Fintech Group Project 01

Azuki, BAYC and Crypto Punks NFTs.

by Github users /@dockingbay24 /@angel-estrada7 and /@mmsaki, **2022.**

Table of Contents

Azuki Token ID: 7558 **| Current Owner:** ron1n.eth **| Sale:** 27ETH

everyone else

So what happens if somebody registers somebody elses art on Deloitte?

You cannot unsteal that art that was put up without consent.

So what stops me from doing this, and saying mine now!

The NFT Alpha

Monetizes memes

Money goes brrrrrrrrrrr

Creates something for the community

1. Hypothesis

Hypothesis of Project

Our motivation and summary…

- Should you invest in Azuki, BAYC or Crypto Punks?
- How can you tell which collection is performing well?

Data Collection

Describing what kinds of data we needed and where to find it.

2. Data Collection

Collecting data for NFT Collections

- Covalent API
- Etherscan Python Dependacy

group_project_01

May 5, 2022

0.1 Import dependancies

[1]:
```
# Import dependancies
import os
import requests
import pandas as pd
import json
from dotenv import load_dotenv
from etherscan_py import etherscan_py
import plotly.express as px
```

[2]:
```
# Loading .env containing our keys
load_dotenv()
```

[2]: True

[3]:
```
# create variable for api key
api_key = os.getenv('COVALENT_API_KEY')
type(api_key)
```

[3]: str

0.2 Current value of ETH

[4]:
```
# import dependancy
from etherscan_py import etherscan_py
etherscan_api = etherscan_py.Client(os.getenv('ETHERSCAN_API'))

# Print current eth price and latest block height
eth_value = etherscan_api.get_eth_price()
eth_value
```

[4]: 2738.51

3. Data Cleanup & Exploration

Exploring our collection data through APIs preparing it for analysis

0.3 Set variables

[5]:
```
# Append url for our api
url = "https://api.covalenthq.com/v1"
chain_id = "/1"
azuki_address = "/0xED5AF388653567Af2F388E6224dC7C4b3241C544"
cryptopunks_address = "/0xb47e3cd837dDF8e4c57F05d70Ab865de6e193BBB"
BAYC_address = "/0xBC4CA0EdA7647A8aB7C2061c2E118A18a936f13D"
date_option = '/?quote-currency=USD&format=JSON&from=2017-01-01&to=2022-05-01'
page_option = '/transactions_v2/?
 ↪quote-currency=USD&format=JSON&block-signed-at-asc=false&no-logs=false&page-number=0&page-s
api_option = "&key=" + api_key
api_no_option = '/?key=' + api_key
```

0.4 1. Azuki Daily Volume

[6]:
```
# Create variables needed for owner data and add to url
azuki_url = url + chain_id + "/nft_market/collection" + azuki_address +␣
 ↪api_no_option

# Get request
azuki_historical_json = requests.get(azuki_url).json()

# Convert historical json data to a dataframe and view data
azuki_df = pd.DataFrame(azuki_historical_json['data']['items'])

# Set index to date
azuki_df = azuki_df.set_index('opening_date')

# Create Volume dataframe
azuki_vol_df = pd.DataFrame(azuki_df, columns = ['volume_quote_day',␣
 ↪'unique_token_ids_sold_count_day']).sort_index()
azuki_vol_df.head()
```

[6]:

opening_date	volume_quote_day	unique_token_ids_sold_count_day
2022-01-12	45941404.0	2402
2022-01-13	25129178.0	1318
2022-01-14	168151840.0	470
2022-01-15	4408686.0	499
2022-01-16	295638336.0	368

[7]:
```
# Plot Volume quote per day
azuki_volume = azuki_vol_df['volume_quote_day'].astype(int)

# Plot Historical daily volume
px.bar(azuki_volume)
```

0.5 1. Azuki Historical transactions

[8]:
```
# Quering the API for transaction data
azuki_tx_url = url + chain_id + "/address" + azuki_address + page_option +␣
 ↪api_option
azuki_tx = requests.get(azuki_tx_url).json()

# Convert transactions data to dataframe
azuki_tx_df = pd.DataFrame(azuki_tx['data']['items'], columns =␣
 ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
 ↪set_index('block_signed_at').sort_index()

azuki_tx_df.head()
```

[8]:

block_signed_at	to_address_label	fees_paid	value_quote
2022-05-02T18:43:51Z	LooksRare: Exchange	22370093235357597	0.000000
2022-05-02T18:47:06Z	None	2448400799338417	0.000000
2022-05-02T18:49:48Z	None	5009529942416780	0.000000
2022-05-02T18:54:24Z	None	5166370935350339	0.000000
2022-05-02T19:06:28Z	None	18519456263386155	86919.574609

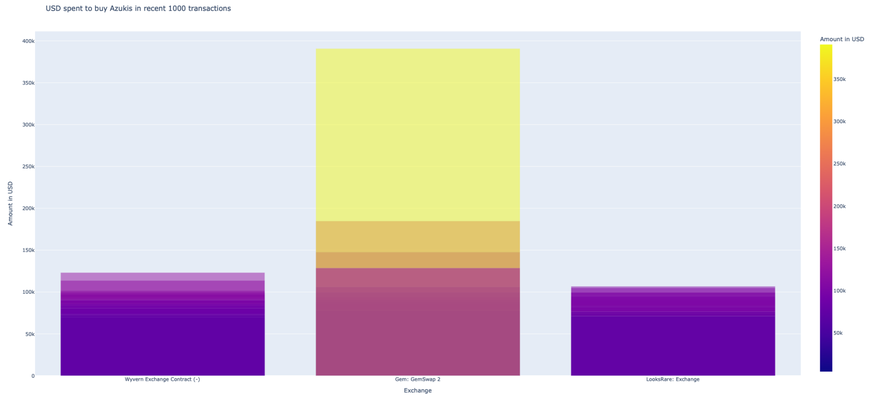
0.6 1.a Azuki Historical Sales

[9]:
```
# Filter Through data for non null transactions
azuki_sales_df = azuki_tx_df[azuki_tx_df['value_quote'] != 0]
azuki_sales = azuki_sales_df[azuki_sales_df['to_address_label'].notnull()]

# Creating the plot using plotly express
azuki_fig = px.bar(azuki_sales,
                   x='to_address_label',
                   y= 'value_quote',
                   color='value_quote',
                   height=1020,
                   width = 1000,
                   barmode='overlay',
                   labels={'value_quote':'Amount in USD', 'to_address_label':␣
 ↪'Exchange'},
```

4. Data Analysis

Analyzing our data and developing figures to answer our questions

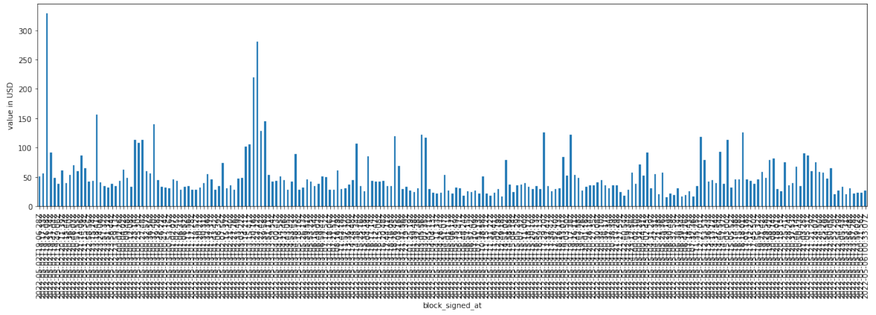
```
                    title='USD spent to buy Azukis in recent 1000 transactions'
                   )
azuki_fig.show()
```



0.7 1.b Azuki transaction fees paid

[10]:
```
# Filter Through data for non null transactions
azuki_fees = azuki_sales_df['fees_paid'].astype(int)/10**18*eth_value

azuki_fees.plot.bar(rot = 90, figsize = (20,5), ylabel = 'value in USD')
```

[10]: <AxesSubplot:xlabel='block_signed_at', ylabel='value in USD'>



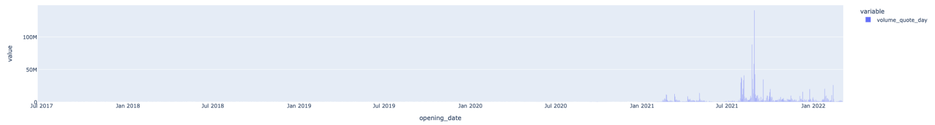
0.8 2. Cryptopunks Daily Volume

[11]:
```
# Create variables needed for owner data and append to url
cryptopunks_historical_url = url + chain_id + "/nft_market/collection" +␣
 ↪cryptopunks_address + api_no_option

# Get request
cryptopunks_historical_json = requests.get(cryptopunks_historical_url).json()

# Convert historical json data to a dataframe and view data
cryptopunks_df = pd.DataFrame(cryptopunks_historical_json['data']['items'])

# Set index to date
cryptopunks_df = cryptopunks_df.set_index('opening_date')

# Create Volume dataframe
cryptopunks_vol_df = pd.DataFrame(cryptopunks_df, columns =␣
 ↪['volume_quote_day', 'unique_token_ids_sold_count_day']).sort_index()
cryptopunks_vol_df.head()
```

[11]:

opening_date	volume_quote_day	unique_token_ids_sold_count_day
2017-06-23	0.0	19
2017-06-24	0.0	22
2017-06-25	0.0	11
2017-06-26	0.0	18
2017-06-27	0.0	35

[12]:
```
# Plot Volume quote per day
cryptopunks_volume = cryptopunks_vol_df['volume_quote_day'].astype(int)

# cryptopunks_volume.plot.line(figsize = (20,4))

px.bar(cryptopunks_volume)
```



0.9 2a Cryptopunks Historical transactions

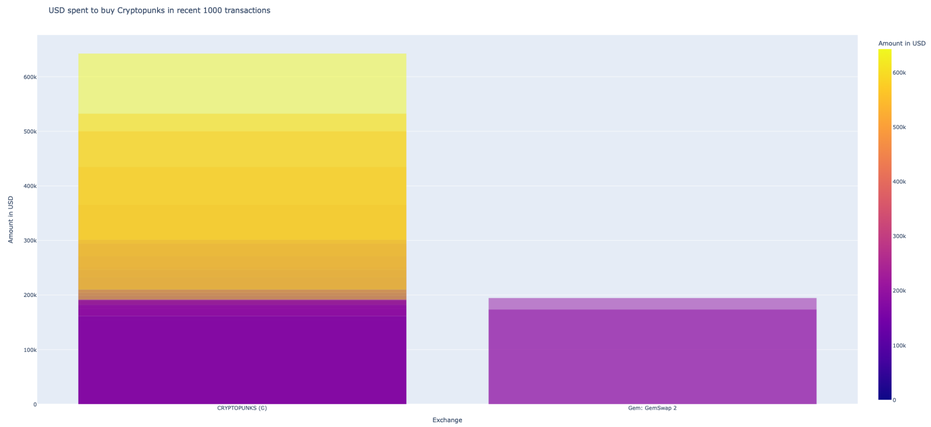
[13]:
```
# Quering the API for transaction data
cryptopunks_tx_url = url + chain_id + "/address" + cryptopunks_address +␣
 ↪page_option + api_option
cryptopunks_tx = requests.get(cryptopunks_tx_url).json()

# Convert transactions data to dataframe
cryptopunks_tx_df = pd.DataFrame(cryptopunks_tx['data']['items'], columns =␣
 ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
 ↪set_index('block_signed_at').sort_index()

cryptopunks_tx_df.head()
```

[13]:

block_signed_at	to_address_label	fees_paid	value_quote
2022-04-27T17:46:26Z	CRYPTOPUNKS ( )	5901776729714157	0.000000
2022-04-27T17:48:18Z	None	28932221174799876	0.000000
2022-04-27T17:53:12Z	CRYPTOPUNKS ( )	6838075277247300	174127.151367
2022-04-27T17:59:35Z	CRYPTOPUNKS ( )	4845298000000000	182552.658691
2022-04-27T18:00:07Z	CRYPTOPUNKS ( )	1797562348480696	0.000000

0.10 2.a Cryptopunks Historical Sales

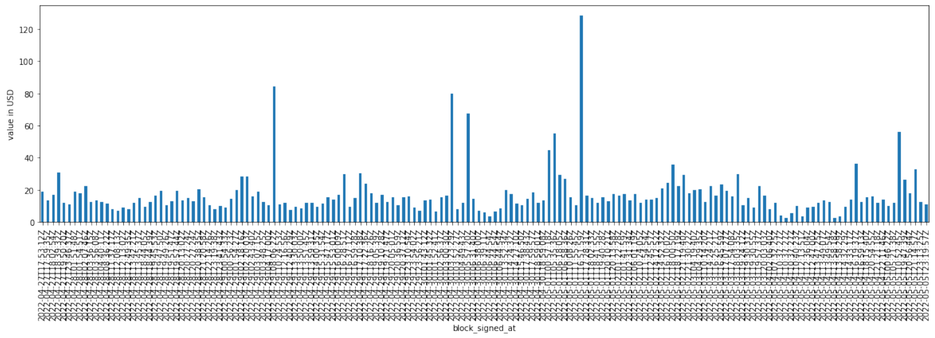
[14]:
```
# Filter Through data for non null transactions
cryptopunks_sales_df = cryptopunks_tx_df[cryptopunks_tx_df['value_quote'] != 0]
cryptopunks_sales =␣
 ↪cryptopunks_sales_df[cryptopunks_sales_df['to_address_label'].notnull()].
 ↪dropna()


# Creating the plot using plotly express
cryptopunks_fig = px.bar(cryptopunks_sales,
                         x='to_address_label',
                         y= 'value_quote',
                         color='value_quote',
                         height=1020,
                         width = 1000,
                         barmode = 'overlay',
                         labels={'value_quote':'Amount in USD',␣
 ↪'to_address_label': 'Exchange'},
                         title='USD spent to buy Cryptopunks in recent 1000␣
 ↪transactions'
                        )
cryptopunks_fig.show()
```



0.11 2.b Cryptopunks Fees paid

[15]:
```
# Filter Through data for non null transactions
cryptopunks_fees = cryptopunks_sales_df['fees_paid'].astype(int)/
 ↪10**18*eth_value

cryptopunks_fees.plot.bar(rot = 90, figsize = (20,5), ylabel = 'value in USD')
```

[15]: <AxesSubplot:xlabel='block_signed_at', ylabel='value in USD'>



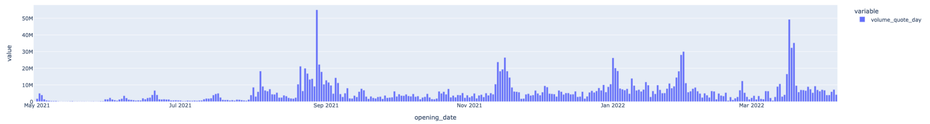
0.12 3. BAYC Daily Volume

[16]:
```
# Create variables needed for owner data and add to url
BAYC_historical_url = url + chain_id + "/nft_market/collection" + BAYC_address␣
↪+ api_no_option

# Get request
BAYC_historical_json = requests.get(BAYC_historical_url).json()

# Convert historical json data to a dataframe and view data
BAYC_df = pd.DataFrame(BAYC_historical_json['data']['items'])

# Set index to date
BAYC_df = BAYC_df.set_index('opening_date')

# Create Volume dataframe
BAYC_vol_df = pd.DataFrame(BAYC_df, columns = ['volume_quote_day',␣
↪'unique_token_ids_sold_count_day']).sort_index()
BAYC_vol_df.head()
```

[16]:

opening_date	volume_quote_day	unique_token_ids_sold_count_day
2021-04-30	8.241964e+02	1
2021-05-01	1.737182e+06	1635
2021-05-02	4.950946e+06	1534
2021-05-03	3.948996e+06	996
2021-05-04	1.388962e+06	336

[17]:
```
# Plot Volume quote per day
BAYC_volume = BAYC_vol_df['volume_quote_day'].astype(int)

# BAYC_volume.plot.bar(figsize = (20,4))
px.bar(BAYC_volume)
```

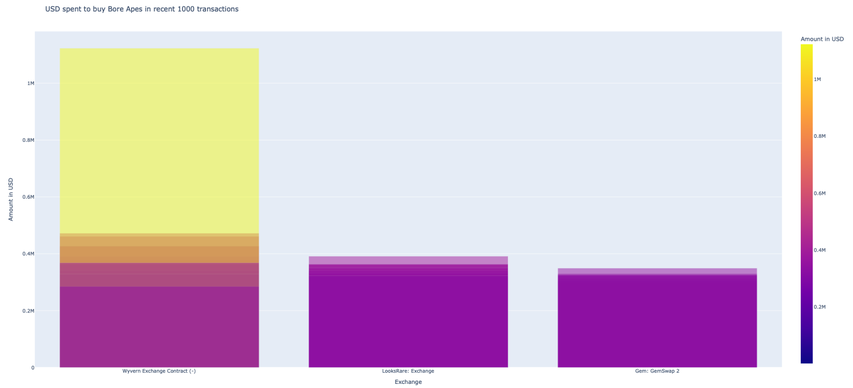


0.13 3a BAYC Historical Sales

[18]:
```
# Quering the API for transaction data
BAYC_tx_url = url + chain_id + "/address" + BAYC_address + page_option +␣
 ↪api_option
BAYC_tx = requests.get(BAYC_tx_url).json()

# Convert transactions data to dataframe
BAYC_tx_df = pd.DataFrame(BAYC_tx['data']['items'], columns =␣
 ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
 ↪set_index('block_signed_at').sort_index()

BAYC_tx_df.head()
```

[18]:
```
                                to_address_label          fees_paid  \
block_signed_at
2022-05-02T20:37:49Z                        None  80753868585244770
2022-05-02T20:44:31Z  Wyvern Exchange Contract (-)  19052443987785043
2022-05-02T20:50:02Z                        None  11388694937249759
2022-05-02T20:55:00Z                        None  17955072367955640
2022-05-02T20:55:32Z                        None   4449937167409760

                      value_quote
block_signed_at
2022-05-02T20:37:49Z      0.00000
2022-05-02T20:44:31Z  303074.83252
2022-05-02T20:50:02Z      0.00000
2022-05-02T20:55:00Z      0.00000
2022-05-02T20:55:32Z      0.00000
```

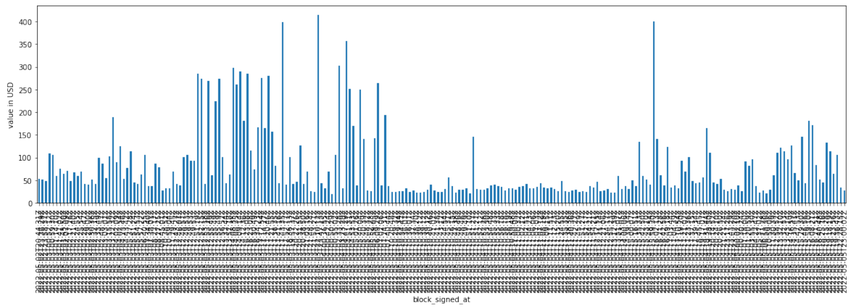
[19]:
```
# Filter Through data for non null transactions
BAYC_sales_df = BAYC_tx_df[BAYC_tx_df['value_quote'] != 0]
BAYC_sales = BAYC_sales_df[BAYC_sales_df['to_address_label'].notnull()].dropna()

# Creating the plot using plotly express
BAYC_fig = px.bar(BAYC_sales,
             x='to_address_label',
             y= 'value_quote',
             color='value_quote',
             height=1020,
             width = 1000,
                  barmode = 'overlay',
                  labels={'value_quote':'Amount in USD', 'to_address_label':␣
 ↪'Exchange'},
             title='USD spent to buy Bore Apes in recent 1000 transactions'
            )
BAYC_fig.show()
```



0.14 3.b BAYC Fees paid

[20]:
```
# Filter Through data for non null transactions
BAYC_fees = BAYC_sales_df['fees_paid'].astype(int)/10**18*eth_value

BAYC_fees.plot.bar(rot = 90, figsize = (20,5), ylabel = 'value in USD')
```

[20]:
```
<AxesSubplot:xlabel='block_signed_at', ylabel='value in USD'>
```



0.15 Combine Total Sales

[21]:
```
# Group by address label and sum the value
azuki_total = azuki_sales.groupby('to_address_label').sum()
cryptopunks_total = cryptopunks_sales.groupby('to_address_label').sum()
```

```
BAYC_total = BAYC_sales.groupby('to_address_label').sum()
```

[22]:
```
# Combine and rename columns for our total sales data
combined_totals = pd.concat([azuki_total,cryptopunks_total,BAYC_total], axis=1)
combined_totals.columns = ['azuki_total', 'cryptopunks_total','BAYC_total']
```

[23]:
```
# Plot for combined figure
combined_total_fig = px.bar(combined_totals)


# Show Figure
combined_total_fig.show()
```

0.16 Combine Total Fees

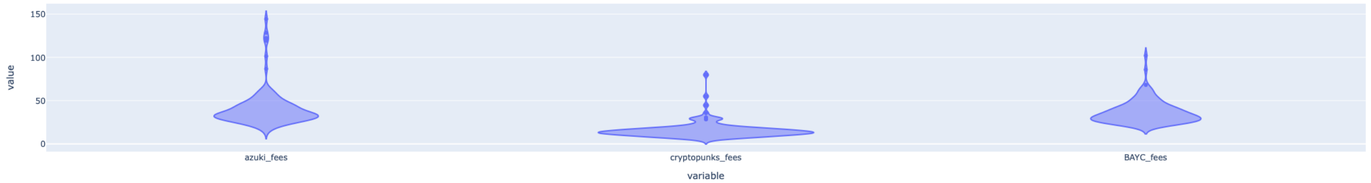
[24]:
```
# Group by address label and sum the value
combined_totals
```

[24]:

to_address_label	azuki_total	cryptopunks_total	BAYC_total
Gem: GemSwap 2	1.494632e+06	3.683260e+05	2.568193e+06
LooksRare: Exchange	2.452695e+06	NaN	8.755434e+06
Wyvern Exchange Contract (-)	9.148483e+06	NaN	2.856156e+07
CRYPTOPUNKS ( )	NaN	2.937500e+07	NaN

[25]:
```
# Combine and rename columns for our total sales data
azuki_usd_fees = azuki_sales['fees_paid'].astype(int)/10**18*eth_value
cryptopunks_usd_fees = cryptopunks_sales['fees_paid'].astype(int)/
 ↪10**18*eth_value
BAYC_usd_fees = BAYC_sales['fees_paid'].astype(int)/10**18*eth_value

# Combine dataframe and drop nulls
combined_usd_fees = pd.concat([azuki_usd_fees.reset_index(drop=True),
                               cryptopunks_usd_fees.reset_index(drop=True),
                               BAYC_usd_fees.reset_index(drop=True)],
                              axis=1
                             ).dropna()
combined_usd_fees.columns = ['azuki_fees', 'cryptopunks_fees','BAYC_fees']
```

[26]:
```
# Plot for combined figure
combined_fees_fig = px.violin(combined_usd_fees)

# Show Figure
combined_fees_fig.show()
```



5. Discussion

Combining our data and discussing our findings

6. Postmoterm

Did we find everything we expected to find?

- Our difficulties and how we dealt with them
- Additional questions that came up that we didn’t answer
- What would we research next if we had more time?

7. Questions

Open floor Q&A with the audience

thank you,