4.3 shows a screenshot from the NMR page in etherscan[5]. This page displays valuable information, such as the token's price, the number of holders, total transfers, details about the transactions, and many more.

[4] https://github.com/erasureprotocol/NMR
[5] https://etherscan.io/token/0x1776e1F26f98b1A5dF9cD347953a26dd3Cb46671

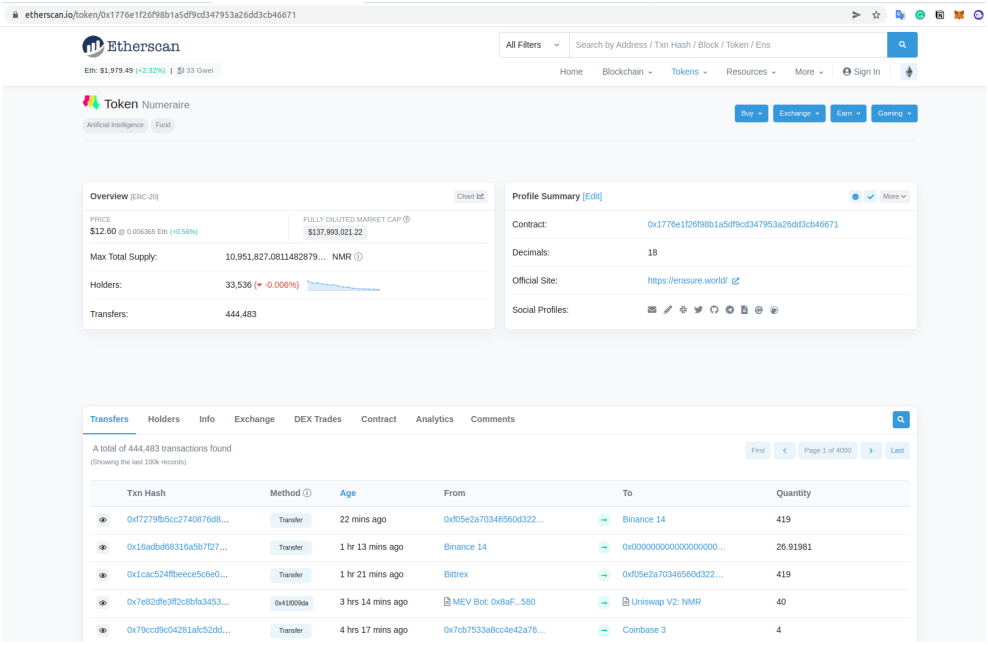


FIGURE 4.3: Etherscan Token Numeraire

4.4.3 NMR Tokenomics

This subsection outlines the common tokenomics metrics for NMR in contrast with 5 other polular coins (BTC, ETH, SOL, XMR, XRP).

As shown in Fig. 4.4, NMR's price heavily fluctuated over the last year. This is due to the general volatility in the crypto space during 2022. Even though NMR is a utility token with a specific use case, it is affected by the general changes and speculations in the broader crypto space.

About Numeraire (NMR) token

- The maximum supply of NMR is 11,000,000 tokens. The supply cap for Numeraire was initially 21 million tokens; however, the platform announced that it is going to cut this maximum by more than half in 2019.
- In a 2018 press release, Numerai announced that three million NMR tokens will remain locked until 2028 and will be released as a reward incentive for tournament participants. The Numeraire token did not undergo an initial coin offering (ICO). Instead, Numerai decided to distribute 1 million NMR tokens among 12,000 platform users with the highest activity statistics.
- NMR tokens are burned every week, as tournament participants who venture losing models and predictions lose their stake in the competition.

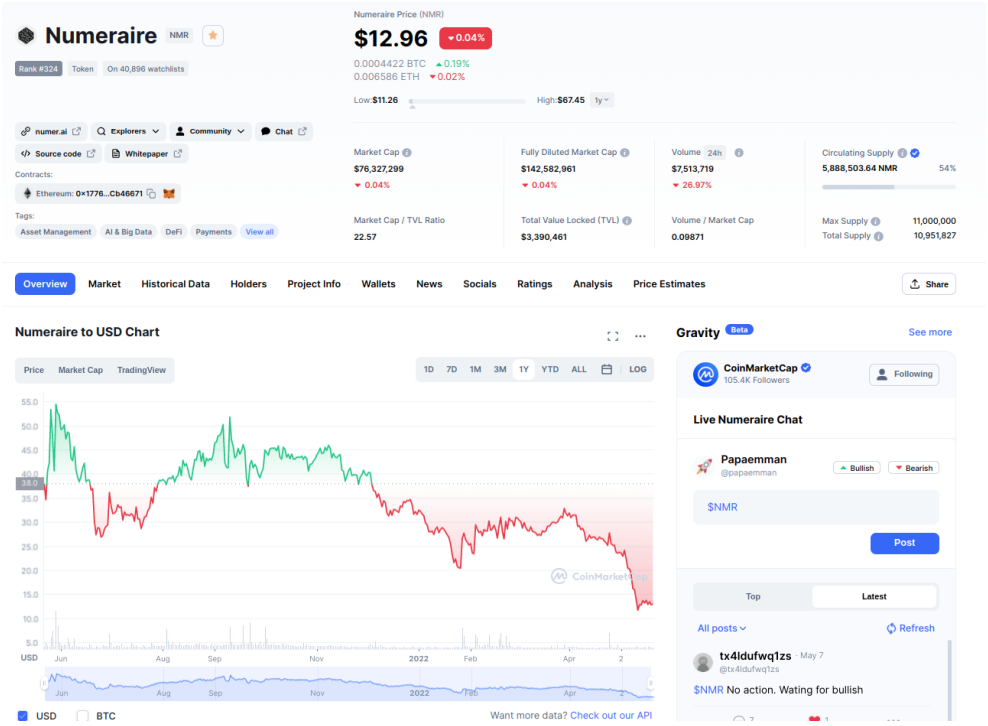


FIGURE 4.4: NMR (Numeraire) at Coinmarketcap

- NMR is growing in popularity as the hedge fund performs better, and more data scientists participate in the weekly tournament. If you are looking to trade NMR, there are several exchanges now offering the token.

Comparative Tokenomics analysis of BTC, ETH, SOL, XMR, XRP, NMR (May, 2022)

		Token		
	Metric	**BTC**	**ETH**	**SOL**
1.	**Price ($)**	29,573.49	1,876.07	45.89
2.	**Market Capitalization ($)**	563,695,264,518	226,877,879,892	15,574,535,164
3.	**Fully Diluted Market Cap ($)**	621,377,537,052	226,877,879,892	23,477,353,857
4.	**Volume ($/24h)**	36,654,181,477	22,546,554,814	2,117,810,220
5.	**Circulating supply**	19,050,075.00 BTC (91%)	120,932,434.75 ETH	339,399,242.64 SOL
6.	**Volume / Market Cap**	0.06482	0.0993	0.1362
7.	**Max Supply**	21,000,000	Inf	Inf
8.	**Total supply**	19,050,075	120,932,435	511,616,946
9.	**Market Cap / TVL Ratio**	-	-	-
10.	**Total Value Locked ($)**	-	-	-

	Metric	Token		
		XMR	**XRP**	**NMR**
1.	**Price ($)**	195.35	0.396	12.38
2.	**Market Capitalization ($)**	3,540,955,854	19,143,072,617	72,913,681
3.	**Fully Diluted Market Cap ($)**	3,546,034,661	39,658,224,411	136,206,164
4.	**Volume ($/24h)**	228,473,246	1,998,990,112	6,791,178
5.	**Circulating supply**	18,126,347.75 XMR	48.34B XRP (48%)	5,888,503.64 NMR (54%)
6.	**Volume / Market Cap**	0.06499	0.1046	0.09269
7.	**Max Supply**	inf	100,000,000,000	11,000,000
8.	**Total supply**	18,126,348	99,989,535,142	10,951,826
9.	**Market Cap / TVL Ratio**	-	-	21.77
10.	**Total Value Locked ($)**	-	-	3,345,434

5 Analysis of the Numerai Dataset

5.1 The Numerai Dataset

As discussed in previous chapters, Numerai is a quantitative hedge fund with an ethereum-based platform allowing developers and data scientists to experiment and create machine learning models. The platform's main goal is to disrupt the Stock Market and bring decentralization to the data science field by allowing individuals to build models in high-quality stock market data, stake NMR cryptocurrency, and be rewarded based on the performance of their models in the Numerai Tournament.

era	feature1	...	feature310	target
era1	0.75	...	0.00	0.25
era1	1.00	...	0.25	0.75
era1	0.25	...	1.00	0.00
era1	0.25	...	0.00	1.00
era1	0.75	...	0.25	0.25
era1	0.00	...	0.75	1.00

FIGURE 5.1: The Numerai Dataset (V2)

At the core of the Numerai Tournament is the free dataset[1]. The Numerai data set consists of standard financial market data for a universe of roughly 5,000 global equities. They are assembled from several market data providers over a 10-20 year timeframe. Each row of the dataset represents a single equity for 1 week (era).

As can be understood, Numerai isn't allowed to freely distribute the original data due to NDAs signed with the stock market data providers. Therefore the data are **obfuscated and normalized** before they are given to the tournament participants. This is a solution to provide participants with a high-quality set of financial data without revealing the original data simultaneously.

It is obvious that the performance of the models in the Numerai Tournament is directly correlated with the quality and quantity of information embedded in the dataset. That's why Numerai data has significantly expanded over several iterations: The original data set contained just 40 individual features and a single target variable. In subsequent releases, the feature count grew to 310 (Fig. 8.1) and then to over 1,000 most recently, along with around 20 different targets.

Participants can download the Numerai Dataset for free, either from the numerai website or by using the Numerai API [2].

The **Numerai Dataset consists of 3 distinct subsets/files**, as shown in Fig. 5.2.

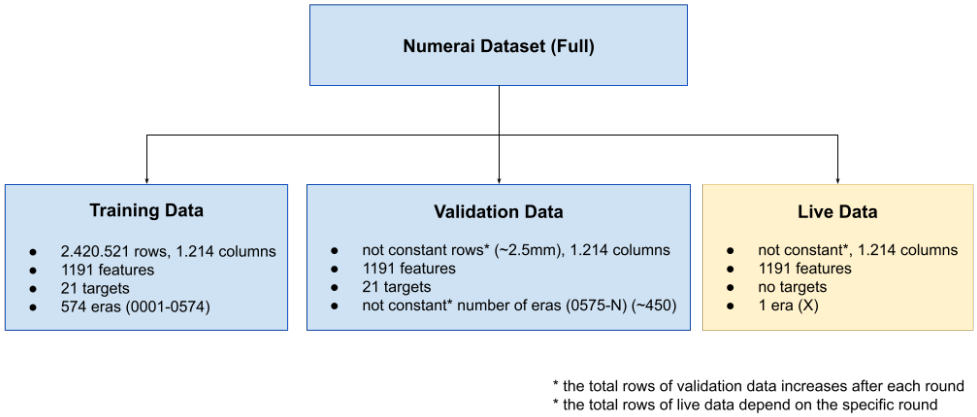


FIGURE 5.2: Breakdown of the Numerai Dataset (V4)

1. **Numerai training dataset**

[1] https://numer.ai/data
[2] https://numerapi.readthedocs.io/en/stable/

The largest dataset is, of course, the Numerai training dataset. This is the main dataset used by participants in order to train the ML/DL models. It consists of ~2.5mm rows and ~1k columns of labeled training data. It represents one continuous period of historical data.

2. **Numerai validation dataset**

   The second dataset that Numerai provides as a separate file is the Numerai validation dataset. This is also a labeled dataset used to generate validation predictions and compute validation metrics. Predictions on these rows are not required for the weekly submission, but they can be submitted at any time to receive diagnostics regarding the performance of the models. This is the most recent data Numerai provides, coming up to the last completed week. This dataset is updated weekly, meaning that the previous week's live data are included in this dataset as soon as the week ends. Evaluating based on this data is particularly useful for seeing how models' performance declines over time and how it would have been performing lately.

3. **Numerai live dataset**

   The final dataset provided by Numerai is the Numerai live dataset. This dataset contains unlabeled feature data used to generate live predictions (updated weekly). Participants must use this file to generate their tournament submission file each week. This file changes every week, so downloading the most recent version is required.

Dataset	Eras	Rows	Features	Targets
numerai_training_data.parquet	574 eras (0001- 0574)	2.420.521	1191	yes
numerai_validation_data.parquet	Increasing each week (0575-N) [~450] eras	Increasing each week (~2.5mm)	1191	yes
numerai_live_data.parquet	1 era (X)	~5k [3]	1191	no

TABLE 5.1: Numerai Tournament Datasets

About eras in the Numerai dataset

Numerai dataset is divided in eras (time intervals). Each era in the dataset corresponds to a single week. The earlier eras are smaller, meaning that they include data for less equities, but generally, each era is 4000-5000 rows (different equities). Fig. 5.3 and Fig. 5.4 show the total number of equities per era for the training and validation dataset.

Even though the time factor appears strongly in the numerai dataset, this doesn't make the dataset suitable for time-series analysis. Each row in the dataset represents a different asset (equity) identified by a

[3] the actual number of rows depending on the specific round

unique id, which changes across eras. This means that for stock XYZ the 'id' will vary from era to era; therefore, there is no indication to link the same asset between different eras.

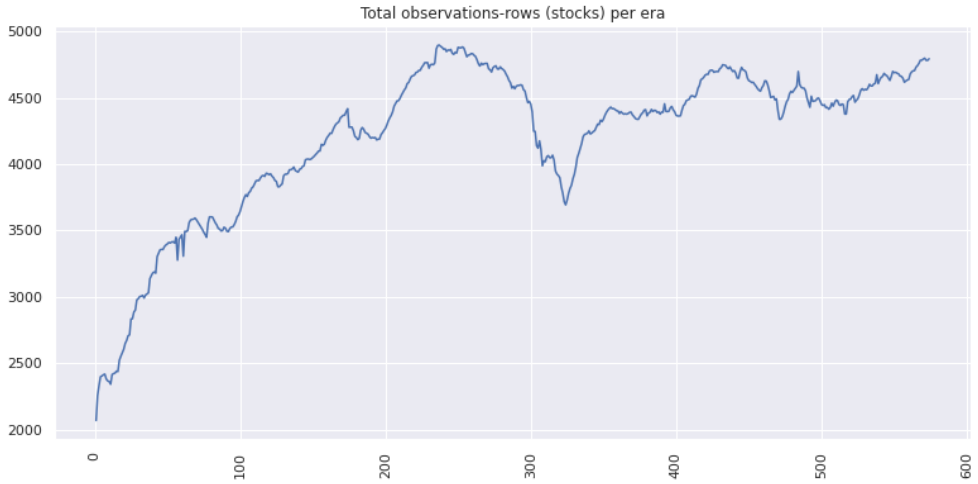


FIGURE 5.3: Total Observations per Era (Numerai Training Data)

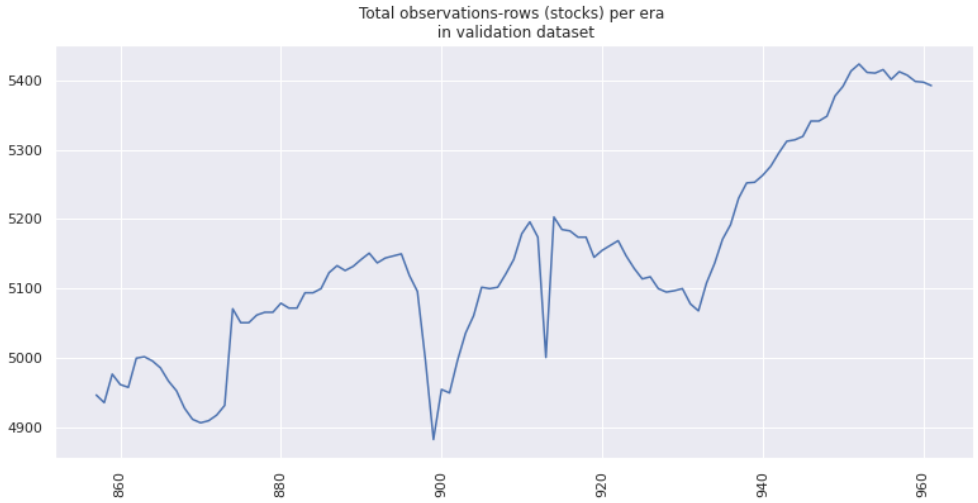


FIGURE 5.4: Total Observations per Era (Numerai Validation Data)

Apart from the 3 primary datasets (Numerai training data, Numerai validation data, and Numerai Live data), Numerai provides 3 auxiliary datasets to assist data scientists in participating in the tournament. These datasets contain metadata and submission file examples.

There are 3 auxiliary datasets:

- ***live_example_preds.parquet***: Example file of numerai live dataset predictions

- ***validation_example_preds.parquet***: Example file of numerai validation dataset predictions
- ***features.json***: Metadata about the features along with correlation metrics

5.2 Numerai dataset evolution

As the Numerai hedge fund evolved through the years, the Numerai dataset has been changed too. More features, targets, and data have been added to the Numerai dataset, helping participants build better AI models and achieve higher rewards.

The current version of the Numerai Tournament Dataset is called *"Numerai Data V4"*, meaning that there were 3 dataset versions before it.

- **Initial dataset** (Released 2017)

  The original data set contained just 40 individual features and a single target variable. This dataset was used for the initial inception and development of the Numerai tournament.

- **V2 Version - Legacy dataset** (Released in late 2019)

  The V2, also called "Legacy Dataset" has 310 features and targets for the training and validation sets, but not the test set. This dataset is still available for download on Numerai's website and contains all legacy data, including training, validation, test, and live. It also includes the legacy example predictions and example models.

- **V3 Version - Supermassive Dataset** (Released in September of 2021)

  The "Supermassive Dataset" has 1050 features and 20 targets for the training and validation sets, but not the test set.

  1. NUMERAI_TRAINING_DATA.PARQUET: Data used to train your model.
  2. NUMERAI_VALIDATION_DATA.PARQUET: Data used to validate or train your model. The set of eras in this data is static.
  3. NUMERAI_LIVE_DATA.PARQUET: The live data used to submit predictions. This changes every week.

- **V4 Version** (Released in April of 2022)

  This is the latest and greatest dataset with 1191 features and 20 targets for all historical data.

1. TRAIN.PARQUET: Data used to train your model
2. VALIDATION.PARQUET: Data used to validate or train your model. This expands every week to include data for the most recent eras with targets available.
3. LIVE.PARQUET: The live data that the AI model predicts. This changes every week.

5.2.1 About new dataset releases

It's clear that Numerai constantly strives to provide better and richer data to participants. Therefore it's safe to assume that V4 will not be the "final" Numerai Dataset. Below we provide the "release notes" regarding the V4 dataset. This list outlines the differences between the V4 and V3 datasets.

1. In the V4, there are 141 new features included, for a total of 1191 features, plus targets available for all eras.
2. Test eras are now considered part of validation. This means there is a train.parquet, validation.parquet, and live.parquet file available.
3. In the V4 dataset, each live era will graduate to validation data, and targets are populated in the graduated eras as soon as possible.
4. The previous features included in the V3 dataset have been slightly improved and thus renamed. There is a map between V3 and V4 features so that users can utilize any previous feature research on new data without starting over if they wish.
5. The optional targets have been slightly improved and so are named with a new pattern, e.g., *target_jerome_v4_20*.
6. Submissions are now only required to have live predictions. Any other predictions submitted will be safely ignored. Numerai will accept submissions from any version as long as the predictions have all of the live indices.
7. Numerai enforces "backward compatibility" in new dataset releases. This means that neither the legacy (v2) nor the current versions (v3) are being changed, so any automated model will continue to work.
8. The example script repo[4] will be updated with v4 versions of each example model. These models will be functionally identical but will be updated to read from the v4 dataset.

[4] https://github.com/numerai/example-scripts

5.3 Features and Targets

The Numerai Tournament datasets cost millions of dollars per year and it contains stock market data related to both fundamental and technical financial analysis.

Fundamental analysis is a method of assessing the intrinsic value of a stock. It combines financial statements, external influences, events, and industry trends. On the other hand, **Technical analysis** evaluate investments by analyzing statistical trends gathered from trading activity, such as price movement and volume. Unlike fundamental analysis, which attempts to evaluate a security's value based on business results such as sales and earnings, technical analysis focuses on the study of price and volume.

Besides a general statement of using both fundamental and technical analysis for the feature generation, Numerai doesn't disclose any specific information about how each feature is calculated.

5.3.1 Data Obfuscation and Normalization

As we mentioned earlier, the Numerai team have implemented an extensive data obfuscation and data normalization process to be allowed to distribute the financial data freely, without any legal hardships. Consequently, the numerai dataset has no value outside the Numerai Tournament competition.

Every feature has a random name, takes 5 unique values [0.00, 0.25, 0.50, 0.75, 1.00] and has the same uniform distribution, as shown in Fig. 5.5. A (funny) hashing function generates feature names in the dataset.

Some examples are:

- feature_honoured_observational_balaamite
- feature_polaroid_vadose_quinze
- feature_untidy_withdrawn_bargeman
- feature_genuine_kyphotic_trehala
- feature_unenthralled_sportful_schoolhouse
- etc...

5.3.2 Correlation & Feature Groupings

In previous releases, features were grouped in explicit feature groups, based on the broader category they belong. After the V3 release, there

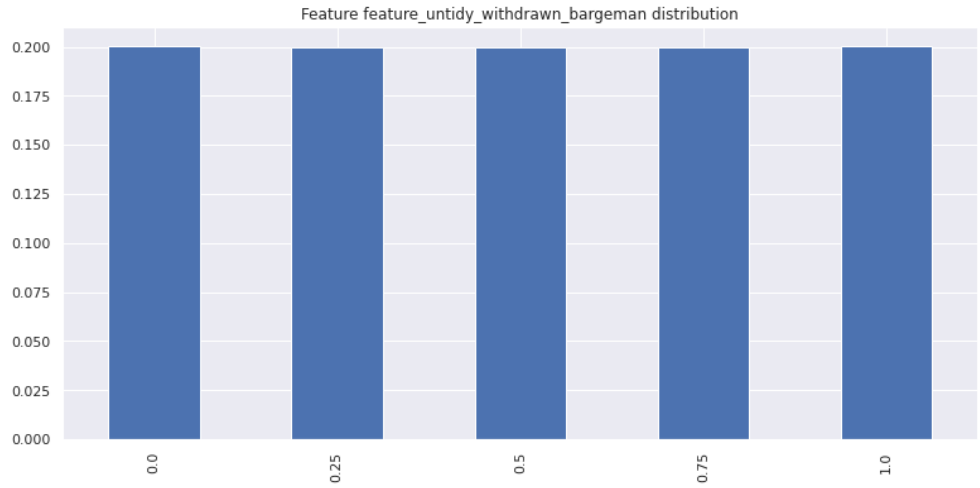


FIGURE 5.5: Feature Distribution

isn't any specific grouping in features, but Data Scientists are free to create their own clustering based on patterns emerging in the dataset.

Some apparent clusters appear when visualizing the correlation matrix (Fig. 5.6) for the full Numerai training dataset (Features and Target). These clusters can be exploited later to lead the model building process.

Another important topic when analyzing the Numerai dataset is the study of the **correlation between the features and the target**. Many features significantly correlate with the target, and some perfectly correlate within a particular era.

Also, features don't have a constant correlation with the target through time. The correlation can change over time, as shown in Fig. 5.7.

Applying a 10-era rolling average filter in correlation of a feature and the target across era, can expose the trends in the dataset (Fig. 5.8). This particular feature (feature_untity_witdrawn_bargeman) negatively correlates with the target in many eras and positively correlated with the target in others.

This means that the underlying relationship between features and target is **drifting** over time. Consequently, this aspect of the dataset requires special treatment in the ML model development phase. This phenomenon is called **Data-drift** in Data Science and occurs when the dataset used to train an AI model does not mimic the data in production. As a result, models start to act strange and underperform.

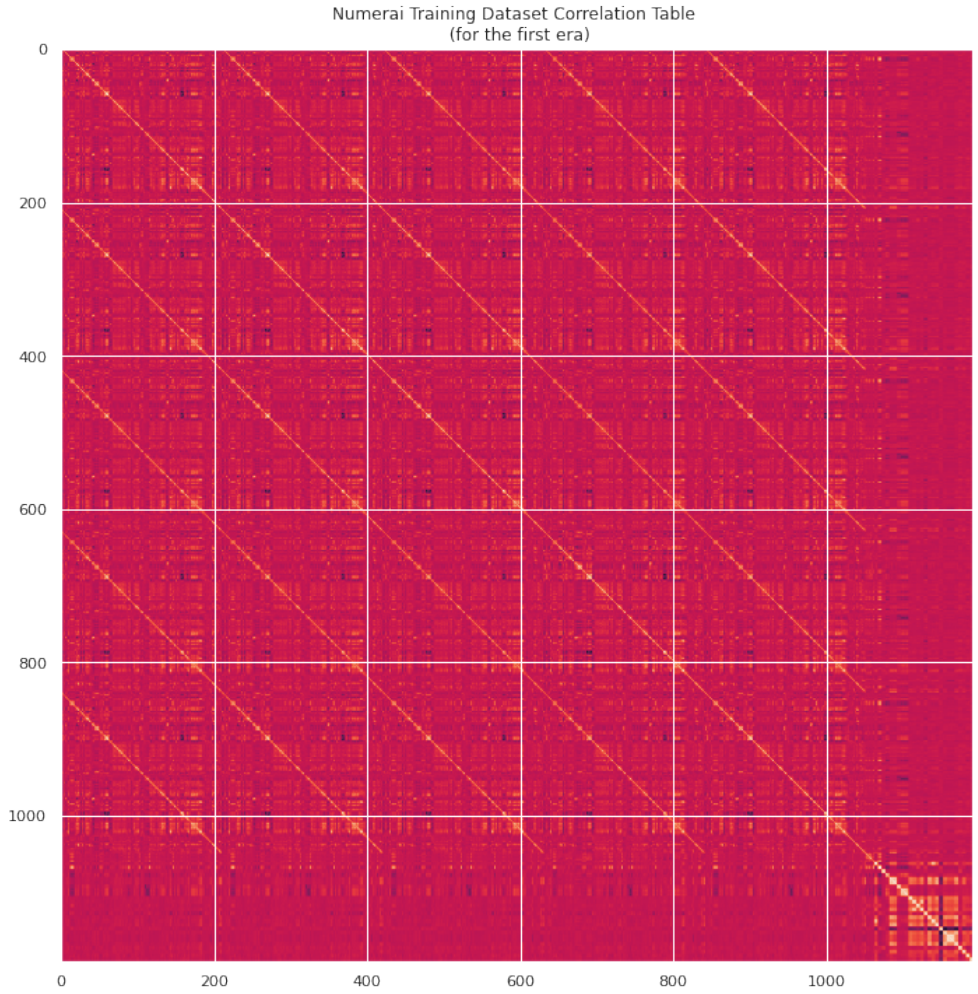


FIGURE 5.6: Numerai Correlation table (era=1)

5.3.3 Targets

In the V4 Numerai Tournament dataset, there are 20 distinct targets. Fundamentally, there are 10 different types of targets, each one constructed with a **20-day** and a **60-day** window. Numerai doesn't provide precise information about the type of targets, how they were calculated, or what they describe.

Competition participants are only scored on a single target, which currently corresponds to **target_nomi_20**, but this could change in the future as the Numerai investment strategy shifts in different directions.

Targets are also **normalized and obfuscated** as the features. Most of the targets are discrete and take only 5 different values following the distribution presented on Table 5.2 and Fig 5.9. Other targets have different binning distributions than what you see with *target_nomi_20*, e.g. 7 bins rather than 5, with less rigid constraints on samples per bin, as shown in Table 5.3

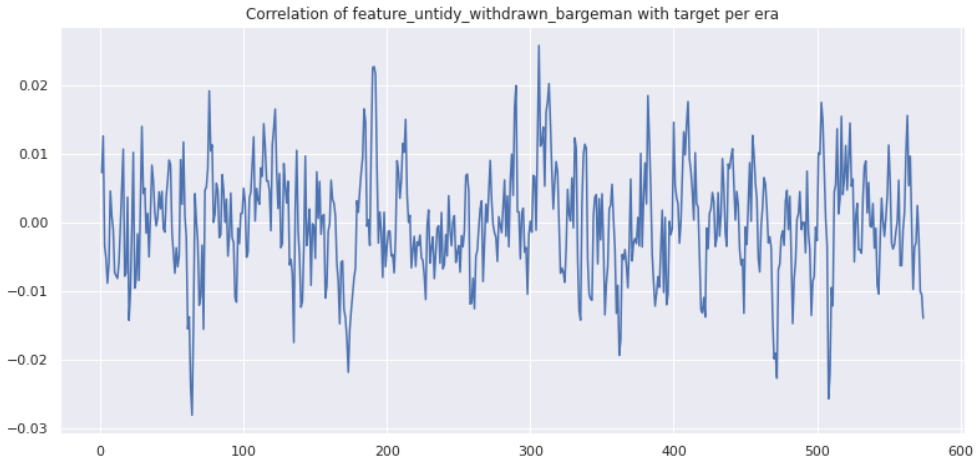


FIGURE 5.7: Correlation of Feature with the Target

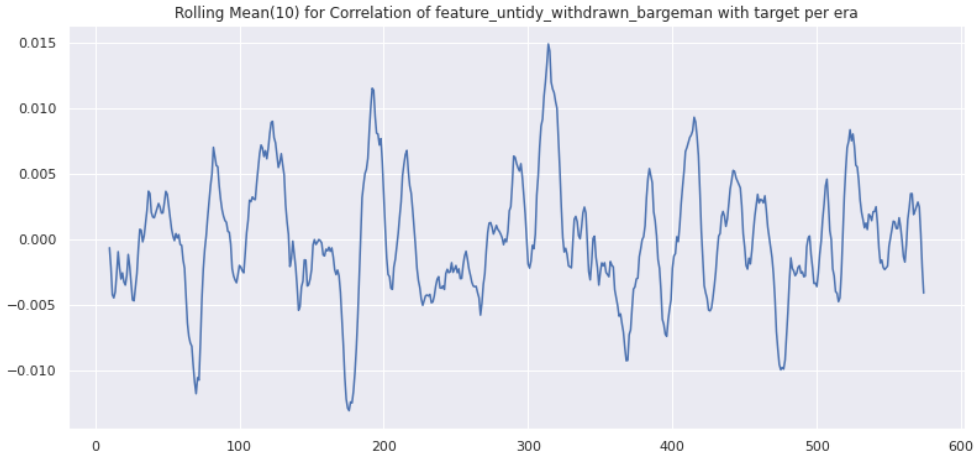


FIGURE 5.8: Correlation of Feature with the Target (Rolling Average (10 eras)

Targets Correlation

Targets have a wide range of correlations with each other from 0.3 to 0.9. This fact allows for the construction of many diverse models that ensemble nicely. Training models to be good at multiple targets and/or ensembling models trained on different targets is a great way to improve generalization performance and increase the model's uniqueness.

Targets overlapping

Targets are calculated in a 20-day or 60-day window, which means an overlap between consecutive eras occurs. Eras are weekly (5 days), but the target is four weeks out (20 days). Diagram 5.10 explains this process in a clear visual format. In order to prevent models from overfitting due to overlapping eras, participants can subsample the eras

Target Value	Percentage
0.00	5%
0.25	20%
0.50	50%
0.75	20%
1.00	5%

TABLE 5.2: 5-bin target distribution

Target Value	target_william_v4_20	target_thomas_v4_20
0.00	0.03%	0.02%
0.16	0.07%	0.05%
0.33	0.20%	0.20%
0.50	0.40%	0.46%
0.66	0.20%	0.20%
0.83	0.07%	0.05%
1.00	0.03%	0.02%

TABLE 5.3: 7-bin target distribution

(every 4th era). Then they can ensemble the models trained on each of the 4 different ways to subsample.

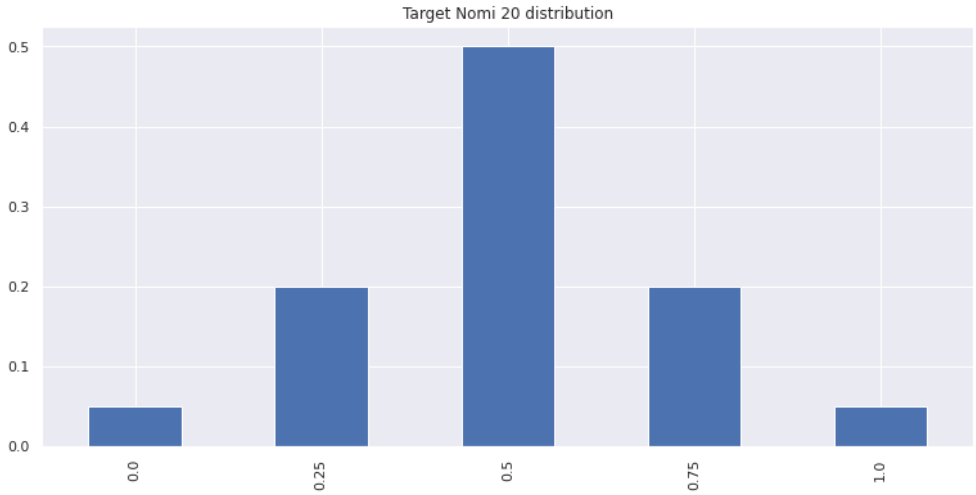


FIGURE 5.9: Target Distribution (target_nomi_20)

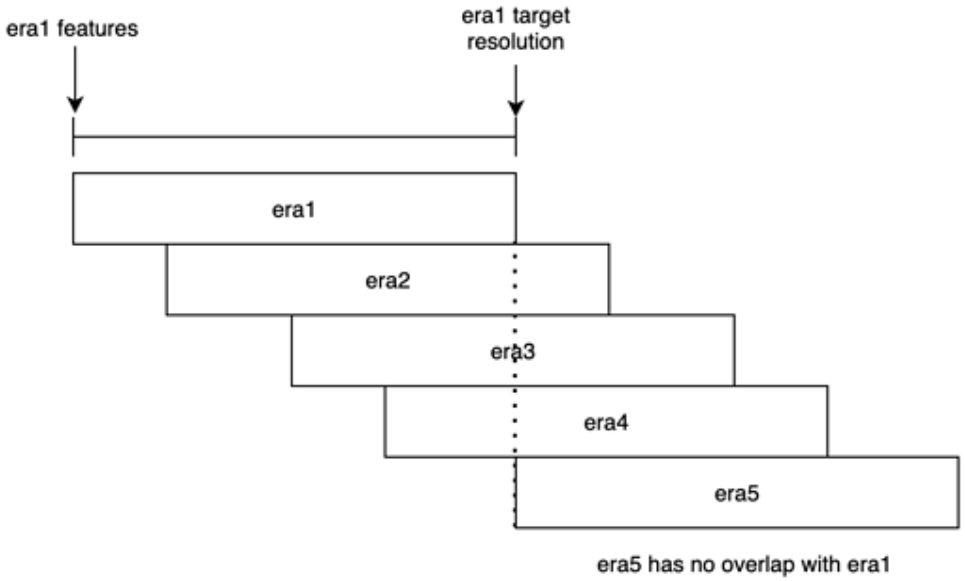


FIGURE 5.10: Numerai Targets

6 Numerai Tournament Evaluation Metrics

Numerai has introduced a wide set of evaluation metrics to assess the performance of models. These metrics aim to better inform users about the strengths and weaknesses of their models and give them more direction and insight into the nuances of this unique problem.

This chapter provides a detailed analysis of the **Evaluation Metrics** used in the Numerai Tournament, along with **code snippets** and **example calculations**.

6.1 Correlation (Spearman's correlation)

The primary evaluation metric of the Numerai Tournament is the Spearman Correlation between predictions and targets.

In statistics, Spearman's rank correlation coefficient or Spearman's ρ, named after Charles Spearman[1] and often denoted by the Greek letter ρ, is a nonparametric measure of rank correlation. It assesses how well the relationship between two variables can be described using a monotonic function.

- **Rank** is the relative position label of the observations within a variable.
- **Correlation** is a measure of the statistical dependence between two variables.

Spearman's correlation is an appropriate metric for both continuous and discrete ordinal variables. The score is always between -1 (perfect negative correlation) and 1 (perfect correlation). Intuitively, Spearman's correlation between two variables will be high when observations have a similar rank and low when observations have a dissimilar rank.

The mathematical formula to calculate Spearman's correlation when there are no tied ranks is,

[1] https://en.wikipedia.org/wiki/Charles_Spearman

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}$$

Where,

- d: is the difference between ranks
- n: is the total number of observations

In cases with tied ranks, the full version of Spearman's formula must be used:

$$\begin{aligned} \rho &= \frac{Covariance(R(X), R(Y))}{St.Dev.(R(X)) * St.Dev.(R(Y))} \\ &= \frac{\frac{1}{n} \sum_{i=1}^{n} \left[(R(X_i) - \overline{R(X)}) \cdot (R(Y_i) - \overline{R(Y)}) \right]}{\sqrt{\left[\frac{1}{n} \sum_{i=1}^{n} (R(X_i) - \overline{R(X)})^2 \right] \cdot \left[\frac{1}{n} \sum_{i=1}^{n} (R(Y_i) - \overline{R(Y)})^2 \right]}} \end{aligned} \tag{6.1}$$

Where,

- R(X) and R(Y) are the ranks of X,Y variables
- $\overline{R(X)}$ and $\overline{R(Y)}$ are the mean ranks

Spearman's correlation between two variables is equal to the Pearson's correlation between the rank values of those two variables. Pearson's correlation assesses linear relationships, while Spearman's correlation assesses monotonic relationships (whether linear or not). If there are no repeated data values, a perfect Spearman's correlation of +1 or 1 occurs when each variable is a perfect monotone function of the other.

In financial machine learning, accurate ranking plays a critical role. Investors want to know what assets are likely to outperform others to take long and short positions accordingly.

Numerai is interested in high Spearman's correlation, which means that prediction is evaluated based on the relative ranking of the equities they imply in each era and not based on raw values.

6.1.1 Example calculations of Spearman's Correlation

Remember that targets distribution for **target_nomi_20** is

$[0.00 : 5\%, 0.25 : 20\%, 0.50 : 50\%, 0.75 : 20\%, 1.00 : 5\%]$.

Assume a random sample of 20 observations and example predictions generated by a machine learning pipeline. The following table explains Spearman's correlation calculations.

TABLE 6.1: Spearman Correlation example calculations

	Target (X)	**Example predictions (Y)**	R(X)	$R(X) - \overline{R(X)}$	R(Y)	$R(Y) - \overline{R(Y)}$	Sum Diffs
1	1.00	0.823	20.00	9.50	20	9.50	90.25
2	0.50	0.519	10.50	0.00	10	-0.50	0.00
3	0.50	0.566	10.50	0.00	13	2.50	0.00
4	0.25	0.432	3.50	-7.00	5	-5.50	38.50
5	0.75	0.633	17.50	7.00	14	3.50	24.50
6	0.50	0.451	10.50	0.00	6	-4.50	0.00
7	0.25	0.409	3.50	-7.00	4	-6.50	45.50
8	0.50	0.342	10.50	0.00	2	-8.50	0.00
9	0.50	0.654	10.50	0.00	16	5.50	0.00
10	0.75	0.650	17.50	7.00	15	4.50	31.50
11	0.50	0.764	10.50	0.00	18	7.50	0.00
12	0.50	0.454	10.50	0.00	7	-3.50	0.00
13	0.25	0.333	3.50	-7.00	1	-9.50	66.50
14	0.50	0.490	10.50	0.00	9	-1.50	0.00
15	0.75	0.761	17.50	7.00	17	6.50	45.50
16	0.50	0.792	10.50	0.00	19	8.50	0.00
17	0.25	0.457	3.50	-7.00	8	-2.50	17.50
18	0.50	0.543	10.50	0.00	11	0.50	0.00
19	0.00	0.354	1.00	-9.50	3	-7.50	71.25
20	0.75	0.556	17.50	7.00	12	1.50	10.50

With,

- $SumDiffs = \sum \left[(R(X) - \overline{R(X)}) \cdot (R(Y) - \overline{R(Y)}) \right] = 441.5$

It is,

	Mean	**Std**
R(X)	10.5	5.49
R(Y)	10.5	5.92

Remember, Spearman's correlation formula for variables with ties is,

$$
\begin{aligned}
\rho &= \frac{cov(R(x), R(y))}{\sigma_{R(x)}\sigma_{R(y)}} \\
&= \frac{\frac{1}{n}\sum_{i=1}^{n}\left[(R(x_i) - \overline{R(x)}) \cdot (R(y_i) - \overline{R(y)})\right]}{\sqrt{\left[\frac{1}{n}\sum_{i=1}^{n}(R(x_i) - \overline{R(x)})^2\right] \cdot \left[\frac{1}{n}\sum_{i=1}^{n}(R(y_i) - \overline{R(y)})^2\right]}} \\
&= \frac{\frac{441.5}{19}}{5.49 \cdot 5.92} \\
&= \frac{23.24}{5.49 * 5.92} \\
&= 0.716
\end{aligned} \tag{6.2}
$$

Note that we use the sample covariance and sample standard deviation formulas, with (n-1) instead of n, in the denominator.

6.2 Meta-Model Contribution (MMC)

Numerai uses the individual predictions submitted by competitors to form the **Numerai Meta-Model**. This is a weighted ensemble of all predictions based on each model's staking amount.

$$
\begin{aligned}
MetaModel = stake_model1 * predictions_model1 + ... \\
+stake_modelN * predictions_modelN
\end{aligned}
$$

This Meta-Model signal is used to construct Numerai's Hedge Fund portfolio.

Because Spearman's correlation is the main evaluation metric of the Numerai tournament, each user is incentivized to maximize their individual correlation score. But Numerai wants to maximize the Meta-Model's correlation score. **Meta-Model contribution (MMC)** is designed to bridge this gap. Whereas correlation rewards individual performance, MMC rewards contribution to the Meta-Model's correlation or group performance.

An important factor directly affecting Meta-Model's performance is the ensemble diversity. Ensemble diversity refers to differences in the decisions or predictions made by the ensemble members. Numerai wants to construct its Meta-Model based on a **diverse set of models** because this variety will increase the performance of the ensemble model. Having just different versions of tree-based models can not

help Numerai achieve outstanding performance. Thus, a model with a high correlation score with targets and a low correlation score with other models will significantly improve the Meta-Model.

Meta-Model Contribution (MMC) calculations

MMC is a measure that aims to quantify the contribution of each individual model to the final correlation between the Meta-Model and the equities performance (target). To calculate a user's (U) MMC for a given round, Numerai executes the following process;

1. Select a random 67% of all staking users (with replacement).
2. Calculate the stake-weighted predictions of these users.
3. Transform the stake-weighted predictions and U's model predictions to be uniformly distributed.
4. Neutralize U's model with respect to the uniform stake weighted predictions.
5. Calculate the covariance between U's model and the targets.
6. Divide this value by 0.0841 (this step is to bring the expected score up to the same magnitude as correlation).

The resulting value is the MMC score for iteration 1. Numerai repeats this process 20 times and keeps the average MMC score as the final MMC.

Uniform tranformation in *Step 3* can be achieved using the ***probability integral transform*** theorem[2]. This theorem states that if X is a continuous random variable with cumulative distribution function F_X, then the random variable $Y = F_X(X)$ has a uniform distribution on [0,1].

Beyond MMC

Numerai used MMC as a supplementary evaluation metric to quantify the performance of a model's prediction in alignment with the portfolio and actual values. As of March 2022, MMC was **replaced** by a different metric, the **True Contribution (TC)**, which is explained in the next section.

6.3 True Contribution (TC)

Alignment between the performance of tournament participants and Numerai hedge fund profitability is a key element in the construction of Numerai. If a model is ranked at the top of the Numerai leaderboard, it should be because it is helping to improve the profitability

[2] https://en.wikipedia.org/wiki/Probability_integral_transform

of the hedge fund the most. Before introducing the Total Contribution metric, users were evaluated only at the signal level: how well their signal correlates with the target (CORR) and their contribution to the Meta Model signal (MMC).

However, Numerai's portfolio is created by running a **custom optimizer** on the Meta-Model signal. The optimizer enforces constraints and penalties on the portfolio that affect which aspects of the Meta-Model signal are reflected in the final portfolio.

$$NumeraiPortfolio = custom_optimizer(stake_model1 * predictions_model1 + ... + stake_modelN * predictions_modelN)$$

This can create a divergence between what appears to be a good model at the signal level and a model that is truly helping the fund create better portfolios.

For example, the optimizer penalizes **feature exposure** (correlation of predictions with particular features), and thus large feature exposures in the Meta-Model signal will not be reflected in the final portfolio. A model with a high feature exposure will greatly correlate with the target (for a while); however, its signal will have limited influence on the portfolio since the feature exposure of the portfolio is constrained. Such a model could earn large payouts without ever contributing much information to the portfolio, which is undesirable.

To better align the evaluation of users and the Numerai hedge fund performance, the Numerai team introduced a new metric called **"True Contribution"**. This metric aims to estimate how much a user's signal improves or detracts from the returns of Numerai's portfolio. Using this metric for payouts, user incentives, and hedge fund performance is in perfect alignment. With True Contribution as the payout metric, a user's stake would increase if their model increased portfolio returns and decrease (burn) if the model reduced returns.

True Contribution calculation (initial approach)

In the initial formulation of the True Contribution score, the Numerai team used a Leave-one-out approach to quantify each user's individual performance. The process was as follows.

- Create the Stake-Weighted Meta-Model (SWMM) by leaving, in turn, each user out.
- Utilize the production optimizer to generate the corresponding portfolios and calculate the returns.

- Finally, compare the returns above with the entire Stake-Weighted Meta-Model's returns (considering the previously left out user's predictions) to calculate "True Contribution".

This definition was problematic for many reasons:

1. A user's contribution is heavily dependent upon their stake, and therefore identical signals with different stakes will get different scores.
2. Users with 0 stake would always have 0 contributions; accordingly, there is no way to calculate the metric for unstaked users.
3. Users with small stakes would always have 0 contributions.
4. The production optimizer starts from the current portfolio and enforces turnover constraints; hence, the TC scores are heavily dependent on past portfolios, which users have no knowledge of or control over

True Contribution calculation (latest approach)

The latest version of Total Contribution cures all previous issues while retaining the realism of portfolio construction and returns.

The Numerai team realized that the leave-one-user-out method is just an approximation for a gradient calculation. The Total Contribution (TC) metric aims to quantify how changing a user's stake changes the portfolio returns, which is the gradient of portfolio returns with respect to users' stakes.

A true gradient calculation would also have the nice properties that

- it can be computed for all users simultaneously from a single portfolio optimization rather than computing a different optimization for each user held out
- it will assign the same values to identical signals with different stakes
- it will assign proper values to 0 stakes

This first property is vital to eliminate the computational costs required for calculations, while the second and third properties are essential for fairness in the tournament.

Performing an accurate gradient calculation would require taking a derivative through the portfolio optimizer. This seemingly difficult task can be accomplished using **cvxpylayers**[3] package. This remarkable package based on an award-winning research paper by Akshay

[3] https://github.com/cvxgrp/cvxpylayers

Agrawal, 2012 allows to include a *cvxpy* defined convex optimization as a layer in a **PyTorch** model.

Interested users can find more information about this in the Numerai official forum[4].

Eventually, the True Contribution (TC) metric is calculated as follows;

The following explanation is adapted from Richard Craib's Medium Post titled **Alien Stock Market Intelligence**[5]

Figure 6.1 outlines the frameworks used by Numerai to calculate the Total Contribution.

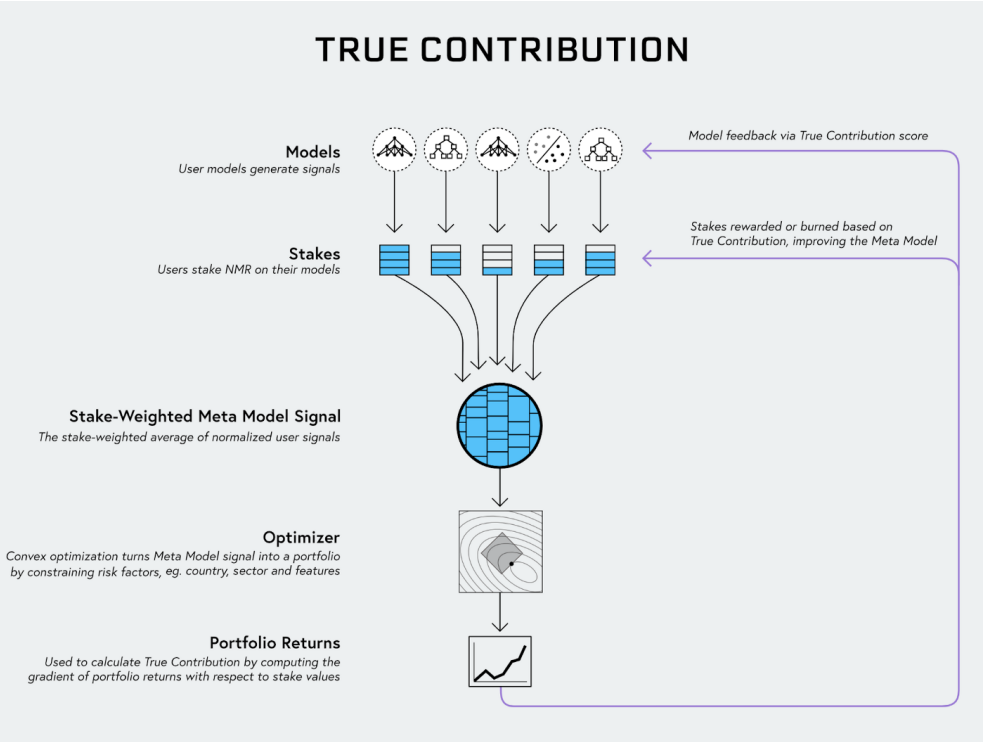


FIGURE 6.1: True Contribution calculation framework

1. Numerai combines submitted predictions via the stake-weighted average of every signal to create the Stake-Weighted Meta-Model (SWMM). A data scientist who stakes a large amount of NMR on their model will have a larger weight in the Stake-Weighted Meta Model.

2. The SWMM and hundreds of risk constraints (e.g., market, country, and sector risk neutralization) are input into an optimizer.

[4] https://forum.numer.ai/t/true-contribution-details/5128/10

[5] https://medium.com/numerai/alien-stock-market-intelligence-numerais-true-contribution-6bc7652bd6ac

This transforms the SWMM into a realistic hypothetical portfolio.

3. Numerai observes the hypothetical returns of that portfolio and calculates the gradient of the optimized portfolio return with respect to the stake. True Contribution is the magnitude of this gradient for each set of predictions constituting the SWMM.

6.4 More evaluation Metrics and the Diagnostic Tool

Besides Spearman's correlation, Meta-Model Contribution, and True Contribution, Numerai uses numerous evaluation metrics to assess different aspects of a model's performance. These evaluation metrics are split into 4 categories.

- **Performance**: Overall measures of performance
- **Risk**: Different ways to assess how likely it is that a model will have a severe decrease in performance in the future
- **MMC (Meta-Model Contribution)**: Different estimates of how a model would perform from an MMC perspective on validation data.
- **Other**: Other useful metrics evaluating the quality of the submitted predictions and the financial projections.

Diagnostics Tool

Diagnostics Tool (Fig. 6.2) is a web tool provided by Numerai to help participants analyze the historical performance of their models over the validation dataset. Users can upload files with predictions for the numerai validation dataset and get a detailed report evaluating their predictions. The Diagnostics tool can be used multiple times without affecting tournament submissions.

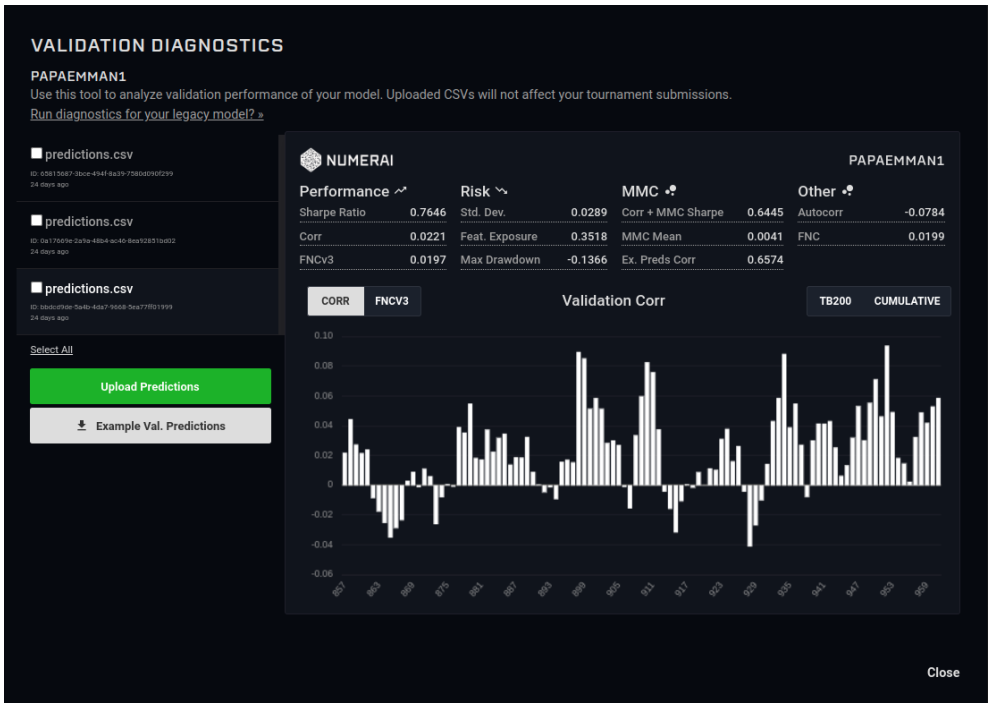


FIGURE 6.2: Diagnostics Tool

Validation Correlation plot

Validation Correlation plot is the main plot at the center of the diagnostics tool. This barplot visualizes the correlation between predictions and targets for eras 857 to 961 in the validation dataset. Also, per-era FNCv3 (Feature Neutral Correlation v3) is available in the diagnostics chart.

Based on this plot, users can individually assess their models' performance in each era (bar) and identify regions with suboptimal performance. Ensembling models based on observations from this plot is a great way to boost performance. For example, combining models with divergent behavior is an excellent starting point. A rule of thumb to select divergent models and ensure the ensembling diversity is to find models with low correlation between their correlation scores. This will ensure that when the first model has a low correlation in a specific regime, another model will have a high correlation in the same regime and vice-versa.

Validation metrics

In the table on top of this plot, there are 11 evaluation metrics that fall into the categories mentioned above; **Performance** (Sharpe ratio, Spearman's correlation, Feature Neutral Correlation) , **Risk** (Std. Dev., Feature Exposure, Max Drawdown), **MMC** (Corr+MMC Sharpe, MMC Mean, Example predictions correlation), and **Other**.

1. **Sharpe ratio**

   The Sharpe ratio was developed by Nobel laureate William F. Sharpe[6].It is used to help investors understand the return of an investment (ROI) compared to its risk. It is the ratio of mean returns divided by the standard deviation of returns. Therefore, it describes the returns distribution. The preferred distribution has a high mean and narrow distribution (and small tails). Consequently, a high Sharpe ratio is good compared to similar portfolios or funds with lower returns. The Sharpe ratio has several weaknesses, including an assumption that investment returns are normally distributed. In the Numerai Tournament, Sharpe Ratio is calculated as the mean of the per-era correlation divided by the standard deviation of the per-era correlation.

   $$SharpeRatio = \frac{corr}{std(corr)} \tag{6.3}$$

   Based on empirical results from the experimentation process, a good model will have a Sharpe Ratio > 1.0 on validation data, and an excellent model will have Sharpe > 1.2.

2. **Corr (Spearman's correlation)**

   This is the mean of the per-era Spearman's correlations between the prediction and the targets. (The nominator of the Sharpe ratio)

3. **FNCv3 (Feature Neutral Correlation)**

   This metric is calculated as the mean of the per-era Spearman's correlations after the predictions have been neutralized to all features. Neutralization is a core concept in the Numerai Tournament, and it will be described in detail in the following section.

4. **Std. Dev.**

   The standard deviation of per era correlations between the predictions and the validation targets. (The denominator of the Sharpe ratio)

5. **Feature Exposure**

   The maximum correlation the predictions have with any single feature.

6. **Max Drawdown**

   The most significant cumulative decrease between two eras in terms of validation correlation. Max Drawdown describes the

[6] https://en.wikipedia.org/wiki/William_F._Sharpe

maximum loss in the returns in a single round. The closer to zero, the max drawdown, the better. Based on empirical results from the experimentation process, a good model will have a max drawdown > -0.04.

7. **(Corr + MMC) Sharpe**

   This is the mean (Corr+MMC) scores per era divided by the standard deviation (Corr+MMC) scores per era.

8. **MMC Mean**

   The estimated Meta Model Contribution (MMC) per validation era. For MMC metrics calculations, the example predictions provided by Numerai are used as a proxy to the Meta-Model.

9. **Corr With Example Preds**

   The correlation between model predictions and the example predictions.

10. **APY (Annual Percentage Yield)**

    The Annual Percentage Yield over the validation period. The APY is the real rate of return earned on an investment, considering the effect of compounding interest. Unlike simple interest, compounding interest is calculated periodically, and the amount is immediately added to the balance. With each period going forward, the account balance gets a little bigger, so the interest paid on balance gets bigger as well.

11. **Autocorrelation**

    This is the autocorrelation of the per-era correlation. Autocorrelation represents the degree of similarity between a given time series and a lagged version of itself over successive time intervals. In other words, autocorrelation measures the relationship between a variable's current value and its past values.

There are 2 additional **options** in the Validation Diagnostics tool.

- **TB200** scores only the highest 200 predictions and the lowest 200 predictions. This evaluates models' predictions based only on the extreme (higher and lower) predictions. TB200 on FNC means predictions are first neutralized, then TB200 Corr of those predictions are taken
- **Cumulative** shows the cumulative validation score through eras.

Every metric in the table is *colored* based on ratings. These ratings are defined based on the distribution of submitted predictions in the tool.

For example, a submission better than 99% of total validation submissions will be green.

1. Bad (Red): 0-30%
2. Fair (White): 31-85%
3. Excellent (Green): 86-100%

7 Numerai Tournament rewards; Staking, Scoring, and Payouts

7.1 The Numerai Tournament timeline

7.1.1 Weekly Round

Competing in Numerai is an ongoing, never-ending challenge. Every week on Saturday, 18:00 UTC, a new round starts, and Numerai releases a new live dataset. Competitors have almost 2 days to submit their predictions based on the newly released data until the submission deadline on Monday at 14:30 UTC.

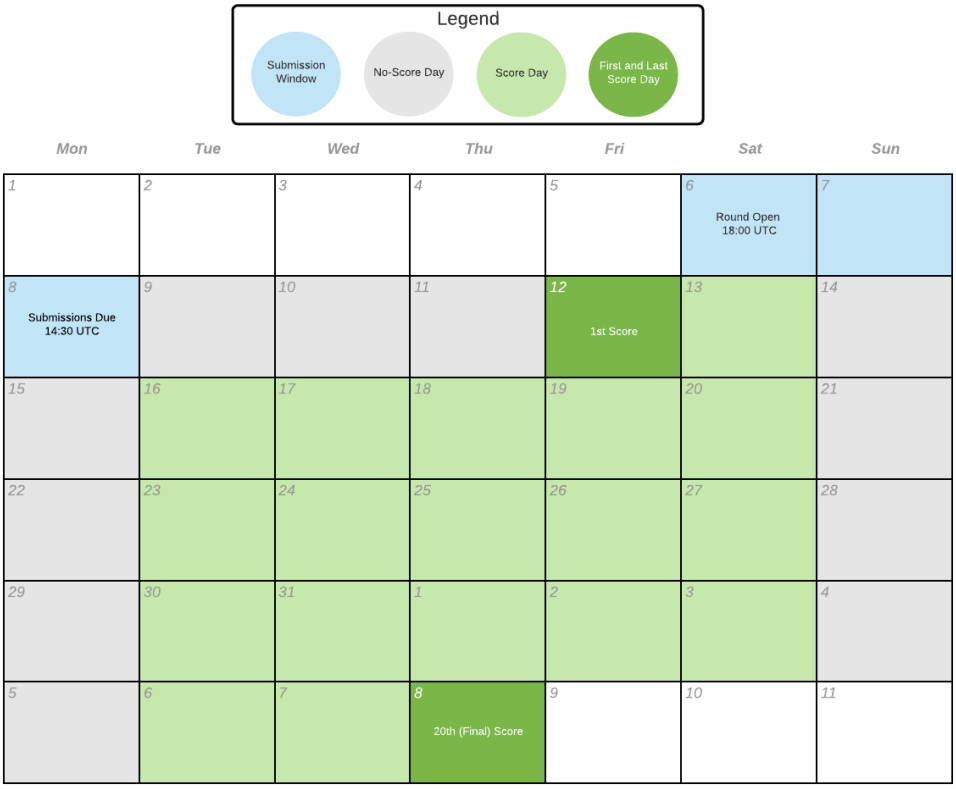


FIGURE 7.1: Numerai Tournament Round

The **live dataset** contains data for a **single era**; therefore, features correspond to the prior business week (Monday-Friday). Because the current target ***(target_nomi_20)*** corresponds to a 20-day window target (counting only working days when the Stock Market is open), a round takes 4 weeks to resolve. Predictions are evaluated for the first time on Friday, and after that, they scored against the live Stock Market over the next 4 weeks, for 5 days a week (20 consecutively business days), as shown in Fig. 7.1.

Since a single round takes 4 weeks to resolve, but the submissions are weekly, if a competitor submits new predictions every week, they will receive multiple (up to 4) overlapping scores on each scoring day from the 4 ongoing rounds. Essentially, after the 4 initial weeks, each era will have 4 different predictions, one from each of the 4 previous weeks, as shown in Fig. 7.2.

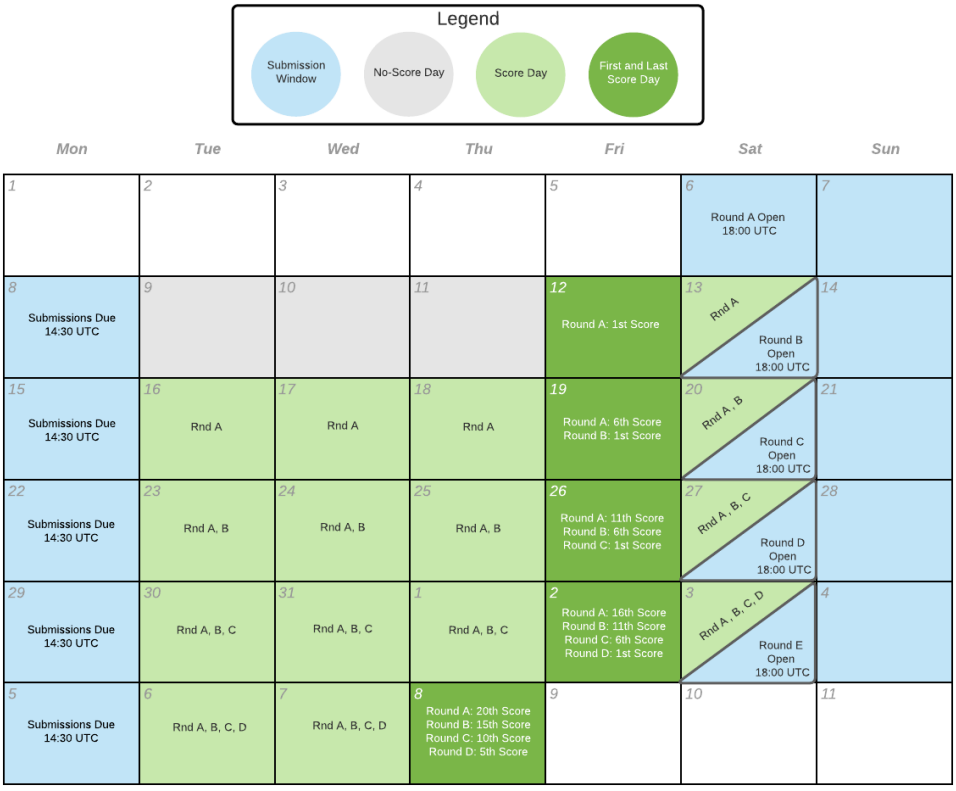


FIGURE 7.2: Numerai Tournament Overlapping scores

7.1.2 Daily scores and Round resolution

After a business day, Numerai releases the daily performance metrics. These daily scores correspond to the current state of the market on that specific day, with 1 day lag from the real-life market (Friday scores are from Thursday's market, etc.). Because of this, there aren't any scores

on Sundays or Mondays because the Stock Market is closed on the previous day.

During the daily evaluation, Numerai also computes a predicted payout value, which is updated with every daily score.

Even though submissions are scored against the stock market daily for four weeks, only the final score after the 20 days will be used to calculate rewards. Participants can track daily payouts from the **model's profile page**[1] (Fig. 7.3) or the community-built **Numerai Payouts app**[2] (Fig. 7.6).

The model's scores can be viewed **publicly** on their profile page, both within a single round and across multiple rounds.

Performance Metrics within a round - Daily metrics

Fig. 7.3, visualizes the performance (correlation and total contribution) of a single model for a specific round, from April 15 to May 12. Note that only the final numbers at the top of the table matter to calculate the net gain/loss of NMR for that round.

So, this model has the following metrics at the round resolution.

- **Corr**=0.019
- **MMC**=-0.0209
- **FNC** = -0.0159
- **FNCv3** = -0.0201
- **TC** = 0.076615

As discussed in Chapter 6, positive correlation (CORR) and total contribution (TC) leads to positive payouts. Accordingly, this model will have a positive reward at the end of the round.

Performance metrics across rounds

Fig. 7.4 shows an example of a model's final performance scores (correlation and total contribution) over the past 19 rounds. Note that the 3rd (gray) line represents the percentile ranking of the model for this particular round compared with all the other submitted models. Therefore, for the last 4 rounds, there aren't values for this metric as the rounds are still in-progress.

[1] https://numer.ai/papaemman1
[2] https://numeraipayouts.com/



DATE	CORR	MMC	FNC	FNCV3	TC	PAYOUT
May 12	0.0190	-0.0209	-0.0159	-0.0201	0.076615	+ 0 NMR
May 11	0.0037	-0.0255	-0.0219	-0.0264	0.076946	+ 0 NMR
May 10	0.0118	-0.0211	-0.0096	-0.0161	0.056516	+ 0 NMR
May 7	0.0225	-0.0136	0.0018	-0.0079	0.026339	+ 0 NMR
May 6	0.0341	-0.011	0.0122	0.0022	0.044534	+ 0 NMR
May 5	0.0409	-0.0063	0.0153	0.0060	0.040242	+ 0 NMR
May 4	0.0458	-0.0042	0.0171	0.0128	0.03312	+ 0 NMR
May 3	0.0261	-0.0097	0.0046	-0.0083	0.015799	+ 0 NMR
Apr 30	0.0304	-0.0082	0.0088	-0.0048	0.009424	+ 0 NMR
Apr 29	0.0157	-0.017	0.0118	-0.0011	0.010889	+ 0 NMR
Apr 28	0.0162	-0.0153	0.0047	-0.0043	0.012605	+ 0 NMR
Apr 27	0.0274	-0.0106	0.0015	0.0017	0.003648	+ 0 NMR
Apr 26	0.0242	-0.0187	-0.0012	-0.0081	-0.001047	+ 0 NMR
Apr 23	0.0097	-0.0147	-0.0031	-0.0127	0.002243	+ 0 NMR
Apr 22	-0.0028	-0.020	-0.0117	-0.0102	-0.01039	- 0 NMR
Apr 21	-0.0024	-0.0157	-0.002	-0.0074	-0.008542	- 0 NMR
Apr 20	-0.0387	-0.024	-0.0121	-0.0138	-0.004952	- 0 NMR
Apr 19	-0.013	-0.0074	0.0207	0.0097	-0.006526	- 0 NMR
Apr 16	-0.0165	-0.0136	-0.0003	-0.0017	-0.011016	- 0 NMR
Apr 15	0.0314	0.0012	0.0175	0.0092	-0.007788	+ 0 NMR

FIGURE 7.3: Numerai Daily Performance Scores

Additionally, at the end of each round, Numerai sends a **notification email** to participants with **Round Summary metrics**. This email informs competitors about their model's round performance, the relative ranking compared to other models, and the total payout.

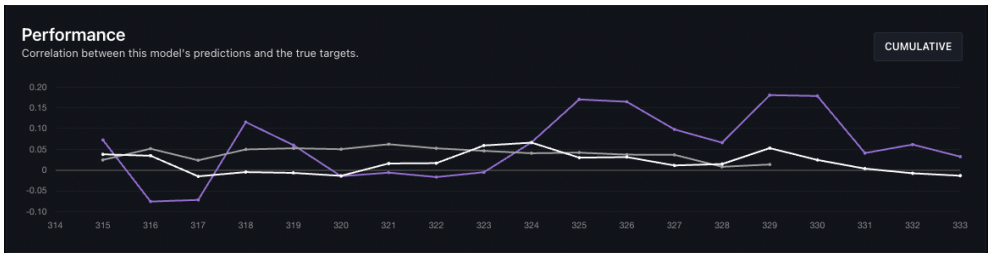


FIGURE 7.4: Model's Round Performance

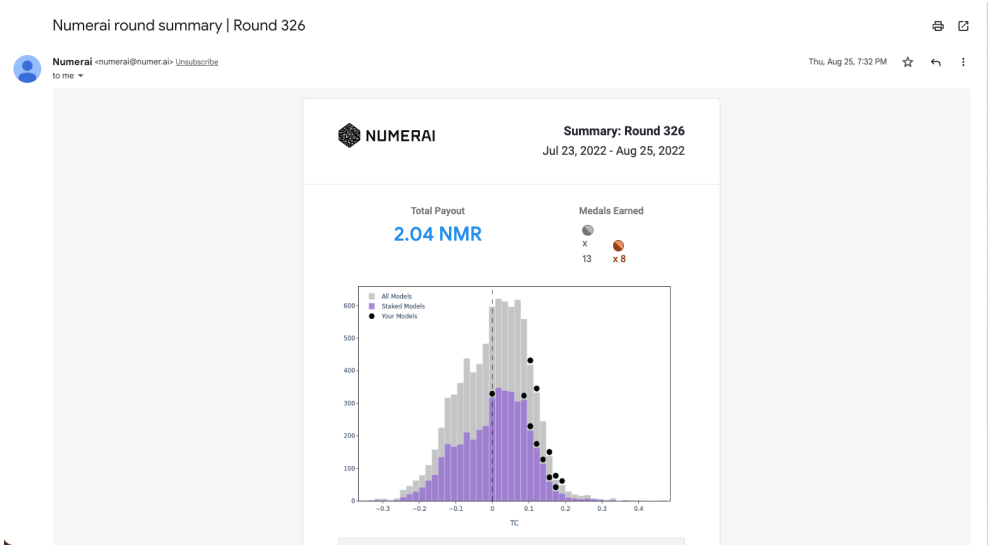


FIGURE 7.5: Email notification with Round summary

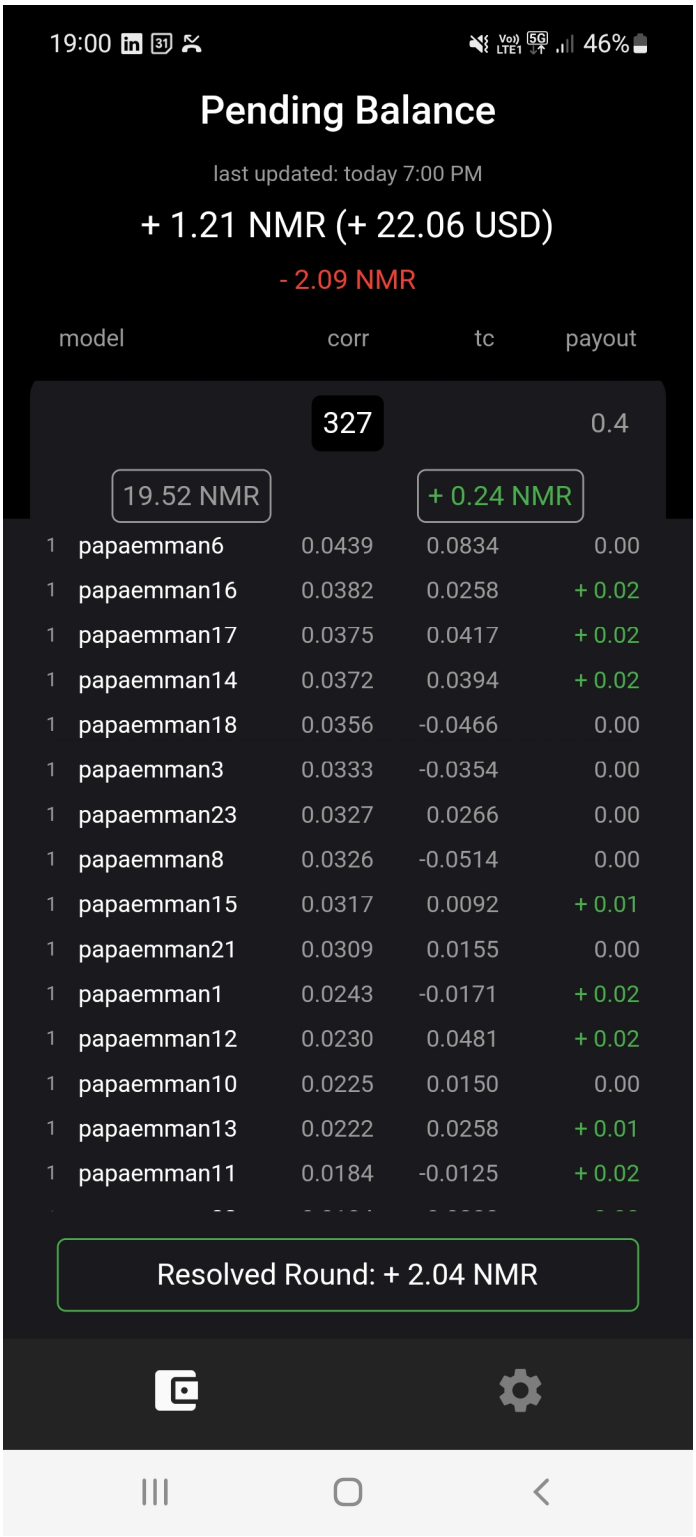


FIGURE 7.6: Numerai Payouts App

7.2 Staking and Payouts

7.2.1 Staking Motivation

As discussed in previous chapters, the Numerai Hedge Fund is controlled by the predictions submitted by Data Scientists participating in the Numerai Tournament. Participation in Numerai is anonymous and requires no authentication. This creates an information asymmetry between Numerai and competitors because the Numerai team hasn't any instrument to evaluate how suitable a model's predictions will be. Also, the fact that the Numerai tournament is open to the public without any restrictions makes the system vulnerable to Sybil attacks if no other controls apply.

In the context of the Numerai Tournament, an adversary could potentially create multiple accounts and try to influence Numerai's investment strategy maliciously. Thus Numerai had to develop a rewarding mechanism to align the interests of Numerai Hedge Fund and competitors to prevent this behavior.

Designing a payout system that removes the information asymmetry between Numerai and Data Scientist and, at the same time, is resistant to Sybil attacks is not a trivial task. In the early days, Numerai introduced a rewarding mechanism to tackle this problem, and since then, this mechanism has been under continuous evaluation and improvement to ensure the alignment of interests between Numerai and competitors

The main idea behind the payout system is **Staking**!

Staking is essentially a type of **bet** that participants make against the performance of their models. Staking enables the necessary skin-in-the-game for participants. If they think their models are suitable, they have to stake them and make themselves accountable for their predictions. Staking is based on NMR cryptocurrency and essentially demonstrates the user's confidence in a model. The amount of NMR staked allows Numerai to quantify the user's trust in this model. Only staked models form part of the Numerai meta-model, which controls the Numerai hedge fund.

Numerai wants accurate predictions and is willing to reward them. Staking gives Numerai a mechanism for weighting prediction for the Numerai Meta-Model and a Sybil-resistant way to allocate payouts to users. Since payouts are a percentage of the staked value, participants aren't motivated to create duplicate accounts.

This payout mechanism is a crucial component of the Numerai Tournament. It makes a potential Sybil attack a not financially sustainable

action for the attacker and ensures that competitors and Numerai have their incentives aligned.

7.2.2 Staking mechanics

As discussed in Chapter 6, model predictions are primarily scored on the **True Contribution (TC) to the Meta Model** and the **Correlation (CORR)** with the targets. Participants can optionally stake NMR on their models to earn or burn tokens based on TC and CORR scores. This is the only way for someone to earn rewards from the Numerai tournament.

Predictions are also scored on other metrics such as **meta-model contribution (MMC) and feature neutral correlation (FNC)**, but currently, there isn't any staking mechanism based on them.

Keep in mind that staking in Numerai isn't necessary. Users can submit predictions to Numerai without staking. But to start earning payouts based on the live performance, it is essential to stake NMR. Staking in Numerai has risks; therefore, it is advised that users mustn't risk money they can't afford to lose.

The **rewarding mechanism** depends on 3 different factors

1. Numerai's payout for that week
2. The amount staked by the user
3. Performance of contributed signals over the next month on the live data

There are no rewards based on backtests or the test set.

How to manage stakes

Numerai offers a web interface for participants to manage their stake in models (Fig. 7.7). In technical terms, staking means locking up NMR in a smart contract on the Ethereum blockchain. For the duration of the stake, Numerai is permitted to add payouts to or burn from the NMR locked up.

This interface interacts with the user's Numerai Wallet and the Numerai smart contract and allows NMR to be transferred between them.

When the stake increases, NMR is transferred from the wallet to the staking contract. When the stake decreases, NMR is transferred from the model's stake into the wallet after a 4-5 week delay based on when the stake release request was made.

Note: If the stake release request was made before the current submission deadline, the release will be effective on the first scoring day of

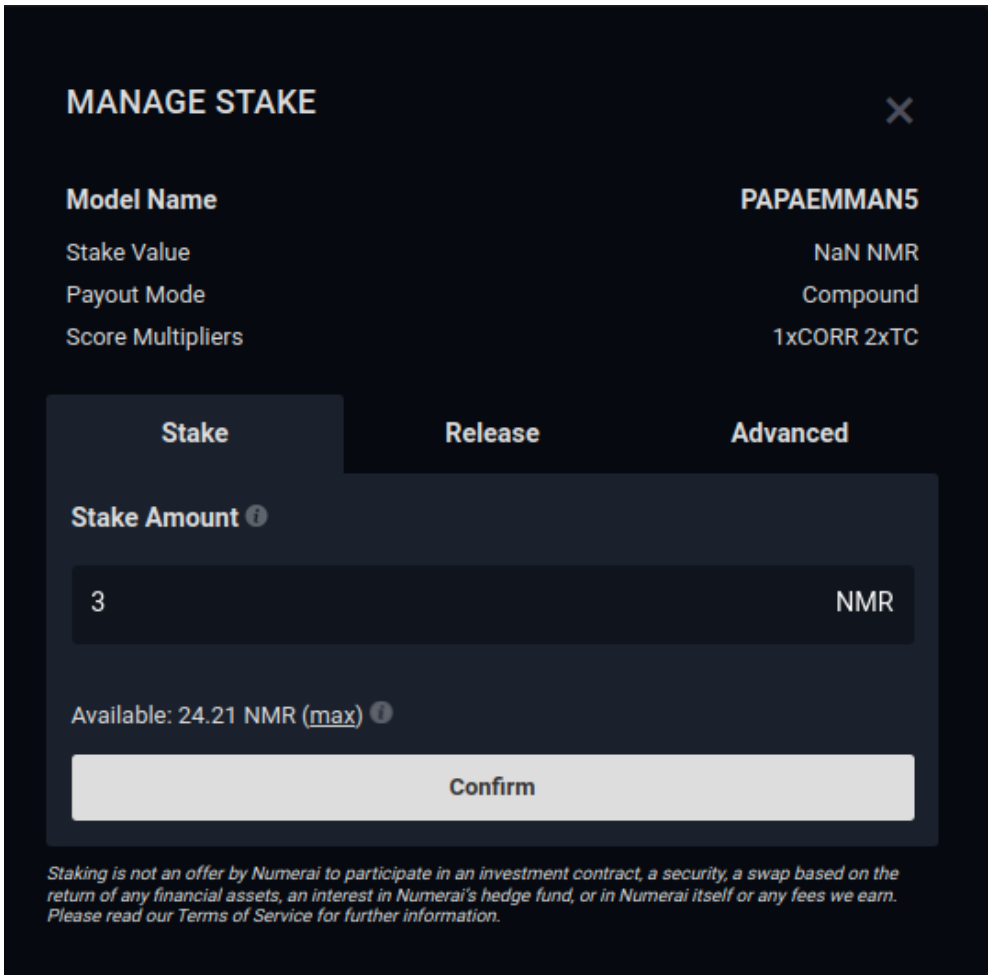


FIGURE 7.7: Web interface to manage stake

the current round and ultimately applied 4 weeks later (from the first scoring day) after the round has resolved. If the stake release request is made after the current submission deadline, the release request is queued for an additional week and will become effective on the first scoring day of the next round and ultimately be released 4 weeks later.

The 4-5 week release delay is applied to all stake releases and does not consider the last active round for a model.

***Numerai Disclaimer**: Staking is not an offer by Numerai to participate in an investment contract, security, a swap based on the return of any financial assets, an interest in Numerai's hedge fund, or in Numerai itself, or any fees we earn. Please read our Terms of Service for further information.*

7.2.3 Payouts

Payouts are a function of stake value and scores. The higher the stake value and the higher the scores, the more a participant will earn. If a

model has a negative score, then a portion of the corresponding stake will be burned. Payouts are limited to ±25% of the stake value per round.

The exact formula for payout calculation is

$$payout = stake_value * payout_factor * (corr * corr_multiplier + TC * tc_multiplier)$$

Before introducing the TC, Numerai used the Meta-Model contribution metric to calculate payouts, with the following formula

$$payout = stake_value * payout_factor * (corr * corr_multiplier + MMC * mmc_multiplier)$$

Currently, Numerai supports **9 different types of stake**:

1. Correlation (CORR)
2. Correlation + Half Total Contribution (CORR + 1/2 TC)
3. Correlation + Total Contribution (CORR + TC)
4. Correlation + Double Total Contribution (CORR + 2*TC)
5. Correlation + Triple Total Contribution (CORR + 3*TC)
6. Half Total Contribution (1/2 TC)
7. Total Contribution (TC)
8. Double Total Contribution (2*TC)
9. Triple Total Contribution (3*TC)

Users can use the web interface to manage their stake for each different model, as shown in Fig. 7.9.

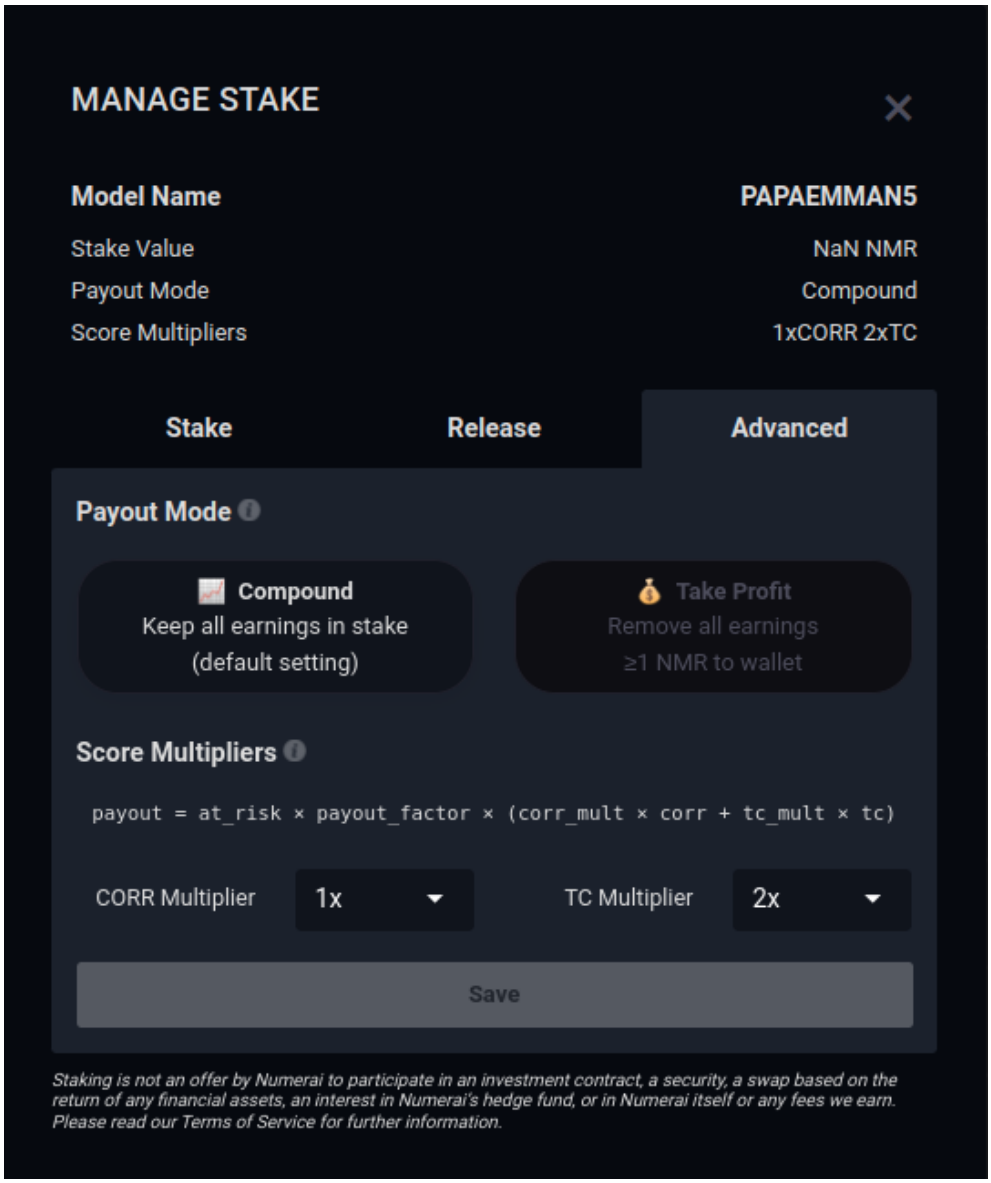


FIGURE 7.8: Staking options

Participants can also choose whether they want their payouts to **compound** into their stake or get paid out to their wallets after each round. Selecting the Compound Payout Mode (default setting), payouts from previous rounds are merged into the current stake value. The following diagram explains how the compounding staking works.

When a user starts staking, the stake value will remain constant for the first 4 rounds. Afterward, the stake value will update every week based on the payout of the round 4 weeks ago. For example, if the stake value on round N is 100 NMR and the payout is +5 NMR, this amount will be applied to the stake value in round N+4.

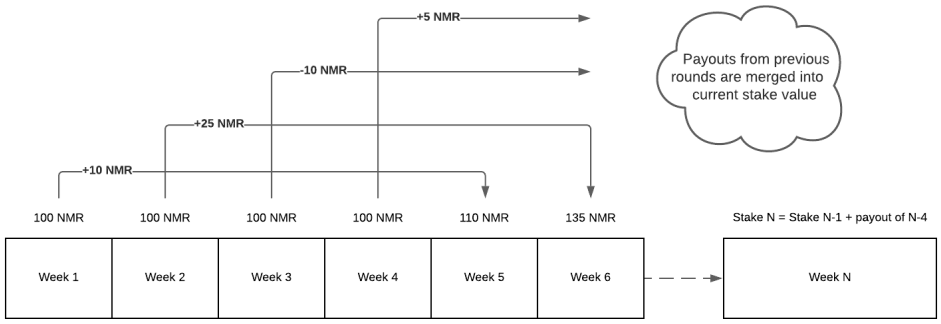


FIGURE 7.9: Compund staking

The ***stake value*** is the total value of NMRs staked on the model at Monday's round submission deadline at 14:30 UTC.

The ***payout factor*** is a number that scales with the total NMR staked across all models by all users in the tournament. The higher the total NMR staked, the lower the payout factor. Currently (May 2022), there are around 785k NMR staked in the Numerai tournament, with a payout factor around 0.45.

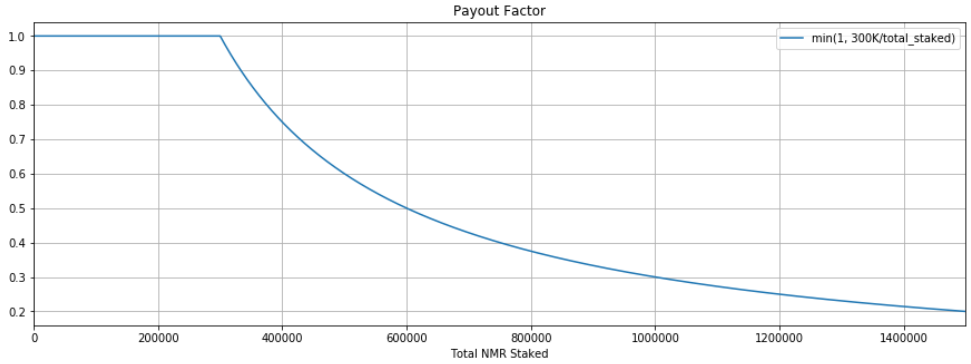


FIGURE 7.10: Payout factor

Numerai Disclaimer: *The payout factor curve and available multiplier options may and will be updated by Numerai in the future alongside major tournament releases.*

The following tables present **example payout calculations** for 2 different scenarios. Table 7.1 demonstrates the impact of adjusting the score multipliers for the same metrics. Table 7.2 demonstrates how a negative score in one of the 2 metrics (CORR, TC) may or may not cause a burn, depending on multipliers, and that the payout is capped at ±25% of the stake value.

TABLE 7.1: Payouts Calculations for positive scores (CORR TC)

Stake value (at risk)	Payout factor	Corr	Corr multiplier	TC	TC Multiplier	Round Payout
100 NMR	0.5	0.02	1.0×	0.04	0.0×	1 NMR (**1%** ROI)
100 NMR	0.5	0.02	1.0×	0.04	0.5×	2 NMR (**2%** ROI)
100 NMR	0.5	0.02	1.0×	0.04	1.0×	3 NMR (**3%** ROI)
100 NMR	0.5	0.02	1.0×	0.04	2.0×	5 NMR (**5%** ROI)
100 NMR	0.5	0.02	0.0×	0.04	0.5×	1 NMR (**1%** ROI)
100 NMR	0.5	0.02	0.0×	0.04	1.0×	2 NMR (**2%** ROI)
100 NMR	0.5	0.02	0.0×	0.04	2.0×	4 NMR (**4%** ROI)

TABLE 7.2: Payouts Calculations for negative scores (CORR or TC) and capped payouts

Stake value (at risk)	Payout factor	Corr	Corr multiplier	TC	TC Multiplier	Round Payout
100 NMR	0.5	-0.03	1.0×	-0.02	2.0×	-3.5 NMR (**-3.5%** ROI)
100 NMR	0.5	-0.03	1.0×	0.005	2.5×	-1 NMR (**-1%** ROI)
100 NMR	0.5	-0.03	0.0×	0.005	2.0×	1 NMR (**1%** ROI)
100 NMR	0.5	0.02	1.0×	-0.02	0.5×	0.5 NMR (**0.5%** ROI)
100 NMR	0.5	0.12	1.0×	0.19	2.0×	25 NMR (**25%** ROI)
100 NMR	0.5	0.13	1.0×	0.20	2.0×	25 NMR (**25%** ROI)
100 NMR	0.5	-0.13	1.0×	-0.20	2.0×	-25 NMR (**-25%** ROI)

Compounding profits

Stake value will grow as long as a model continues to have positive scores. Fig. 7.11 illustrates some example payout projections assuming that the model gets the same positive scores every week for 52 weeks, with compounding profits.

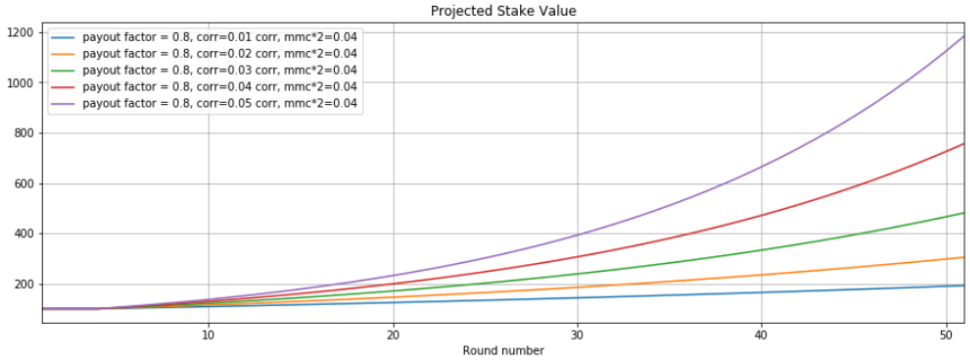


FIGURE 7.11: Stake Value growth

7.2.4 Leaderboard and Medals

All models' stake performances are **public**. Participants can review other models on the Numerai Tournament Leaderboard (Fig. 7.12) to understand how stakes can perform. The leaderboard can be sorted by the reputation of the models' **CORR, MMC, FNC, FNCV3, TC**.

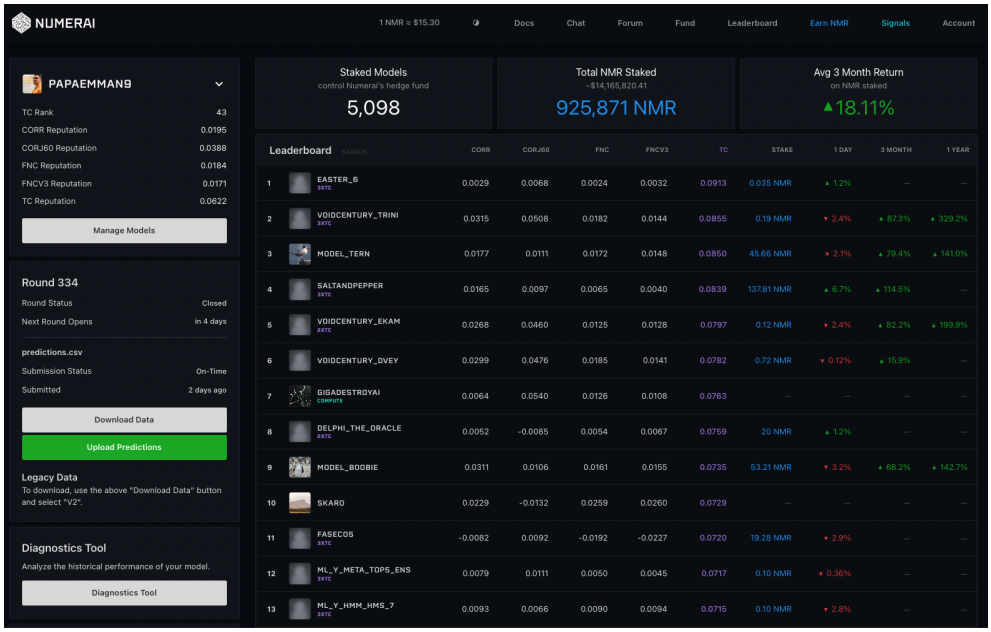


FIGURE 7.12: Numerai Tournament Leaderboard

Reputation is the weighted average of a given metric over the past 20 rounds, including rounds that are currently resolving. The weights on each round change as the round progresses. New rounds start low and gain weight as it nears resolution. Resolved rounds have full weight, but lose weight as it ages until it is dropped entirely. The specific weights for each round shown in Table **??**.

- Current round: N
- Resolving rounds: $N-1, N-2, N-3$
- Resolved Round: $< N-3$

Round	Weight
N-20	0
N-19	0.10
N-18	0.35
N-17	0.60
N-16	0.85
N-15	1
N-14	1
N-13	1
N-12	1
N-11	1
N-10	1
N-9	1
N-8	1
N-7	1
N-6	1
N-5	1
N-4	1
N-3	0.90
N-2	0.65
N-1	0.40
N	0.15

Medals are awarded for scores at the end of each round, based on the 3 main competition metrics; CORR (Correlation), TC (Total Contribution), and MMC (Meta-model contribution). Note that all submitted models are eligible to receive a medal, regardless of whether they have been staked or not.

Currently, the medals are rewarded as follows;

1. The top 1% of the models per metric get a gold medal.
2. The top 5% of the models per metric get a gold medal.
3. The top 15% of the models per metric get a gold medal.

Fig. 7.13 illustrates the round summary for a model that earned a gold medal for CORR because it had a greater correlation than the 99% of the models (top 1%) and a bronze medal for MMC because it had a greater MMC than the 87% of the models (top 15%), for the specific round.

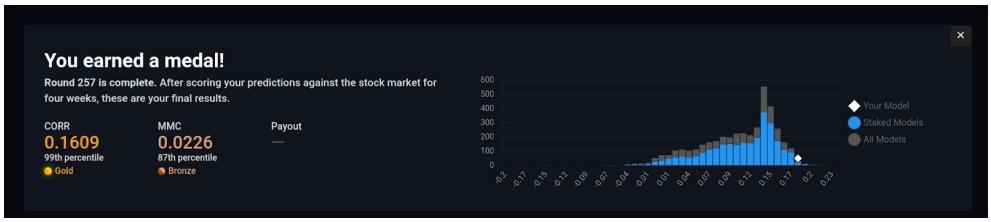


FIGURE 7.13: Round summary - Medals

Medals are also visible on the model's profile page. Fig. 5.14 shows a the model which has won;

1. 0 gold medals
2. 2 silver medals (1xCorr, 1xMMC)
3. 9 bronze medals (2xCorr, 4xMMC, 3xTC)

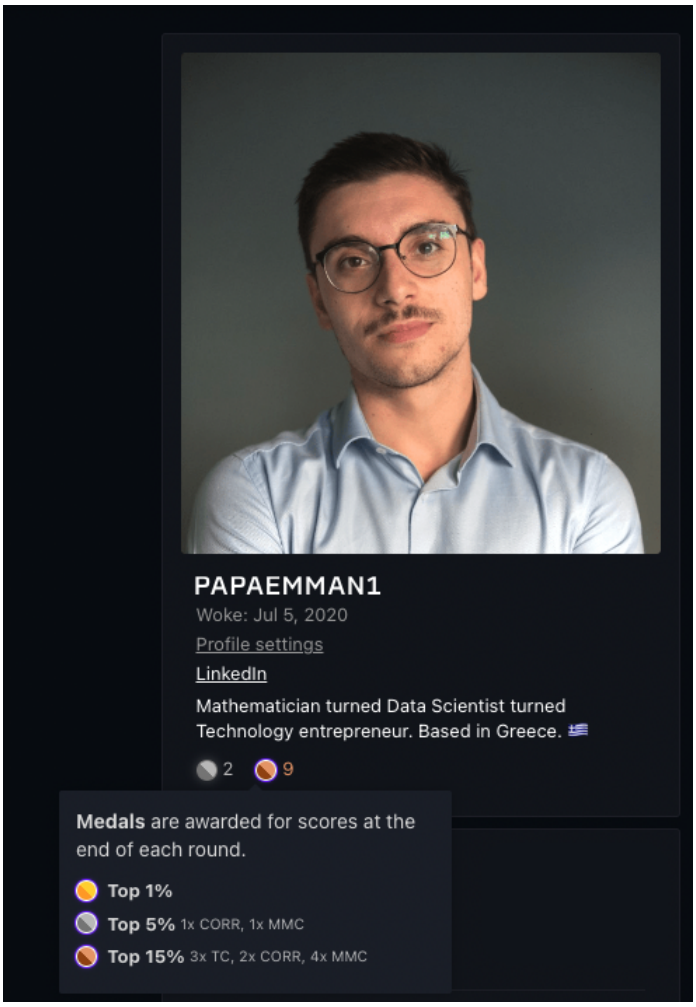


FIGURE 7.14: Model medals [PAPAEMMAN1]

8 Competing in Numerai tournament; Machine Learning Modeling

After establishing a background in Numerai Hedge Fund, NMR Cryptocurrency token, and Numerai tournament and discussing the Numerai dataset, the evaluation metrics, and the applied machine learning process in previous Chapters, it is time to delve into the actual algorithms and different methodologies employed to compete in the Numerai tournament.

Due to the competitive nature of the Numerai tournament, we decided not to discuss the specific ML pipeline utilized for our production models. Instead, we will focus on standard techniques, algorithms, and methodologies employed and shared publicly by Numerai community members.

The main goal of this chapter is to provide a solid guide for interesting readers with a background in machine learning to start competing in the Numerai tournament. This chapter, along with other valuable resources such as ***Numerai example scripts***[1], ***Numerai forum***[2] and ***Numerai rocket chat***[3], is sufficient to help readers develop their first ML model for the Numerai tournament.

8.1 Machine Learning problem formulation

From a high-level point of view, Numerai competitors, develop AI models to predict the stock market. As described in Chapter 5, the Numerai dataset consists of cleaned, normalized, and obfuscated data; therefore, ML models trained on this dataset aren't useful outside the Numerai tournament

Also, because the dataset is anonymized, there is no known connection with the real-world Stock Market. This means that participants cannot apply their knowledge about financial markets or incorporate outside

[1] https://github.com/numerai/example-scripts
[2] https://forum.numer.ai/
[3] https://rocketchat.numer.ai/

information into the dataset. Therefore, strategies for dealing with this problem are limited to solving it as a **black-box optimization problem** and using advanced computational techniques to optimize the given evaluation metrics.

As we described in Chapter 5, every feature takes 5 different values [0.00, 0.25, 0.50, 0.75, 1.00]. The same applies to the current target (***target_nomi_20***), too, as it takes 5 different values [0.00, 0.25, 0.50, 0.75, 1.00] as well (Fig. 8.1). Numerai tournament's evaluation is based on **Spearman's rank correlation coefficient (Corr)** and **Total Contribution (TC)**. This means that predictions are scored solely based on their ordering.

era	feature1	...	feature310	target
era1	0.75	...	0.00	0.25
era1	1.00	...	0.25	0.75
era1	0.25	...	1.00	0.00
era1	0.25	...	0.00	1.00
era1	0.75	...	0.25	0.25
era1	0.00	...	0.75	1.00

FIGURE 8.1: The Numerai Dataset (V2)

As specific target values exist in the given dataset, this problem should be addressed as a **Supervised Learning (SL)** task. In supervised machine learning, the model learns a function that maps an input called feature vector to an output called target, based on example input-output pairs.

Given the specific data and evaluation metrics, there are 3 suitable machine learning modeling techniques:

1. Modeling the problem as **Regression** task.
2. Modeling the problem as a **Multi-class classification** task.
3. Modeling the problem as a **Ordinal Regression / Ordinal Classification** task.

8.2 Models evaluation: Regression vs. Classification

Many different types of models were trained and evaluated during the experimentation stage. We used Linear Regression, k-Nearest Neighbors, Random Forest, and Gradient Boosting Machine algorithms for the **regression** approach.

Regarding the **multiclass classification**, 3 different algorithms offered by the Scikit-Learn (*sklearn.multiclass*[4]) framework were evaluated;

1. **OneVsRestClassifier**
2. **OneVSOneClassifier**
3. **OutputCodeClassifier**

Apart from single Regression and Classification models, more modeling techniques were tried, such as **ensembling** and **stacking**.

Experimentation results showed that **Regression models are more suitable for this problem.**

The main reason is that we cannot control how the multiway ties will be broken in classification models. The problem requires a relative ordering across all equities (rows in the dataset), which classification methods don't offer for equities predicted to belong in the same. If we provide a submission file with ties, the Numerai will resolve them according to their order in the text file, which will lead to suboptimal results.

Additionally, when approaching the model as a multi-label classification task, the model cannot incorporate intuition about the relative ranking between different classes. Targets values are [0.00, 0.25, 0.50, 0.75, 1.00], denoting that equities with a target value equal to 0 have the lowest performance for this particular era. In contrast, equities with a target value of 1.00 have greater performance among all assets. Relative ranking between target values can't be represented out-of-the-box in multi-label classification models, so these models tend to have inferior performance to regression models.

On the other hand, regression models offer a predefined ranking as they output real values. Fig. 8.2 shows the distribution of predictions for 3 different regression models. The average value of predictions is 0.5, which is equal to the target's average value. But, each distribution has different properties (standard deviation, long or short tails, etc.) depending on the internal mechanics of the algorithm used.

[4] https://scikit-learn.org/stable/modules/classes.html#module-sklearn.multiclass

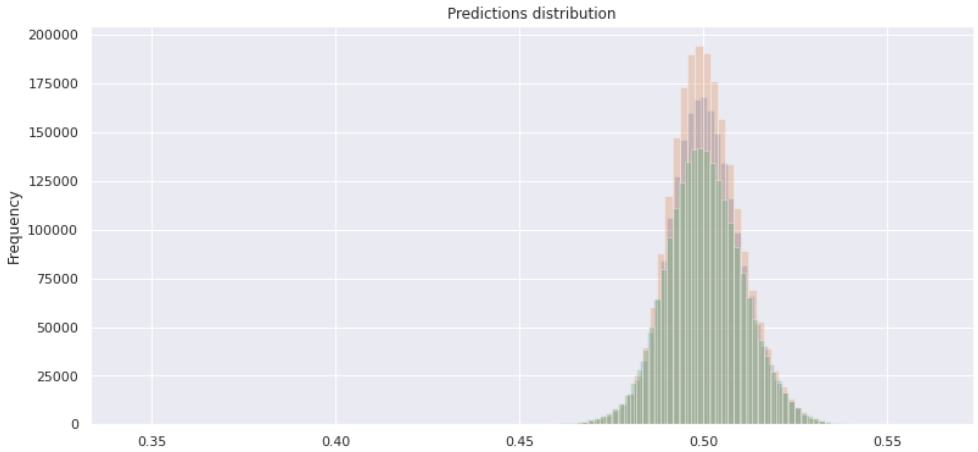


FIGURE 8.2: ML models predictions distributions

8.3 Feature Engineering

Data Science is an interdisciplinary field as shown in the Fig. 8.3. Best Data Scientists stand out because they can combine their technical expertise and scientific background with their business and domain knowledge.

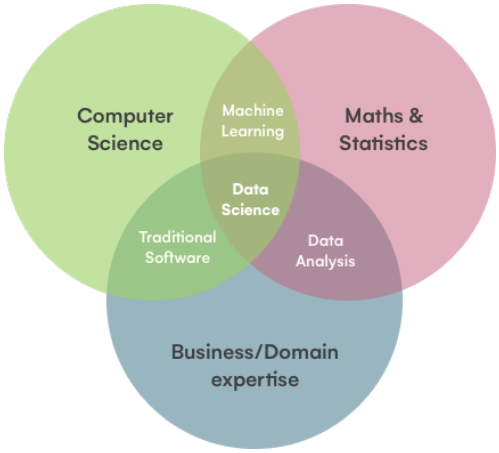


FIGURE 8.3: Data Science field

This combination of domain knowledge and technical expertise is materialized in the form of **feature engineering**. Feature engineering is the process of creating and transforming variables when developing a predictive model using machine learning. It's a great way to enhance predictive models as it involves isolating key information and incorporating it into the dataset.

Due to the fact that the Numerai tournament's dataset is anonymized and normalized (Fig. 8.4), participants don't know what each feature represents. Therefore, applying any specific financial knowledge or

intuition to create additional features is impossible. Thus, only simple computational approaches are available for new feature creation.

```
numerai_training_data.head()
```

id	era	data_type	feature_honoured_observational_balaamite	feature_polaroid_vadose_quinze	feature_untidy_withdrawn_bargeman
n003bba8a98662e4	0001	train	1.0	0.50	1.00
n003bee128c2fcfc	0001	train	0.5	1.00	0.25
n0048ac83aff7194	0001	train	0.5	0.25	0.75
n00691bec80d3e02	0001	train	1.0	0.50	0.50
n00b8720a2fdc4f2	0001	train	1.0	0.75	1.00

FIGURE 8.4: Numerai Dataset (screenshot)

For new features creation, we used standard mathematical methods such as

- rolling statistics of features across columns
- lag values of features across columns
- distribution of values per row
- sum of rows per row

and more.

8.4 Feature exposure and model performance

The feature exposure metric has changed several times throughout the history of the Numerai tournament. Numerai moved from Pearson to Spearman's rank correlation coefficient (the same metric used for CORR). And instead of aggregating individual feature exposures with standard deviation, it's now using root mean square (RMS) as the aggregation function. Along with the RMS aggregation function, the max feature exposure is also used to access a model's risk.

There's a trade-off between feature exposure and the model's performance (in terms of Spearman's correlation). It's possible to train models with incredibly low max feature exposure, which aren't very useful in practice. Generally, models with very low max feature exposure tend to have low correlation. On the other hand, models with high max feature exposure will likely have a higher correlation but are also more likely to degrade in future eras.

Feature exposure is a great heuristic to compare models with similar in-sample evaluation metrics (Sharpe ratio and Spearman's correlation). There is an inverse correlation between max feature exposure and out-of-sample performance. Experimentation showed that a baseline ML model would have a max feature exposure of 0.30, which

means that the correlation of the target with a particular feature will be at most 0.30.

8.4.1 Reducing Feature Exposure with Regularization

Applying regularization to machine learning models efficiently reduces feature exposure while maintaining the model's performance. Machine learning frameworks have hyperparameters to adjust for regularization.

```
model = XGBRegressor(max_depth=5, learning_rate=0.01,
        n_estimators=2000, n_jobs=-1, colsample_bytree=0.1,
        lambda=0.1, alpha=0.1)
```

8.4.2 Reducing Feature Exposure with Neutralization

Yet another way to reduce feature exposures is to use feature neutralization. Feature neutralization consists in subtracting from a model's predictions the linear relation between one of the features and the target.

The intuition behind this process is as follows: neutralization of a prediction for risky features is the first-order approximation of the operation that removes the risky feature's component alone, leaving only the interactions with other features. Neutralizing the predictions with respect to the features reduces both feature exposure and max feature exposure.

The following code snippet outlines the neutralization process.

```
import numpy as np
def neutralize(df, columns, by, proportion=1.0):
    scores = df[columns]
    exposures = df[by].values
    scores = scores - proportion *
          exposures.dot(np.linalg.pinv(exposures).dot(scores))
    return scores / scores.std(ddof=0)
```

Description:

1. Feature neutralization begins by taking the entire dataframe, the column to neutralize, the features to neutralize by, and the neutralization proportion as ***inputs***.
2. The first and second lines isolate the column to neutralize (scores) and the features to neutralize (exposures), respectively.
3. For the third line, the code reduces the neutralization column by a vector multiplied by the proportion specified. The vector being

used is computed by first taking the dot product of the pseudo inverse of the exposures with the scores and second taking a dot product of the resultant and the exposures.

For a more detailed explanation, interested readers can refer to this article[5]

8.5 Evaluation framework

A core characteristic of the Numerai dataset is the presence of the time factor in it. The *"era"* column represents a specific week; thus, the dataset can be considered a time-based dataset. But as we have already described in Chapter 5, participants cannot perform any time series analysis because there isn't any connection between equities across different eras due to the modification of ids. Moreover, as described in the same chapter, the current target (*target_nomi_20*) it's created based on a 4 weeks (20 trading days) window in the future. This means that different targets were created from overlapping periods.

Considering the unique Numerai dataset characteristics when designing an evaluation framework for the Numerai tournament is essential.

Hold-out evaluation framework

The most straightforward evaluation framework is the hold-out strategy. This method uses the Numerai training dataset (574 eras) as the training set and the Numerai validation dataset (~450 eras) as the validation dataset. Out-of-the-box Numerai doesn't offer a testing dataset other than the weekly live dataset. Not having a testing dataset for the final evaluation is extremely dangerous as it can lead to an overfitted model with excellent performance in the validation data but low performance in the live data. A sensible solution is to keep a subset of the validation data of at least 50 eras as the test set before any modeling steps.

training set	Numerai training set (574 eras)
validation set	Subset of the Numerai validation set (400 eras)
test set	Subset of the Numerai validation set (~50 eras)

TABLE 8.1: Hold-out split

Cross-validation evaluation framework

[5] https://medium.com/analytics-vidhya/our-experience-with-numerai-2b0777acc12e

A more sophisticated cross-validation approach is better than a simple hold-out evaluation as it can more accurately assess the generalization error of an ML model. However, establishing a cross-validation framework for the Numerai tournament isn't a trivial task.

As we described in Chapter 3, selecting an inappropriate cross-validation framework can lead to over-optimistic results, thus biasing the final selection of the model. For example, for the numerai dataset, a random cross-validation will look much better for a particular setting than cross-validation by epoch. Even for a simple linear model, random KFold shuffle reports a correlation of ~5.4%, but a time series split reports a lower score of ~2.9%. The KFold and GroupKFold cross-validation methods have comparable correlations of ~4.2% because they use similar folds (Table 8.3). KFold divides the dataset into 5 subsets with an equal number of rows, preserving the order of rows in the original dataset. GroupKFold divides the dataset into 5 subsets, ensuring that the same group is not represented in both the test sets while preserving the row order of the original data set. Thus the differences in each fold are limited to a few rows at the last era of each fold.

Cross-validation method	**Correlation**
KFold(n_splits=5, random_state=None, shuffle=True)	0.054
KFold(n_splits=5, random_state=None, shuffle=False)	0.043
GroupKFold(n_splits=5)	0.041
TimeSeriesSplitGroups(n_splits=5)	0.029

TABLE 8.2: Correlation for different cross-validation methods for the same model

It is evident that the appropriate cross-validation framework for the Numerai tournament is the **era-wise time-series cross-validation**. This means we need to use TimeSeriesSplit cross-validation with respect to the eras. All the other frameworks will lead to data leakage.

An example of an era-wise time-series cross-validation scheme might look like this;

1. Combine the Numerai training dataset (574 eras) and the Numerai validation dataset (~451 eras) into a single dataset (~1025 eras)
2. Keep the last ~25 eras for the test set
3. Use the remaining 1000 eras for a 4-fold era-wise time series cross-validation, as shown in Table.

This setting ensures that every era from 201-1000 is used precisely once for an evaluation, participating equally in the estimation of the generalization error calculations.

Fold	Train set eras	Validation set eras
1	1-200	201-400
2	1-400	401-600
3	1-600	601-800
4	1-800	801-1000

TABLE 8.3: Correlation for different cross-validation methods for the same model

8.6 Improving metrics

There are many suitable techniques to improve evaluation metrics. But it's important to keep in mind that all metrics are connected, and by fixing one metric, we can do worse on other metrics. Although the best models will perform well on all metrics.

The 5 main ways to improve risk metrics are

1. Feature Selection
2. Eras selection (hard vs. easy eras)
3. Regularization and Feature Normalization
4. Model Ensembling
5. Hyperparameter optimization
6. Retraining

8.6.1 Feature Selection

Regarding the Numerai dataset, all feature selection methods mentioned in Chapter 7 are applicable. For example, the **correlation method** can be used to select features that are predictive on their own, i.e., they have a high correlation with the target and to eliminate other features with zero correlation with the target. An issue with this approach is that single features do not have a constant correlation across eras, as explained in Chapter 5. Therefore special treatment is needed to select features that work consistently.

Another example of feature selection in the numerai dataset is using the **Random Forest (RF) feature importances** (Fig. 8.5). Training an

RF model and selecting the features with the highest feature importance can significantly reduce training times and improve evaluation metrics.

	feature_name	feature_importance
1059	feature_crispy_neighboring_jeffersonian	46
796	feature_acclimatisable_unfeigned_maghreb	42
838	feature_censorial_leachier_rickshaw	39
1021	feature_mancunian_stalky_charmeuse	36
645	feature_asteriated_invigorated_penitence	36
...	...	...
1157	feature_ironclad_coppery_labour	0
1086	feature_unholy_residential_anabaptism	0
1085	feature_pardonable_ungraceful_bedazzlement	0
182	feature_lamarckian_tarnal_egestion	0
1089	feature_predominant_unmown_concealing	0

FIGURE 8.5: Random Forest feature importances

8.6.2 Eras selection (hard vs. easy eras)

Eras in the Numerai dataset are homogenous but different from each other. This means that eras can be more or less applicable to other eras.

We confirmed this by splitting the eras into blocks of 15, training on each block, and evaluating based on the other data blocks. A model trained on a block doesn't have consistent performance across all the other blocks, which is a proxy for the similarity of the different era blocks.

To exploit this finding, we used a baseline model to divide the training data into two sets:

- **Easy eras**: eras with a high correlation between targets and baseline model's predictions
- **Hard eras**: eras with a high correlation between targets and baseline model's predictions

Having an algorithm to separate between "easy" and "hard" eras, enabled us to use different modeling approaches for different types of eras.

8.6.3 Regularization and Feature neutralization

Applying regularization and feature neutralization, as described in section 8.4 can increase the performance of a model and make it more stable through the eras, as it will reduce the feature exposure.

8.6.4 Model Ensembling

Combining different models is an excellent way to improve performance. ML practitioners can combine models with low correlated predictions to form more accurate models incorporating information from multiple weaker models.

A crucial observation is that the distribution of features for target values of 0, 1 is noticeably different from the respective metric for target values of 0.25, 0.75. This makes training a classifier one-versus-rest behave counterintuitively. Specifically, a multinomial classifier seems to learn how to pick out extreme targets, and their predictions are the most correlated. Hence, combining regression and multi-label classification models is a great recipe for improving performance.

8.6.5 Hyperparameter tuning

ML models' performance is sensitive to the choice of hyperparameters. Therefore, the hyperparameter optimization process is vital.

We used the Bayesian optimization method implemented in the hyperopt python package for the hyperparameter tuning.[6]

Regarding the **LightGBM** model, the following hyperparameters were tuned using the cross-validation approach described in section.

Note: Regarding the *n_estimators* hyperparameter, when doing experiments, we observed that better results are obtained when early stopping is used instead of a fixed number of estimators.

8.6.6 Model Retraining

As the training data does not change over time, re-training models aren't necessary. Users who also use the validation dataset to train the production models can re-train their models as soon as a sufficient

[6] https://gist.github.com/papaemman/70f363d54f57a2dfced84b7c7bdb6220

Hyperparameter	Description
num_leaves	Maximum tree leaves for base learners
max_depth	Maximum tree depth for base learners
learning_rate	Boosting learning rate
n_estimators	Number of boosted trees to fit
reg_alpha	L1 regularization term on weights
reg_lambda	L2 regularization term on weights
subsample	Subsample ratio of the training instance
subsample_freq	Frequency of subsample
colsample_bytree	Subsample ratio of columns when constructing each tree.

TABLE 8.4: LightGBM hyperparameters

amount of new data has been added. A rule of thumb is to wait for at least 8 new eras to be added (2 months).

8.6.7 More ideas on how to improve performance and reduce risk

It is essential to remember that all metrics are connected, and changing one metric will also change others. Therefore, the goal is to find a model that will balance risk and profit metrics well. Mixing different ideas and looking for original solutions is the best approach for the Numerai tournament.

For example, a production pipeline can combine several steps as follow;

1. Remove 10% of features (the least important)
2. Remove 10 of the easiest eras
3. Apply neutralization at level 50%
4. Run hyperparameters search for optimal *learning_rate, max_depth, colsamples_bytree*.

9 Results and Conclusions

"So, you are good at Machine Learning, huh? Do you wanna bet?"

—Me to myself

9.1 Production Models evaluation and results

As explained in Chapter 7, the current payout mechanism is based on 2 metrics; correlation and total contribution. However, as participants don't have a sufficient algorithm to proxy the calculations for total contribution, we relied solely on the correlation score to select our production models.

During the machine learning experimentation process, we got divergent results. Even though cross-validation returned a relative raking between the different machine learning pipelines, a more careful investigation revealed that there isn't a single best model. Different models have different results in various market regimes. Of course, the Numerai quant team is aware of this observation; that's why they allow up to 100 different models to be submitted per participant.

Considering this, we decided to deploy 24 different models[1] to production and inspect their performance in the live data across multiple weeks.

After evaluating the models for 20 consecutively rounds, all production models had a positive correlation (Fig. 9.1) and total contribution (Fig. 9.2), meaning they all yielded a positive return. This is a successful result highlighting the quality of the models.

The best performing models regarding each evaluation metric were the ***PAPAEMMAN17***[2] and ***PAPAEMMAN9***[3].

[1] https://numer.ai/models
[2] https://numer.ai/papaemman17
[3] https://numer.ai/papaemman9

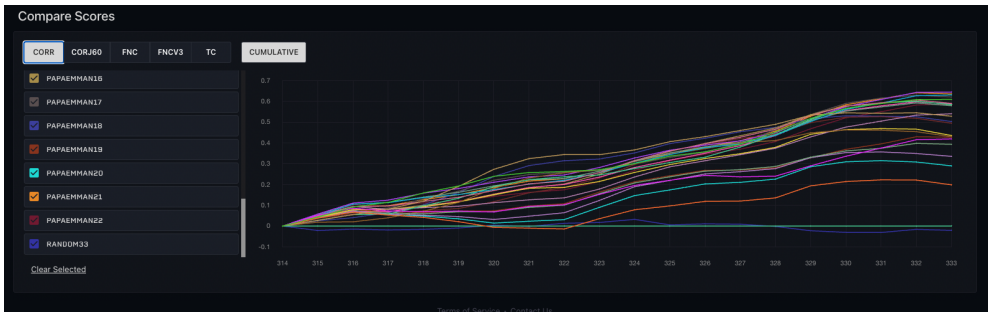


FIGURE 9.1: Models' correlation score (cumulative)

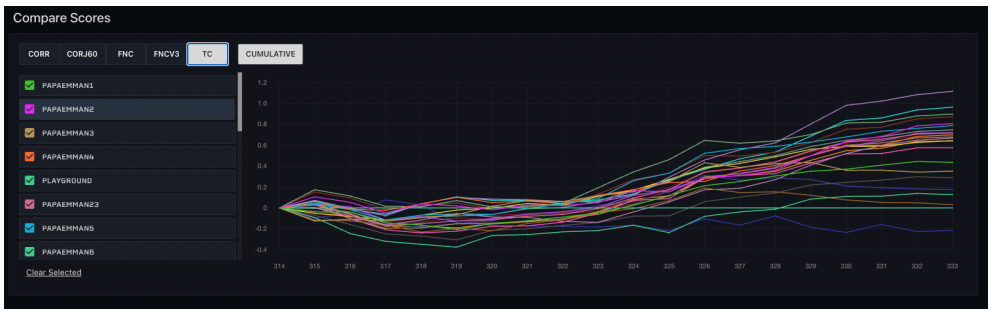


FIGURE 9.2: Models' total contribution score (cumulative)

- **PAPAEMMAN17** model had the best correlation reputation among all submitted models, of ~0.0351 (3.5% week-over-week), placing it in the 1292 position on the leaderboard.
- **PAPAEMMAN9** model had the best total contribution reputation, among all submitted models, of ~0.062161 (6.2% ROI week-over-week), placing it in the 43 positions on the leaderboard. (Fig. 9.3)

Considering that almost 15.000 unique models are submitted each round, these results place the PAPAEMMAN17 model in the **top 8%** and the PAPAEMMAN9 model in the **top 0.3%** of the Numerai tournament.

Table 9.1 shows the correlation reputation, correlation rank, total contribution reputation, and total contribution rank for every model. Also, we present each model's stake type and total stake amount along with these metrics.

Along with normal ML models (prefix *PAPAEMMAN*), we decided to keep submitting a random model called *RANDOM33*. We are doing this to compare the performance of the random and trained ML models.

Stock market prediction is a tough problem mainly because the signal-to-noise ratio of the datasets is low. Therefore, it is not rare for random

Model ID	Corr Reputation	Corr Rank	TC Reputation	TC Rank	Stake Type	Stake Amount (NMR)
PAPAEMMAN1	0.033	1756	0.027	1198	1×CORR	5.6
PAPAEMMAN2	0.02	4948	0.04	390	-	-
PAPAEMMAN3	0.031	2083	0.027	1187	1×CORR 0.5×TC	2.5
PAPAEMMAN4	0.011	6276	0.038	496	-	-
PAPAEMMAN5	0.033	1773	0.045239	243	1×CORR 2×TC	8.2
PAPAEMMAN6	0.034	1515	0.014506	2727	1×CORR	3
PAPAEMMAN7	0.021	4776	0.04686	206	1×CORR	1.35
PAPAEMMAN8	0.027	3279	0.008871	3792	1×CORR	1
PAPAEMMAN9	0.019	5071	0.062161	43	2×TC	4
PAPAEMMAN10	0.028	3105	0.037171	534	1×CORR 0.5TC	1
PAPAEMMAN11	0.033	1590	0.039574	426	1×CORR 1×TC	9.36
PAPAEMMAN12	0.026	3617	0.040305	396	1×CORR 0.5×TC	4.09
PAPAEMMAN13	0.031	2076	0.043823	289	1×CORR 1×TC	6.05
PAPAEMMAN14	0.033	1787	0.035201	632	1×CORR 1×TC	6.06
PAPAEMMAN15	0.03	2512	0.042985	308	1×CORR 0.5×TC	4.04
PAPAEMMAN16	0.033	1623	0.041209	360	1×CORR 2×TC	8.06
PAPAEMMAN17	0.035	1292	0.021657	1764	1×CORR	6.07
PAPAEMMAN18	0.031	2256	0.017542	2282	1×CORR	2
PAPAEMMAN19	0.021	4612	0.044723	259	1×CORR 0.5×TC	2
PAPAEMMAN20	0.016	5525	0.053255	105	1×TC	3
PAPAEMMAN21	0.034	1398	0.037135	535	1×CORR 0.5×TC	23
PAPAEMMAN22	0.03	2368	0.042617	318	1×CORR 0.5×TC	3.96
PAPAEMMAN23	0.033	1719	0.043021	306	1×CORR 2×TC	10.78
RANDOM33	-0.001	7345	-0.01095	12802	-	-

TABLE 9.1: Production models

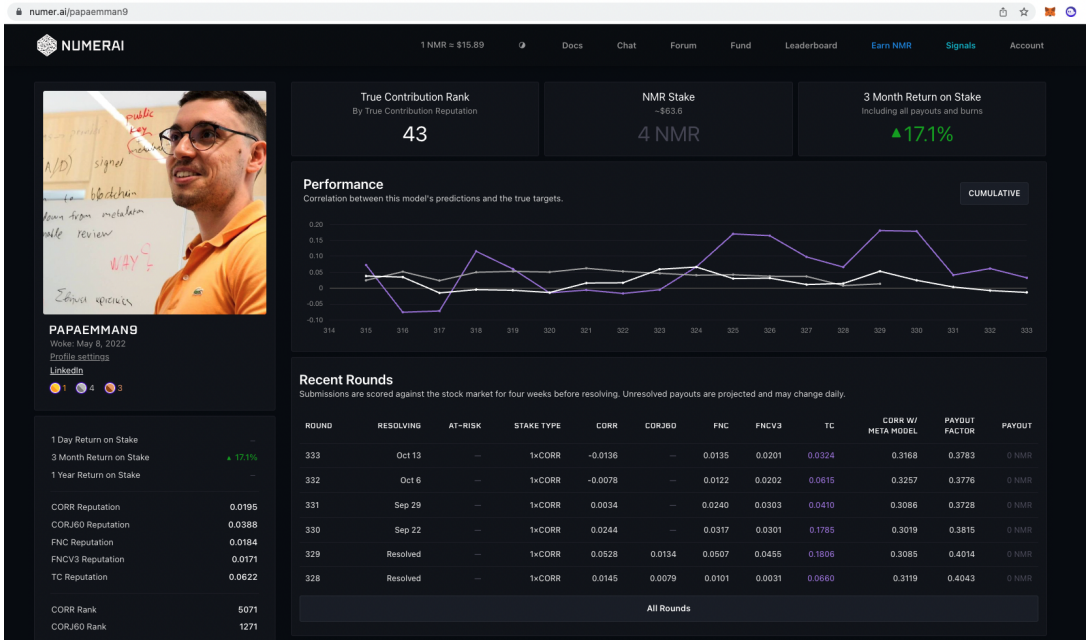


FIGURE 9.3: Models' performance [PAPAEMMAN9]

stock selection to perform better than carefully designed portfolios.[4]

For our case, all the models successfully outperformed the random model across a period of 15 weeks. However, there were multiple rounds in which ML models had inferior performance compared to the random model, revealing the problem's difficulty. Specifically, the *RANDOM33* model won 2 silver and 2 bronze medals, outperforming the vast majority of trained ML models submitted for these 4 rounds.

Considering the average performance of the random model across 15 completed rounds, it has a ~0 Correlation reputation and a ~0 Total Contribution, as expected.

Production models also won multiple **medals**, underlining the fact that for various rounds outperform the largest part of submitted models. Table 9.2 shows the total medals each model won for the same period. The random models got 2 silver and 2 bronze medals, which forms a reasonable comparison benchmark. Figure 9.5 provides some examples as an indication.

[4] https://www.forbes.com/sites/rickferri/2012/12/20/any-monkey-can-beat-the-market/?sh=705245a1630a

Model ID	Gold	Silver	Bronze
PAPAEMMAN1	0	2	9
PAPAEMMAN2	0	3	11
PAPAEMMAN3	0	1	8
PAPAEMMAN4	1	0	14
PAPAEMMAN5	0	2	1
PAPAEMMAN6	0	2	1
PAPAEMMAN7	1	2	3
PAPAEMMAN8	0	0	4
PAPAEMMAN9	1	4	3
PAPAEMMAN10	0	2	2
PAPAEMMAN11	0	1	3
PAPAEMMAN12	0	0	5
PAPAEMMAN13	0	0	5
PAPAEMMAN14	0	1	3
PAPAEMMAN15	0	1	4
PAPAEMMAN16	0	2	2
PAPAEMMAN17	0	0	4
PAPAEMMAN18	0	1	3
PAPAEMMAN19	0	3	3
PAPAEMMAN20	0	2	4
PAPAEMMAN21	0	1	4
PAPAEMMAN22	0	1	4
PAPAEMMAN23	0	1	2
RANDOM33	0	2	2

TABLE 9.2: Production models medals

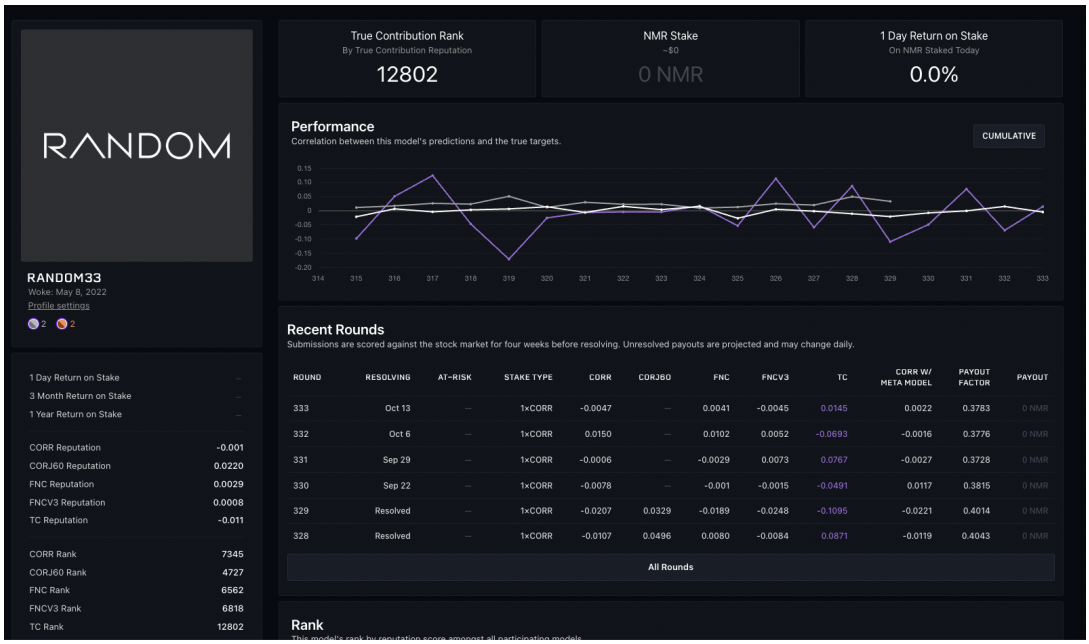


FIGURE 9.4: Random model performance

9.2 Portfolio optimization: Multipliers and Total stake value

Deciding each model's stake type and the amount isn't a trivial task. Numerai gives the agency to participants to determine the multiplier for correlation and total contribution metrics, as well as the total stake amount for each submitted model.

We used the following formula to define the ***correlation*** and ***total contribution*** multipliers.

We evaluated each model across 15 completed rounds. For models with 0, 1, or 2 rounds with a negative correlation score (burning rounds), we assign a **1X Corr** multiplier. For models with 3 or more burning rounds, we set a **0X Corr** multiplier.

It is noteworthy to highlight that 19 of the 24 total models were assigned a 1X corr multiplier. From that 19 models, 9 models have consistently positive correlation scores across the investigated period. This means that these 9 models always returned a positive return on stake, making them highly appealing investment opportunities.

For the TC, we used a similar formula, counting the number of rounds with a negative total contribution score and assigning an appropriate multiplier, as shown in Table 9.3.

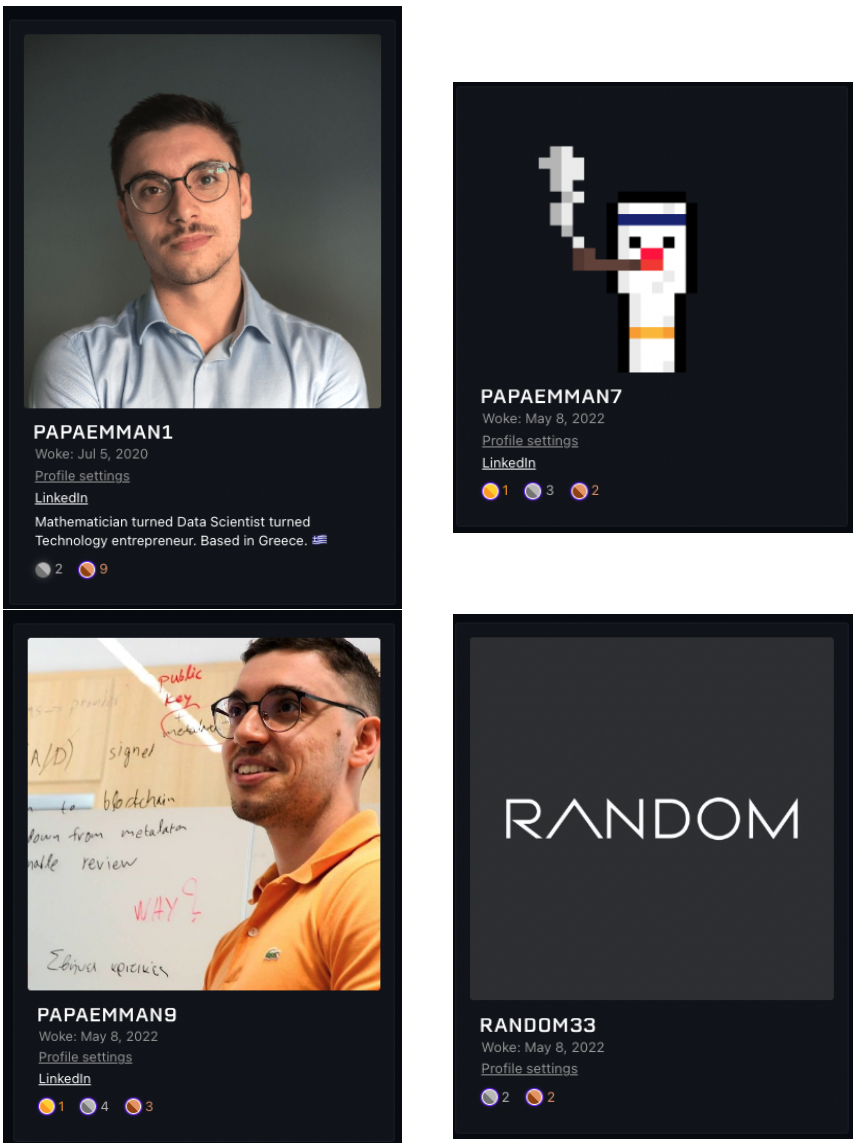


FIGURE 9.5: Production Models

TC Multiplier	Past rounds with negative TC (out of 15 rounds)	Total models
3X	0,1	0
2X	2	2
1X	3	4
0.5X	4	7
0	5+	11

TABLE 9.3: Total Contribution multipliers

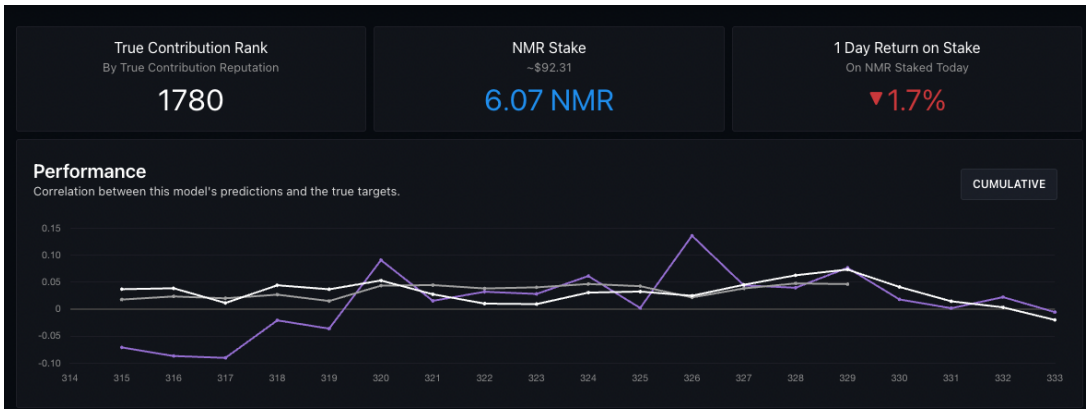


FIGURE 9.6: Model performance [PAPAEMMAN17]

Fig. 9.6 shows one of the models having positive correlation across all 15 completed rounds and negative total contribution at the first 5 completed rounds and positive afterward.

Regarding the **stake amount**, we used the following algorithm.

1. Calculate the relative ranking of each model for Correlation and Total Contribution (1-24)
2. Calculate the Sharpe ratio for correlation and total contribution
3. Find the total stake using the formula

$$StakeAmount = \frac{Corr_shapre * TC_sharpe}{Corr_relative_ranking + TC_relative_ranking} * 100$$

The intuition behind this formula is obvious. Models with high Sharpe ratio and low relative ranking will have the maximum stake. Also, all models will have a staking value proportioned to their Sharpe ratios and relative ranking.

The combination of multipliers and total amounts forms a **diverse portfolio of models** with **low risk** and **high expected returns**.

9.3 Return On Investment

Return on Investment (ROI) is a performance measure used to evaluate the profitability of an investment. ROI is often used as a comparison metric between different assets. It is designed to directly measure the return on a particular investment relative to its cost.

The formula to calculate ROI is:

$$ROI = \frac{FutureValueOfInvestment - CostOfInvestment}{CostOfInvestment} * 100$$

Overall, the performance of the models in Numerai Tournament is significantly good, gaining 0.1%-3% ROI each week. After evaluating the

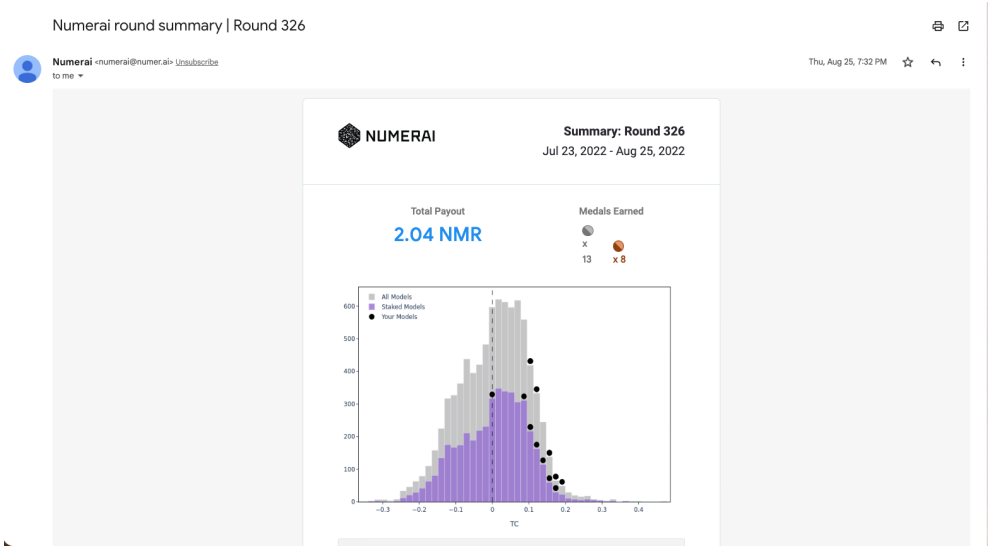


FIGURE 9.7: Example round summary

portfolio for 20 consecutively eras, only 1 era had a negative (burning) result. For most of the rounds, the models outperformed the rest of the submitted models and earned a significant payout.

As of September 2022, the 3 Month ROI is 57.76%.

9.4 Conclusions

Numerai tournament is an innovative challenge which combines AI / Data Science / Machine Learning, Cryptocurrencies, and Stock Market investing. The goal of this thesis was to study the Numerai Tournament and to provide relevant information on every aspect of it, as well as to develop great performant Machine Learning pipelines for the Numerai tournament.

Considering the breadth and depth of the present thesis, together with the evaluation results of the submitted models, the thesis is successful in achieving both goals to the maximum extent.

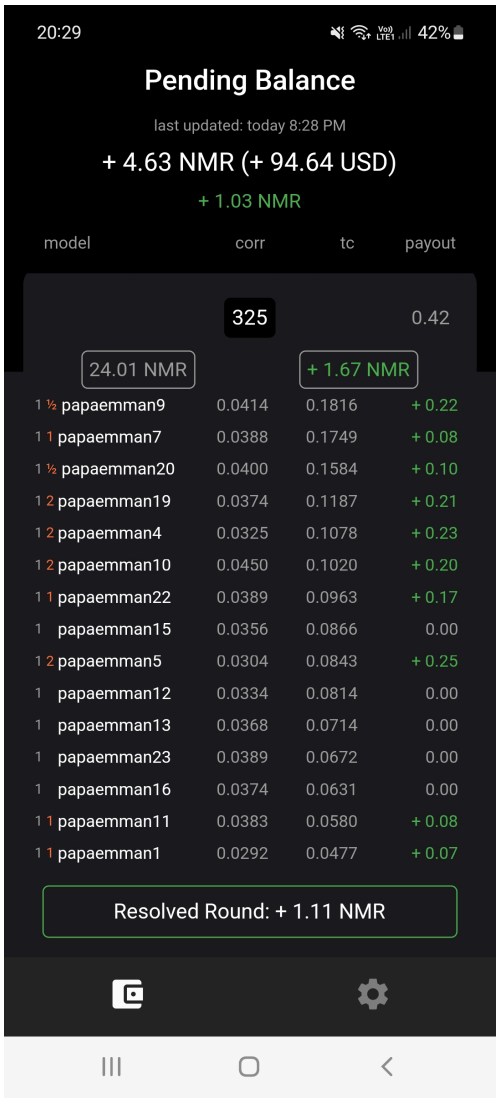


FIGURE 9.8: Screenshot from Numerai Payouts App

A Technical Considerations and Cloud Computing

Numerai datasets consist of around 5 million rows and 1200 features. Therefore the dataset size is considerably large, and working on a personal computer with limited resources in RAM capacity and CPU power is prohibited.

Providing such a huge dataset, Numerai raised the barrier for Data Scientists wanting to participate in the Numerai Tournament. Many individuals don't have access to the computational resources required to tackle a problem on this scale. I struggled with this issue myself, so I decided to set up a Cloud-based high-performance Data Science workspace on Microsoft Azure to be able to participate in the Numerai Tournament.

Azure is a public cloud computing platform with solutions including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) that can be used for services such as analytics, and virtual computing, storage, networking, and much more.



FIGURE A.1: Micrososft Azure (Cloud platform)

Azure is a public cloud computing platform with solutions including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) that can be used for services such as analytics, and virtual computing, storage, networking, and much more.

It can be used to replace or supplement your on-premise servers.

Microsoft Azure's characteristics make it the best choice for this work. More specifically, MS Azure is:

- **Flexible**: Allow to move to compute resources up and down as needed
- **Open**: Supports almost any OS, language, tool, or framework
- **Reliable**: 99.95% availability SLA and 24×7 tech support
- **Global**: Data housed in geosynchronous data centers
- **Economical**: Only pay for what you use

My setup included 6 different resources, organized under the same **resource group** called "NMR". To meet the computing power need for this problem, I provisioned the following:

1. Azure Virtual Machine
2. Public IP address (static)
3. SSD storage disk
4. Virtual network
5. Regular network interface
6. Network Security Group

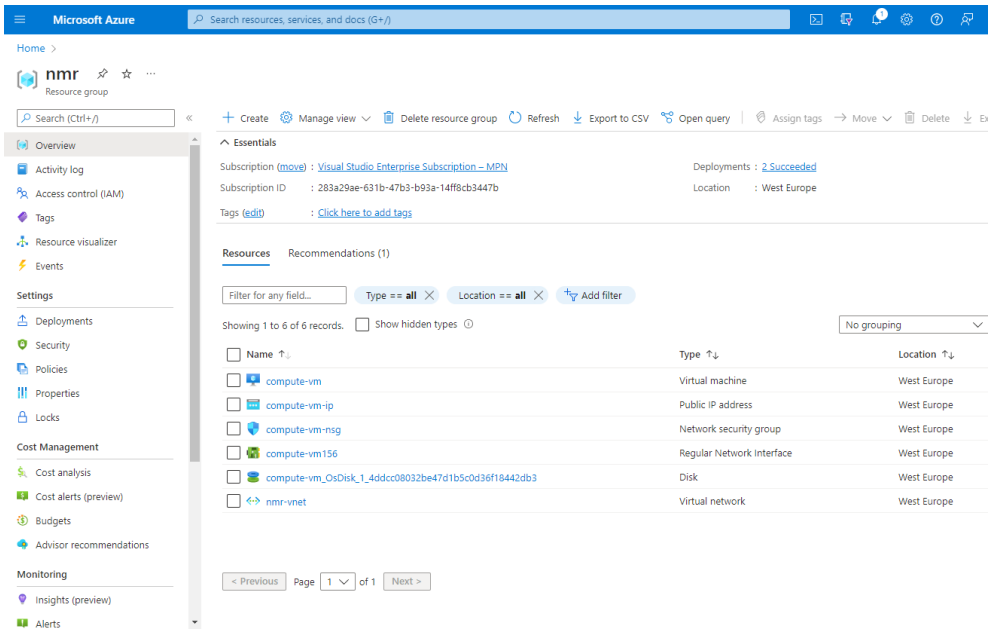


FIGURE A.2: Micrososft Azure (Resource group and Services)

An **Azure virtual machine** is an on-demand, scalable computer resource available in Azure. Virtual machines are generally used to host applications when the customer requires more control over the computing environment than what is offered by other computing resources.

I chose **Ubuntu Virtual Machine** (Ubuntu Server 20.04 LTS - Gen2), which I managed manually during the development phase. Azure is a consumption-based service, so I had to deallocate the VM when I didn't need it. Thus, I wasn't paying for computing power when the VM wasn't active. Moreover, I hat to modify the size of the Azure VM depending on the state of development.

- For the initial EDA, I used the E4bds_v5 VM with 4 vCPUs, and 32 GiB RAM at 0.37$/hour.
- During the ML experimentation process, I used the Standard E20as v5 VM with 20 vCPUs and 160GiB RAM at the cost of 1.27$/hour

Interested readers can find more info about Azure Virtual Machine prices in the dedicated pricing page[1].

Apart from Azure VM, I used an SSD Storage disk of 32GiB for the necessary storage for the VM and Numerai Datasets. Also, The Public IP address and Network related services ensured that I could connect to Azure VM from my local environment using a secure shell (ssh), keeping the same configuration every time.

Before I decided to purchase all required services, I conducted a Total Cost of Ownership Analysis using the TCO calculator provided by Microsoft Azure.

The total cost of Azure services for the complete development phase (experimentation and final training) was around 150$.

[1] https://azure.microsoft.com/en-us/pricing/details/virtual-machines/windows/

B How to purchase NMR

Numerai offers each competitor a unique digital wallet (software) to send, withdraw, store, and stake their NMR tokens. At first, a user needs to purchase NMR, and then they can send it to the related deposit address shown on their wallet page and start staking in the Numerai tournament.

Warning: This wallet only supports NMR, so anything else sent to it will be lost forever.

There are multiple ways to get NMR. The 3 most common ones are

1. **Buy from a centralized Crypto exchange**

   Participants can buy NMR with a debit card or bank transfer on a centralize crypto exchange (web platform) in supported regions. Alternatively, they can acquire Bitcoin (or another cryptocurrency) and convert it to NMR on an exchange. Coinbase[1] is a safe and easy platform that enables users to buy NMR with USD, GBP, EUR, or Bitcoin. Binance[2] is another popular exchange that trades NMR for Bitcoin and other cryptocurrencies.

2. **Buy from a decentralized crypto exchange**

   Uniswap[3] is a decentralized exchange that allows users to swap many tokens built on the ethereum blockchain for NMR using a browser wallet like MetaMask[4]. Users can top up their metamask wallet with Ethereum, convert it to NMR, and send the amount to their wallet.

3. **Buy NMR in the physical world**

   Users can also buy NMR in the physical world, with cash, from other individuals. Of course, this requires a matching between a potential seller and buyer in the real world, which might be difficult to achieve.

[1] https://www.coinbase.com/price/numeraire
[2] https://www.binance.com/en/trade/NMR_BTC?type=spot
[3] https://app.uniswap.org/
[4] https://metamask.io/

B.1 Example workflow: How to send NMR from Binance to the Numerai wallet

Note: The following process applies to all centralized crypto exchanges such as Coinbase, Binance, Kraken, Crypto.com, etc.

1. **Step 1**: Create an account in Binace[5]
2. **Step 2**: Top up your account by depositing fiat currencies (EUR, GBP, USD, etc.) over a bank transfer (SEPA) or a bank card (Visa, Mastercard) or by depositing crypto.
3. **Step 3**: With these funds, buy NMR tokens
4. **Step 4**: Go to numer.ai and create an account
5. **Step 5**: Go to numer.ai/wallet and copy the wallet address

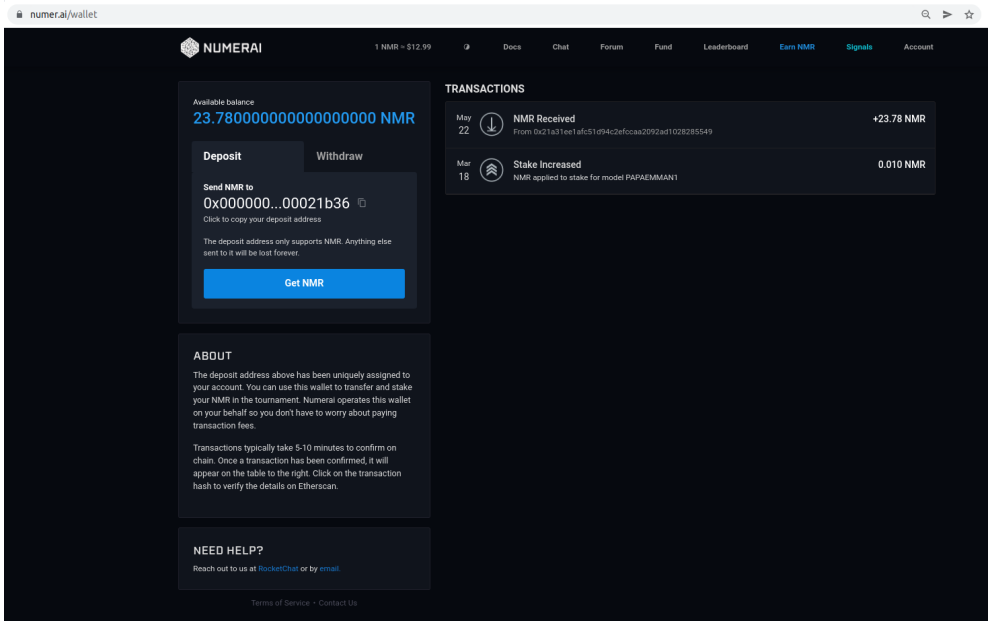


FIGURE B.1: Numerai Wallet

6. **Step 6**: Return to Binance and whitelist your numerai wallet address

[5] https://accounts.binance.com/en/register?ref=53322121

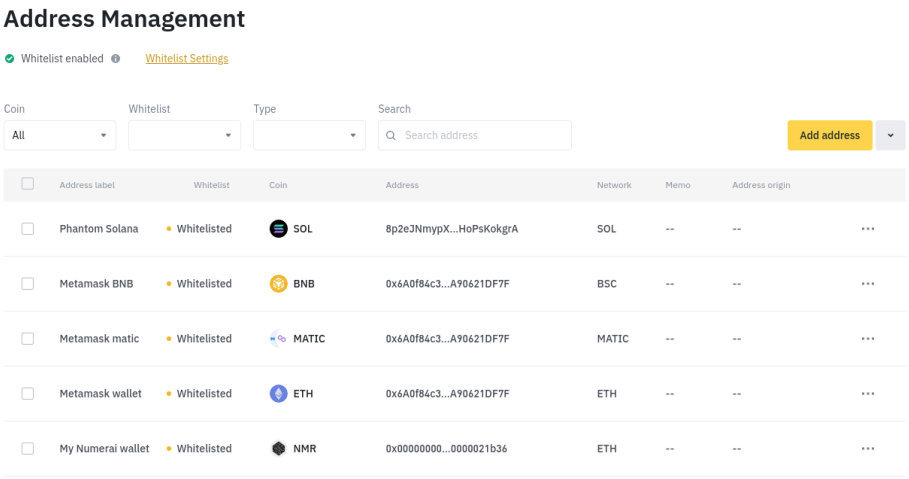


FIGURE B.2: Whitelist NMR wallet address

7. **Step 7**: Withdraw NMR from Binance to your numerai wallet
    - Select Wallet -> Withdraw -> Select coin: NMR (Numeraire)
    - Withdraw to: <numerai_wallet_address>
    - Select the withdraw amount
    - Review the "receive amount" after any gas fees

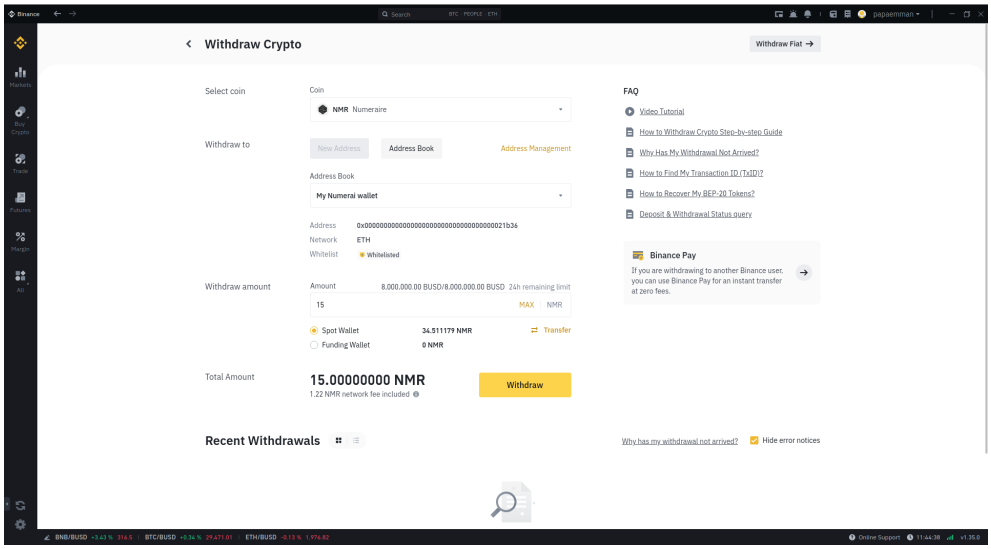


FIGURE B.3: Binance to NMR Wallet

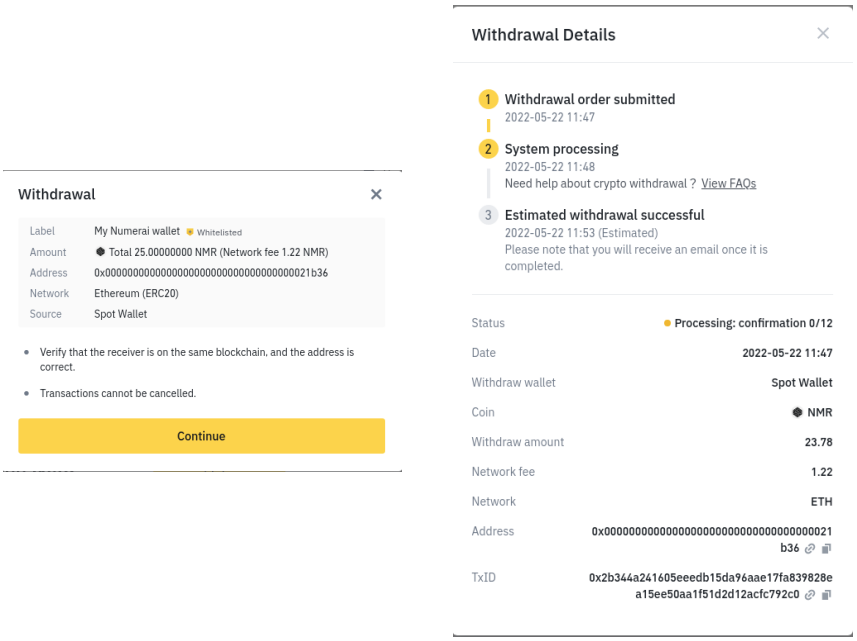


FIGURE B.4: Withdraw NMR from Binance, transaction details

Network fees or "gas fees" are used to compensate Ethereum miners for their work in verifying transactions and securing the network. Gas fees also help keep the network from becoming bogged down by malicious users spamming the network with transactions. On the Ethereum blockchain, gas refers to the cost necessary to perform a transaction on the network.

After some time, the transaction will be processed by the Ethereum network and will be visible in the etherscan explorer[6] and in the numerai wallet (Fig. B.5). The following details are public and uniquely identify the transactions in the ethereum network.

A transaction is a cryptographically signed instruction from an account that changes the state of the blockchain. Block explorers track the details of all transactions in the network. Etherscan is a Block Explorer and Analytics Platform for Ethereum, a decentralized smart contracts platform.

[6] https://etherscan.io/tx/0x2b344a241605eeedb15da96aae17fa839828ea15ee50aa1f51d2d12acfc792c0

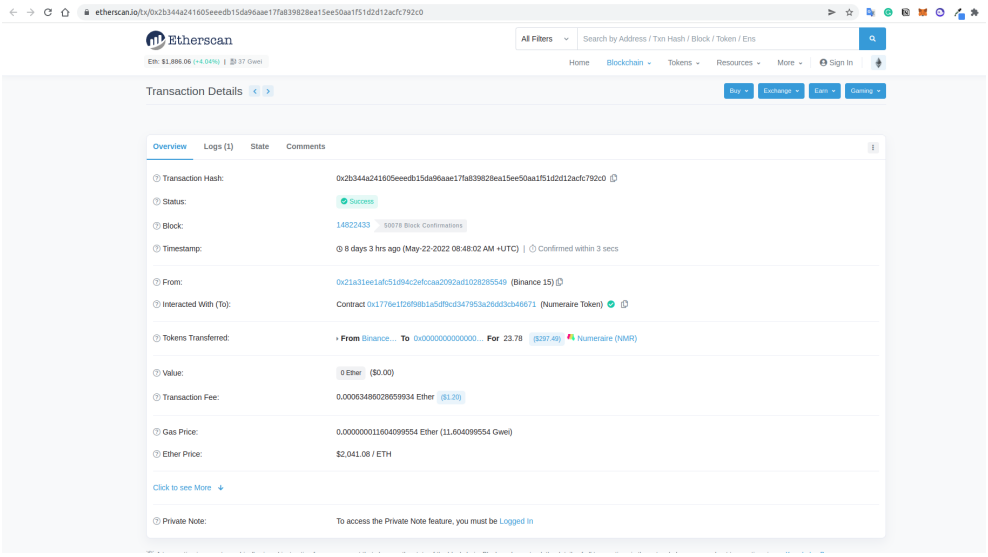


FIGURE B.5: Etherscan transaction details

TABLE B.1: Transaction ID details in etherscan

Information	**Description**
Transaction Hash	A TxHash or transaction is a unique 66-character identifier that is generated whenever a transaction is executed.
Status	The status of the transaction (Success)
Block	Number of the block in which the transaction is recorded. Block confirmations indicated how many blocks have been added since the transaction was minted.
Timestamp	The date and time at which a transaction is mined.
From	The sending party of the transaction
Interacted With (To)	The receiving party of the transaction (could be a contract address)
Token Transferred	List of tokens transferred in the transaction
Value	The values being transacted in Ether and fiat value.
Transaction Fee	(Block Base Fee Per GAs + MaxPriorityFee Per Gas) * Gas Used
Gas Price	Cost per unit of gas specified for the transaction, in Ether and Gwei. The higher the gas price, the higher chance of getting included in a block.
Ether price	Closing price of Ether on date of transaction
Gas Limit & Usage by Txn	Maximum amount of gas allocated for the transaction & the amount eventually used. Normal ETH transfers involve 21,000 gas units, while contracts involve higher values.
Gas Fees	Base Fees refers to the network Base Fee at the time of the block, while Max Fee & Max Priority Fee refer to the max amount a user is willing to pay for their tx & to give to the miner, respectively.
Burnt & Txn Savings Fees	Total amount of ETH burnt from this tx & total fees saved from the amount the user was willing to pay for this tx.
Others	Other data related to this transaction
Input data	Additional data included for this transaction. Commonly used as part of contract interaction or as a message sent to the recipient.
Private note	Private notes are used to keep track of the transaction. Only viewable to Etherscan's user who assign them.

C Risk Management and Hedging

Risk management is the process of identifying, assessing, and controlling threats regarding a situation. These risks stem from various sources, including financial uncertainties, legal liabilities, technology issues, strategic management errors, accidents, natural disasters, and many more.

Specifically, when it comes to investing, Risk Management is a crucial element. As noted in the previous sections, tournament participants mustn't stake funds they can't afford to lose. Numerai has capped the maximum loss for a model in one round at 25% of the stake. It made it to protect participants from severe losses in a single round. Of course, to make things balanced, the maximum earning for a model in one round is capped at 25% of the stake.

C.1 Diversification

In traditional markets, Asset management firms are required by the Securities and Exchange Commission (SEC) to say that the returns an asset made in the past don't mean that future returns will be the same. [1]

The phrase *"Past performance is no guarantee of future results"* is generally treated as a warning label to potential investors, to notice them as to not assume an investment will continue to do well in the future simply because it's done well in the past.

The same applies to Numerai Tournament as well. The past performance of a model doesn't guarantee future performance. A good model will not necessarily make returns every week, so participants should keep track model's performance in the long term. Generally, **staking multiple models** is an excellent way to hedge against volatile performance.

[1] https://www.investor.gov/introduction-investing/investing-basics/glossary/mutual-funds-past-performance

C.2 Hedging against NMR crypto volatility

Hedging is a risk management strategy employed to offset losses in investments by taking an opposite position in a related asset. The reduction in risk provided by hedging also typically results in a reduction in potential profits. Hedging requires one to pay money for the protection it provides, known as the premium.

Hedging techniques generally involve the use of financial instruments known as **derivatives**. Two of the most common derivatives are options and futures. With derivatives, users can develop trading strategies where a loss in one investment is offset by a gain in a derivative. For example, if you wanted to hedge a long stock position, you could buy a put option or establish a collar on that stock. One challenge is that such strategies work for single stock positions.

A **put option** (or "put") is a contract giving the option buyer the right, but not the obligation, to sell—or sell short—a specified amount of an underlying security at a predetermined price within a specified time frame. This predetermined price at which the buyer of the put option can sell the underlying security is called the strike price.

By default, Numerai tournament participants trust the NMR token. To be able to stake in their models they have to **go long to NMR** (meaning that they're buying NMR), therefore they are exposed to the NMR price volatility.

It is not uncommon for some participants to have consistently good returns in the Numerai tournament, earning increasingly more NMRs, but the total value of their Numerai wallet drops in value as the price of NMR drops.

The following diagram shows the price of the NMR to USD for the last year (June 2021 - June 2022).



FIGURE C.1: NMR to USD price

Last year the crypto market experienced a brutal bear market (i.e. a market in which prices are falling, encouraging selling). Moreover, the recent collapse of Luna [2] initiated a more extreme drop in prices. At the beginning of May 2022, Terra Luna the digital coin in the Terra ecosystem crashed from almost 100$ to near zero. Luna was supposed to underpin UST, a stable coin algorithmically pegged at one U.S. dollar. This incident brought additional panic to investors and affected the whole market.

Of course, NMR was impacted by market conditions and lost around 75% of its value, gdropping from around 50$ to 12$. This is a significant loss for long-time Numerai participants as for them to recover their losses they need to experience a growth of around 420%.

To be protected from a fall in NMR, participants can use put options on the NMR, which gives them the right to sell NMR at a specific price (strike price). So, if the NMR price falls below the strike price, these losses will be offset by gains in the put option. This strategy is known as a married put.

It is evident, that every hedging strategy has a cost associated with it. So, before an investor proceeds to use hedging, they should carefully calculate if the potential benefits justify the expense. Furthermore, it is important to highlight, that the goal of hedging isn't to make money; it's to protect from losses. The cost of the hedge, whether it is the cost

[2] https://www.forbes.com/sites/jonmarkman/2022/05/16/cryptos-luna-catastrophe-threatens-conventional-assets/?sh=27f8b24043cb

of an option–or lost profits from being on the wrong side of a futures contract–can't be avoided.

D Published articles and open source code

- How to set up a High-Performance Data Science workspace using Microsoft Azure[1]
- How I automated my Numerai weekly submission pipeline for free, using Azure functions and python[2]
- Machine Learning hyperparameter for XGBoost and LightGBM using Bayesian optimazation with hyperopt[3]
- How to manage an Azure Virtual Machine (VM) from Azure CLI[4]

[1] https://gist.github.com/papaemman/1e6c48eef07e1ea010a49b1deb5f799b

[2] https://medium.com/@papaemman.pan/how-i-automated-my-numerai-weekly-submissions-pipeline-for-free-using-azure-functions-and-python-9bcf8382af1c

[3] https://gist.github.com/papaemman/70f363d54f57a2dfced84b7c7bdb6220

[4] here

E Contact information

About Me

Panagiotis (Panos) Papaemmanouil

Mathematician turned Data Scientist turned Technology entrepreneur. Passionate about real-world data-driven problems. Born and raised in Greece.

Contact information:

- email: papaemman.pan@gmail.com
- Linkedin: https://www.linkedin.com/in/panagiotis-papaemmanouil/
- GitHub: https://github.com/papaemman

If you found this work helpful, consider donating NMR. Numerai wallet address: 0x0000000000000000000000000000000000021b36

Bibliography

Akshay Agrawal Brandon Amos, Shane Barratt Stephen Boyd Steven Diamond Zico Kolter (2012). "Differentiable Convex Optimization Layers". In: DOI: 10.48550/arXiv.1910.12430.

Buterin, Vitalik (2014). "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform". In: URL: https://ethereum.org/en/whitepaper/.

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

Craib, Richard. *Numerai's Master Plan*. URL: https://medium.com/numerai/numerais-master-plan-1a00f133dba9.

Craib, Richard, Geoffrey Bradway, Xander Dunn, and Joey Krug (2017). "Numeraire: A Cryptographic Token for Coordinating Machine Intelligence and Preventing Overfitting". In: URL: https://numer.ai/whitepaper.pdf.

Douceur, John R. (2002). "The Sybil Attack". In: URL: https://www.freehaven.net/anonbib/cache/sybil.pdf.

Fabian Vogelsteller, Vitalik Buterin. *EIP-20: Token Standard*. URL: https://eips.ethereum.org/EIPS/eip-20/.

Hastie, Trevor (2020). "Ridge Regularization: An Essential Concept in Data Science". In: *Technometrics* 62.4, pp. 426–433. DOI: 10.1080/00401706.2020.1791959. eprint: https://doi.org/10.1080/00401706.2020.1791959. URL: https://doi.org/10.1080/00401706.2020.1791959.

Investopedia. *Quantitative Hedge Fund*. URL: https://www.investopedia.com/terms/q/quantfund.asp.

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu (2017). "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 3149–3157. ISBN: 9781510860964.

LLC, Numerai. *Numerai*. URL: https://numer.ai/.

– *Numerai Hedge Fund*. URL: https://numer.ai/fund/.

Nakamoto, Satoshi (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System". In: URL: https://bitcoin.org/bitcoin.pdf.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Refaeilzadeh, Payam, Lei Tang, and Huan Liu (2009). "Cross-Validation". In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, pp. 532–538. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_565. URL: https://doi.org/10.1007/978-0-387-39940-9_565.

Tibshirani, Robert (1996). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1, pp. 267–288. ISSN: 00359246. URL: http://www.jstor.org/stable/2346178 (visited on 09/27/2022).