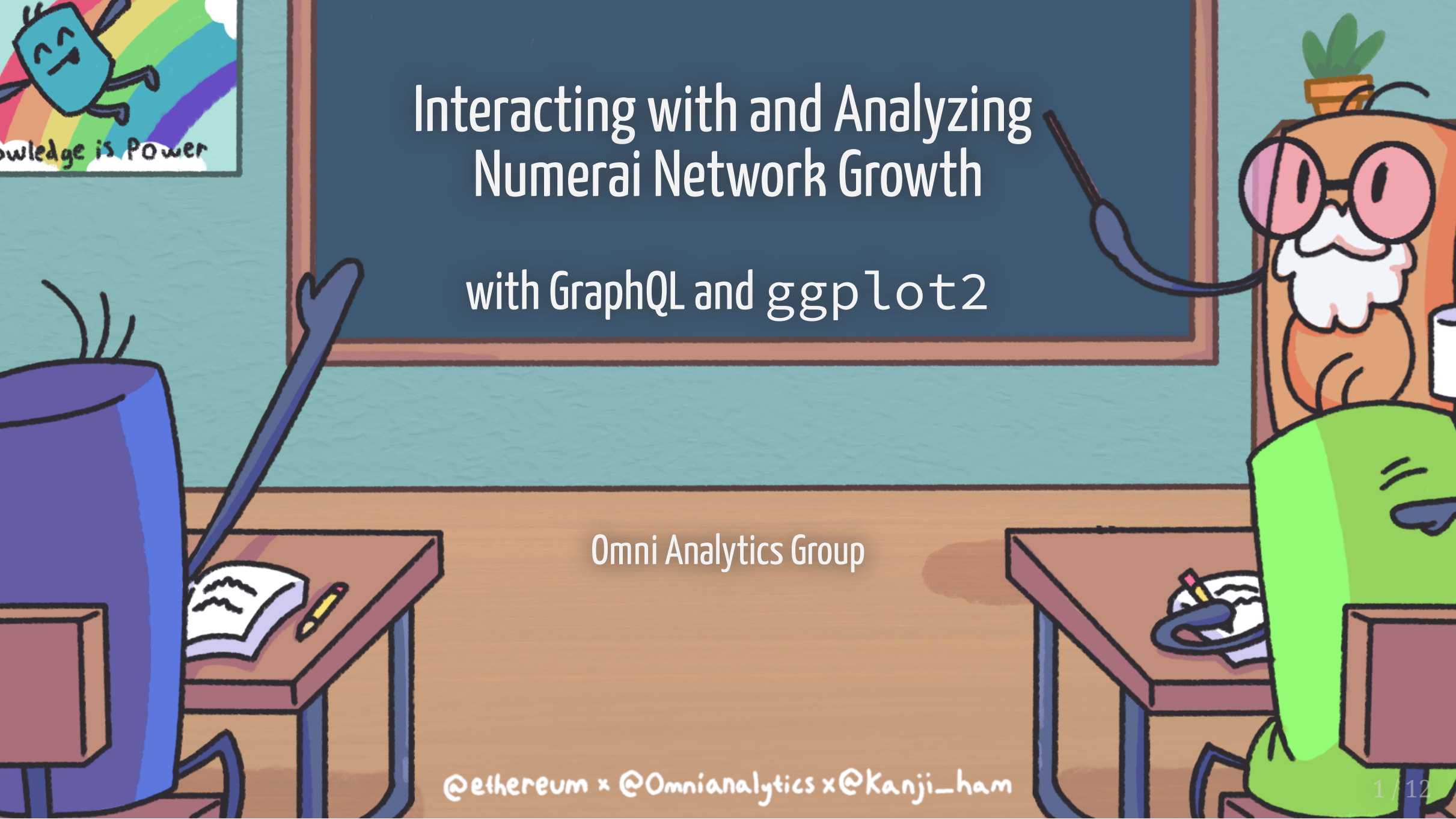


Interacting with and Analyzing Numerai Network Growth

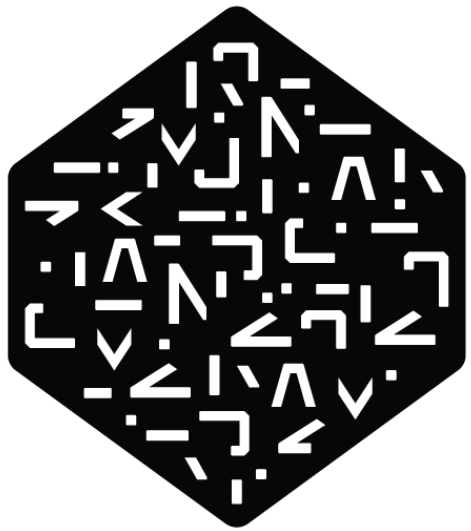
with GraphQL and `ggplot2`

Omni Analytics Group

@ethereum x @Omnianalytics x @Kanji_ham

Background

Numerai is a modern-day hedge fund where a large pool of anonymous data scientists submit models. These models are then used by the AI-based Numerai backend to perform trades. Numerai has seen explosive growth in recent years. Can we use R-based tools to analyze this growth?



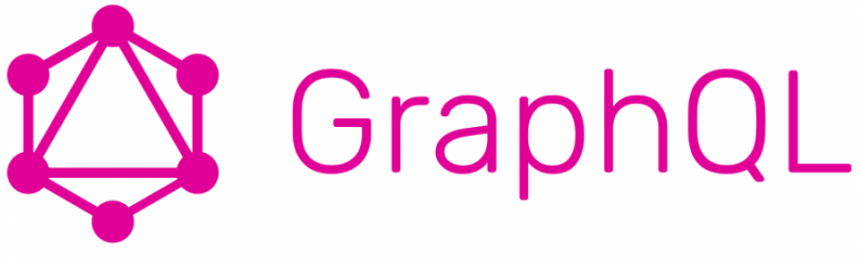
NUMERAI

GraphQL

Numerai's API uses GraphQL as a backend, which is a structured query language much like SQL itself.

- Developed initially as an internal project by Facebook
- Provides a method for development of APIs much like REST
- Flexible and rich compared to REST and therefore may be less suitable for more simple web APIs
- Where REST APIs are organized as a collection of endpoints, GraphQL is organized as a collection of types and fields with their associated datatype specification.

Let's take a look at how to interact with GraphQL in R...

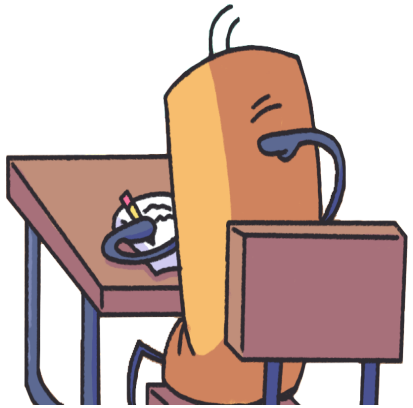


GraphQL in R

There are two primary methods of accessing the Numerai data:

- Directly, using the `ghql` R package
- Indirectly, by downloading the parsed JSON data from the Numerai API and reading it into R

We will use `ghql` to keep the steps reproducible.



ghql

Let's begin by installing `ghql`:

```
install.packages("ghql")
```

Next, we connect to the Numerai API:

```
library(ghql)

con <- GraphqlClient$new(
  url = "https://api-tournament.numer.ai/"
)
```

This connection object maintains the GraphQL client connection to the Numerai API server. Note that it is an R6-style object and hence is initialized with

Making a Leaderboard Query

We can perform a query with the following:

```
qry <- Query$new() #create a new query

qry$query('leaderboard', '{
  v2Leaderboard {
    username
    corrRep
    mmcRep
    return_52Weeks
    return_13Weeks
  }
}')

result <- con$exec(qry$queries$leaderboard) #execute query
```

Note that we begin with the initialization of a new instance of the R6 `Query` class, and then call the `query()` method, passing in two arguments:

- The name of the resulting object
- The raw GraphQL query that is to be executed

Viewing the Results 

A quick peak at the raw return value shows JSON data that we need to parse using the `fromJSON()` function, in order to retrieve a data frame:

```
print(paste0(substring(result, 1, 50), "..."))
```

```
## [1] "{\"data\":{\"v2Leaderboard\":[{\"corrRep\":0.03867359120..."
```

And the parsed data:

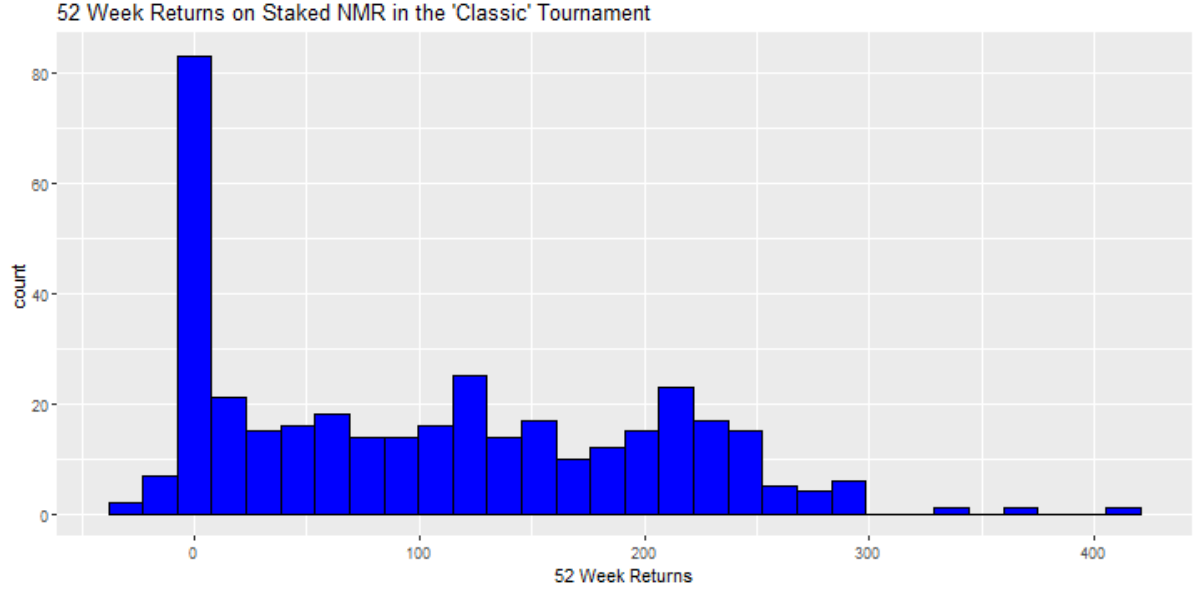
```
nmr <- fromJSON(result)[[1]]$v2Leaderboard

head(nmr, 3) %>%
  kable()
```

corrRep	mmcRep	return_13Weeks	return_52Weeks	username
0.0386736	0.0245899	89.41417	NA	hiryuu
0.0383408	0.0195207	75.55821	NA	era__mix__2000
0.0368198	0.0158979	83.96748	NA	lazerfazer4

52 Week Returns

Now we can use `ggplot()` in order to visualize aspects of the data!

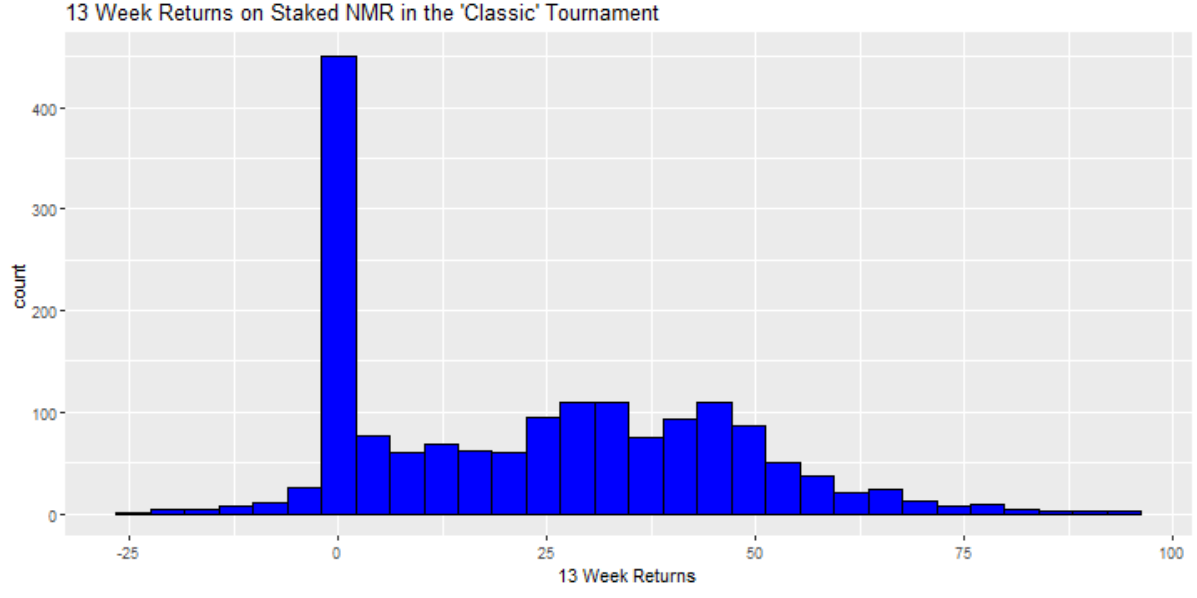
```
ggplot(data = nmr, aes(x = return_52Weeks)) +
  geom_histogram(fill = "blue", colour = "black") +
  labs(title = "52 Week Returns on Staked NMR in the 'Classic' Tournament", x = "52 Week Returns'
```



13 Week Returns

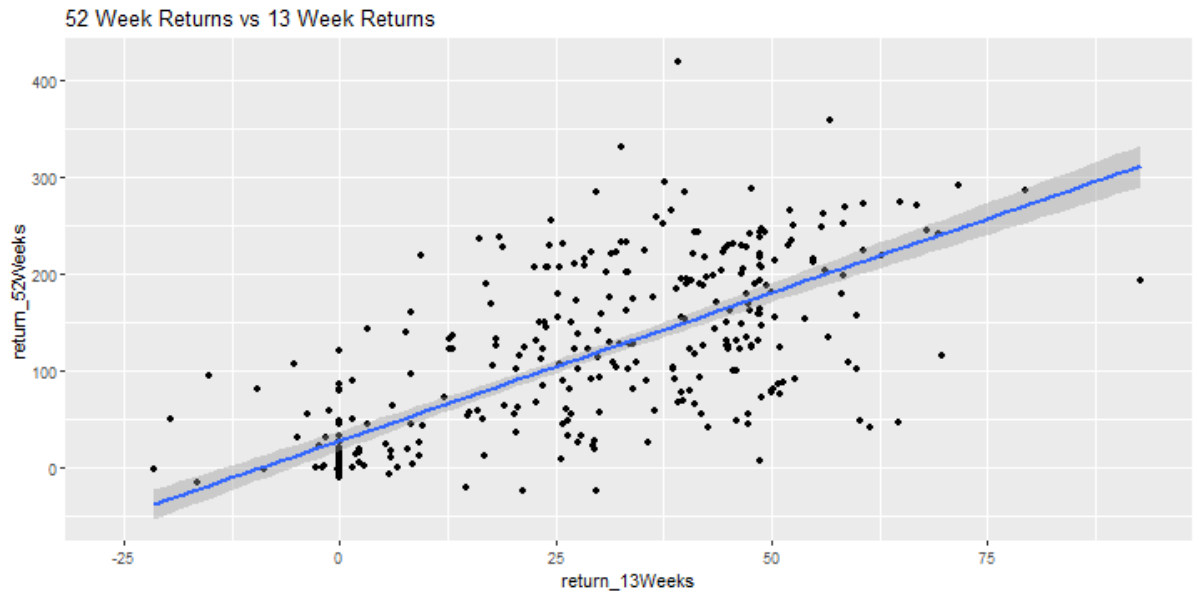
Now we can use `ggplot()` in order to visualize aspects of the data!

```
ggplot(data = nmr, aes(x = return_13Weeks)) +
  geom_histogram(fill = "blue", colour = "black") +
  labs(title = "13 Week Returns on Staked NMR in the 'Classic' Tournament", x = "13 Week Returns'
```



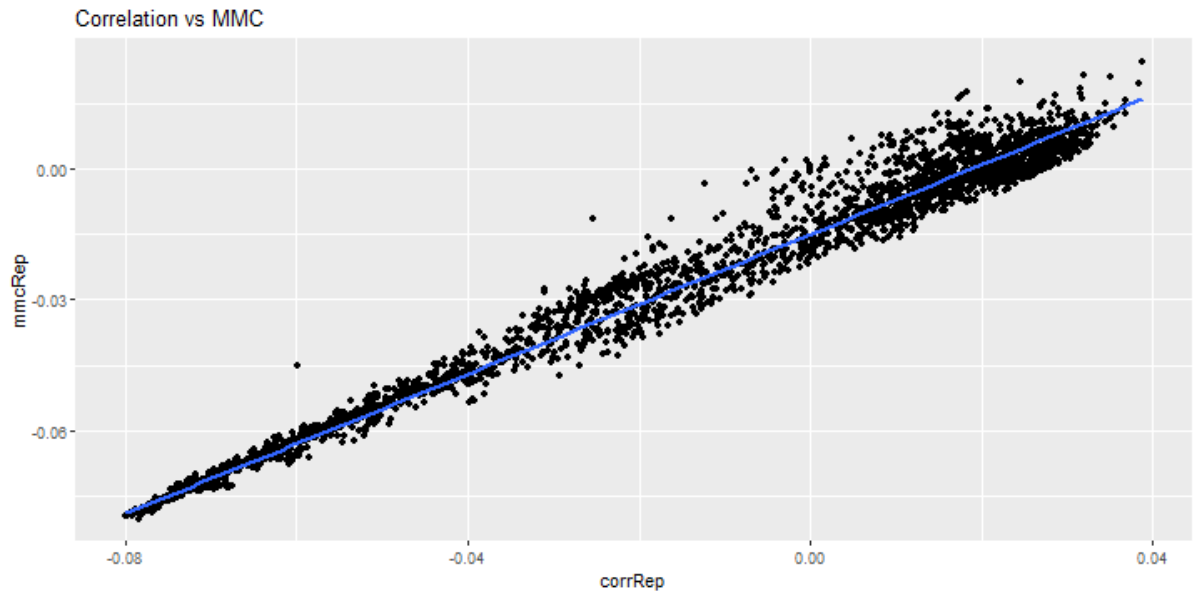
52 Week Returns vs 13 Week Returns

```
ggplot(data = nmr, aes(x = return_13Weeks, y = return_52Weeks)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "52 Week Returns vs 13 Week Returns")
```



Correlation vs MMC

```
ggplot(data = nmr, aes(x = corrRep, y = mmcRep)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Correlation vs MMC")
```



Conclusion

You can now do the following using `ghql` and `jsonlite` R packages:

- Connect to Numerai API
- Create a Query
- Execute a query
- Parse the data

Try it yourself!