NFTAnalysis

May 1, 2023

[2]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import plotly.express as px
import plotly.graph_objects as go
import warnings
warnings.filterwarnings('ignore')
```

1 Data Description

[3]:
```
df = pd.read_csv('Azuki_BAYC_MAYC_Otherdeed_Moonbirds.csv',␣
 ↪parse_dates=['timestamp', 'last_refreshed'])
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 345521 entries, 0 to 345520
Data columns (total 37 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   __indexer_id            345521 non-null  object
 1   __confirmed             345521 non-null  bool
 2   __block_number          345521 non-null  int64
 3   block_number            345521 non-null  int64
 4   log_index               345521 non-null  int64
 5   transaction_hash        345521 non-null  object
 6   timestamp               345521 non-null  datetime64[ns, UTC]
 7   exchange_name           345521 non-null  object
 8   contract_version        345521 non-null  object
 9   aggregator_name         26046 non-null   object
 10  contract_address        345521 non-null  object
 11  token_id                345521 non-null  int64
 12  is_multi_token_sale     345521 non-null  bool
 13  multi_token_sale_index  345521 non-null  int64
 14  price                   345521 non-null  object
 15  usd_price               345521 non-null  float64
 16  eth_price               345521 non-null  float64
```

```
 17  native_price            345521 non-null  float64
 18  payment_token_address   345521 non-null  object
 19  quantity                345521 non-null  int64
 20  seller_address          345521 non-null  object
 21  buyer_address           345521 non-null  object
 22  royalty_fee             345521 non-null  float64
 23  platform_fee            345521 non-null  float64
 24  minted_timestamp        345521 non-null  object
 25  supply                  345521 non-null  int64
 26  name                    75804 non-null   object
 27  description             0 non-null       float64
 28  image_url               345521 non-null  object
 29  external_url            30070 non-null   object
 30  media_url               0 non-null       float64
 31  properties              345521 non-null  object
 32  metadata_url            332333 non-null  object
 33  last_refreshed          345521 non-null  datetime64[ns, UTC]
 34  flattened_properties    345521 non-null  object
 35  __updated_block_number  345521 non-null  int64
 36  collection_name         345521 non-null  object
dtypes: bool(2), datetime64[ns, UTC](2), float64(7), int64(8), object(18)
memory usage: 92.9+ MB
```

we drop columns with null or repeating data

[6]:
```
df = df.drop(['quantity', 'description', 'media_url', 'supply'], axis=1)
```

Here is the choice of collections to research for the Ocean Task:

[7]:
```
df.collection_name.value_counts()
```

[7]:
```
Otherdeed              175690
MutantApeYachtClub      60544
Azuki                   45734
BoredApeYachtClub       33483
Moonbirds               30070
Name: collection_name, dtype: int64
```

2 Choosing Collections

We choose Azuki and MoonBirds

[8]:
```
AzukiDf = df[df.collection_name=='Azuki']
MoonbirdsDf = df[df.collection_name=='Moonbirds']
```

3 First question: Analyze how the number of daily transactions for the collections has changed over time

Here are the graphs the daily usdc price for Azuki and Moonbirds

[9]:
```
fig = px.line(AzukiDf, x='timestamp', y="usd_price")
fig.show()
```

[10]:
```
fig = px.line(MoonbirdsDf, x='timestamp', y="usd_price")
fig.show()
```

Here are more granular graphs of the daily usdc price for Azuki and Moonbirds that aggregated with average function

[8]:
```
fig = px.histogram(AzukiDf, x='timestamp', y="usd_price", histfunc="avg",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="D1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="D1",␣
 ↪tickformat="%b\n%Y")
fig.update_layout(bargap=0.1)
fig.show()
```

[11]:
```
fig = px.histogram(MoonbirdsDf, x='timestamp', y="usd_price", histfunc="avg",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="D1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="D1",␣
 ↪tickformat="%b\n%Y")
fig.update_layout(bargap=0.1)
fig.show()
```

We can see that the prices spiked initially but declined steadily afterwards. Azuki managed to bounce back eventually. However, Moonbirds continues to decline.

Now we present graphs of the daily usdc price volume for Azuki and Moonbirds:

[12]:
```
fig = px.histogram(AzukiDf, x='timestamp', y="usd_price", histfunc="sum",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="D1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="D1",␣
 ↪tickformat="%b\n%Y")
fig.update_layout(bargap=0.1)
fig.show()
```

[13]:
```
fig = px.histogram(MoonbirdsDf, x='timestamp', y="usd_price", histfunc="sum",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="D1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="D1",␣
 ↪tickformat="%b\n%Y")
```

```
fig.update_layout(bargap=0.1)
fig.show()
```

Similarly with daily volume can observe similar pattern with the prices spiked initially but declined steadily afterwards. Both Azuki and Moonbirds managed to bounce back eventuall, Azuki at a higher rate.

4 Provide a visual overview of the NFT collections of your choice and its characteristics (e.g. size, type of NFTs, date range)?

[9]:
```
fig = px.histogram(AzukiDf, x='timestamp', y="usd_price", histfunc="avg",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="M1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="M1",␣
 ↪tickformat="%b\n%Y")
fig.update_layout(bargap=0.1)
#fig.add_trace(go.Scatter(mode="markers", x=AzukiDf["timestamp"],␣
 ↪y=AzukiDf["usd_price"], name="daily"))
fig.show()
```

[10]:
```
fig = px.histogram(AzukiDf, x='timestamp', y="usd_price", histfunc="sum",␣
 ↪title="Histogram on Date Axes")
fig.update_traces(xbins_size="M1")
fig.update_xaxes(showgrid=True, ticklabelmode="period", dtick="M1",␣
 ↪tickformat="%b\n%Y")
fig.update_layout(bargap=0.15)
fig.show()
```

[11]:
```
def convert_properties(x):
    bla = str(x)[1:-1].split(",")

    eyes = ""
    face = ""
    ear = ""
    hair = ""
    type_ = ""
    offhand = ""
    clothing = ""
    headgear = ""
    background = ""
    neck = ""
    special = ""
    mouth = ""

    for x in bla:
        a = x.split(":")
```

```
        if (a[0] == '"Eyes"'):
            eyes = a[1]
        elif (a[0] == '"Face"'):
            face = a[1]
        elif (a[0] == '"Ear"'):
            ear = a[1]
        elif (a[0] == '"Hair"'):
            hair = a[1]
        elif (a[0] == '"Type"'):
            type_ = a[1]
        elif (a[0] == '"Offhand"'):
            offhand = a[1]
        elif (a[0] == '"Clothing"'):
            clothing = a[1]
        elif (a[0] == '"Headgear"'):
            headgear = a[1]
        elif (a[0] == '"Background"'):
            background = a[1]
        elif (a[0] == '"Neck"'):
            neck = a[1]
        elif (a[0] == '"Special"'):
            special = a[1]
        elif (a[0] == '"Mouth"'):
            mouth = a[1]
        else:
            print(a[0])

    return pd.Series([eyes, face, ear, hair, type_, offhand, clothing,␣
↪headgear, background, neck, special])
```

[12]:
```
AzukiDf.flattened_properties.apply(lambda x: convert_properties(x))
```

[12]:
```
                  0            1  2                    3       4   \
269717     "Pensive"                    "Indigo Fluffy"  "Human"
269718  "Daydreaming"                       "Maroon Bun"  "Human"
269719      "Closed"  "Eye Patch"          "Black Bangs"  "Human"
269720       "Bored"    "Bandaid"         "Blonde Messy"  "Human"
269721       "White"                   "Brown Ponytail"  "Human"
…              …          … ..                 …      …
315446        "Ruby"                 "Black Teal Bangs"  "Human"
315447      "Closed"    "Bandaid"    "Brown Blonde Long"  "Human"
315448      "Joyful"                      "Blonde Flowy"  "Human"
315449      "Closed"  "Eye Patch"          "Dreadlocks"  "Human"
315450     "Focused"                     "Silver Long"  "Human"

                  5                    6              7             8   \
269717                           "Suikan"                "Off White C"
```

```
269718  "Fishing Rod"  "Blue Kimono with Bow"                  "Off White A"
269719         "Fan"      "Red Panda Hoodie"     "Sandogasa"  "Off White A"
269720      "Banner"     "Azuki Tech Jacket"                  "Off White D"
269721      "Banner"       "Hoodie with Bag"                  "Off White B"
…              …                     …             …            …
315446                "Plated Samurai Armor"                  "Off White D"
315447      "Katana"         "White T-Shirt"                  "Off White A"
315448      "Shinai"         "Red Ninja Top"                         "Red"
315449         "Fan"           "Frog Kimono"                        "Red"
315450                       "Kung Fu Shirt"  "Cat Headband"         "Red"

            9  10
269717
269718
269719  "Towel"
269720
269721
…          … ..
315446
315447
315448
315449
315450

[45734 rows x 11 columns]
```

[13]:
```
AzukiDf[["eyes", "face", "ear", "hair", "type", "offhand", "clothing",␣
↪"headgear", "background", "neck", "special"]] = AzukiDf.flattened_properties.
↪apply(lambda x : convert_properties(x))
```

```
/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/tmp/ipykernel_15367/2564803635.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

[14]:

```
AzukiDf['eyes'].value_counts()
```

[14]:

```
"Closed"            7581
"Determined"        3397
"Concerned"         2471
"Striking"          2322
"Calm"              2271
"Amethyst"          2236
"Daydreaming"       2160
"Joyful"            2079
"Hopeful"           1908
"Ruby"              1862
```

```
"Relaxed"            1817
"Pierced Eyebrow"    1403
"Tired"              1382
"Chill"              1364
"Careless"           1356
"Curious"            1336
"Indifferent"        1271
"Focused"            1201
"Bored"              1150
"Meditating"         1134
"Suspicious"         1089
"Pensive"            1069
"White"               874
"Red"                 525
"Glowing"             203
"Fire"                158
"Lightning"           115
Name: eyes, dtype: int64
```

[15]:
```
AzukiDf['face'].value_counts()
```

[15]:
```
                          31440
"Eye Scar"                 1251
"Red Fang Face Paint"      1218
"Bandaid"                  1207
"Red Stripes Face Paint"   1189
"Sleep Mask"               1110
"Eye Patch"                1110
"Round Blue Sunglasses"    1078
"Reading Glasses"           959
"Black Glasses"             904
"Blue Sunglasses"           600
"Kabuki Facepaint"          560
"Clear Glasses"             537
"Blush"                     507
"Seer Eyeband"              456
"Ji Eyeband"                369
"Round Purple Sunglasses"   350
"Red Bandana"               324
"Heart Eye Patch"           303
"Lipstick Kiss"             262
Name: face, dtype: int64
```

[16]:
```
AzukiDf.corr()
```

[16]:
```
                 __confirmed  __block_number  block_number  log_index  \
__confirmed         1.000000       -0.014375     -0.014375  -0.005960
```

__block_number	-0.014375	1.000000	1.000000	-0.098707
block_number	-0.014375	1.000000	1.000000	-0.098707
log_index	-0.005960	-0.098707	-0.098707	1.000000
token_id	-0.001917	0.025088	0.025088	-0.005156
is_multi_token_sale	0.000537	-0.049224	-0.049224	-0.022478
multi_token_sale_index	0.000392	-0.040186	-0.040186	-0.019662
usd_price	0.000615	-0.032883	-0.032883	0.064850
eth_price	-0.001376	0.281627	0.281627	0.015928
native_price	-0.001376	0.281627	0.281627	0.015928
royalty_fee	0.005065	-0.290794	-0.290794	0.087900
platform_fee	0.006024	-0.351726	-0.351726	0.090660
__updated_block_number	-0.004557	0.019177	0.019177	0.005082

	token_id	is_multi_token_sale	multi_token_sale_index	\
__confirmed	-0.001917	0.000537	0.000392	
__block_number	0.025088	-0.049224	-0.040186	
block_number	0.025088	-0.049224	-0.040186	
log_index	-0.005156	-0.022478	-0.019662	
token_id	1.000000	-0.004976	-0.005889	
is_multi_token_sale	-0.004976	1.000000	0.730591	
multi_token_sale_index	-0.005889	0.730591	1.000000	
usd_price	-0.011151	0.000560	-0.000164	
eth_price	-0.002958	-0.024150	-0.018674	
native_price	-0.002958	-0.024150	-0.018674	
royalty_fee	-0.013852	-0.003276	-0.000237	
platform_fee	-0.015638	0.001867	0.003480	
__updated_block_number	0.454213	-0.014565	-0.009819	

	usd_price	eth_price	native_price	royalty_fee	\
__confirmed	0.000615	-0.001376	-0.001376	0.005065	
__block_number	-0.032883	0.281627	0.281627	-0.290794	
block_number	-0.032883	0.281627	0.281627	-0.290794	
log_index	0.064850	0.015928	0.015928	0.087900	
token_id	-0.011151	-0.002958	-0.002958	-0.013852	
is_multi_token_sale	0.000560	-0.024150	-0.024150	-0.003276	
multi_token_sale_index	-0.000164	-0.018674	-0.018674	-0.000237	
usd_price	1.000000	0.890594	0.890594	0.848330	
eth_price	0.890594	1.000000	1.000000	0.659913	
native_price	0.890594	1.000000	1.000000	0.659913	
royalty_fee	0.848330	0.659913	0.659913	1.000000	
platform_fee	0.785360	0.602075	0.602075	0.916108	
__updated_block_number	0.009356	0.017379	0.017379	0.008318	

	platform_fee	__updated_block_number
__confirmed	0.006024	-0.004557
__block_number	-0.351726	0.019177
block_number	-0.351726	0.019177

```
log_index                 0.090660              0.005082
token_id                 -0.015638              0.454213
is_multi_token_sale       0.001867             -0.014565
multi_token_sale_index    0.003480             -0.009819
usd_price                 0.785360              0.009356
eth_price                 0.602075              0.017379
native_price              0.602075              0.017379
royalty_fee               0.916108              0.008318
platform_fee              1.000000              0.006022
__updated_block_number    0.006022              1.000000
```

[ ]:

[ ]: