```
!pip install yfinance
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: yfinance in /usr/local/lib/python3.7/dist-packages (0.1.85)
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist-packages (from yfinance) (1.3.5)
Requirement already satisfied: appdirs>=1.4.4 in /usr/local/lib/python3.7/dist-packages (from yfinance) (1.4.4)
Requirement already satisfied: requests>=2.26 in /usr/local/lib/python3.7/dist-packages (from yfinance) (2.28.1)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packages (from yfinance) (1.21.6)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.7/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.5.1 in /usr/local/lib/python3.7/dist-packages (from yfinance) (4.9.1)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.0->yfinance) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.0->yfinance) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.7.3->pandas>=0.24.0->yfinance) (1.15.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests>=2.26->yfinance) (1.24.3)
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages (from requests>=2.26->yfinance) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests>=2.26->yfinance) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests>=2.26->yfinance) (2022.9.24)
```

```
import pandas as pd
import numpy as np
import yfinance as yf
from datetime import datetime, timedelta

def options_chain(symbol):

    tk = yf.Ticker(symbol)

    # Expiration dates
    exps = tk.options

    # Get options for each expiration
    options = pd.DataFrame()
    for e in exps:
        opt = tk.option_chain(e)
        opt = pd.DataFrame().append(opt.calls).append(opt.puts)
        opt['expirationDate'] = e
        options = options.append(opt, ignore_index=True)

    # Bizarre error in yfinance that gives the wrong expiration date
    # Add 1 day to get the correct expiration date
    options['expirationDate'] = pd.to_datetime(options['expirationDate']) + timedelta(days = 1)
    options['dte'] = (options['expirationDate'] - datetime.today()).dt.days / 365

    # Boolean column if the option is a CALL
    options['CALL'] = options['contractSymbol'].str[4:].apply(
        lambda x: "C" in x)

    options[['bid', 'ask', 'strike']] = options[['bid', 'ask', 'strike']].apply(pd.to_numeric)
    options['mark'] = (options['bid'] + options['ask']) / 2 # Calculate the midpoint of the bid-ask

    # Drop unnecessary and meaningless columns
    options = options.drop(columns = ['contractSize', 'currency', 'change', 'percentChange', 'lastTradeDate', 'lastPrice'])
```

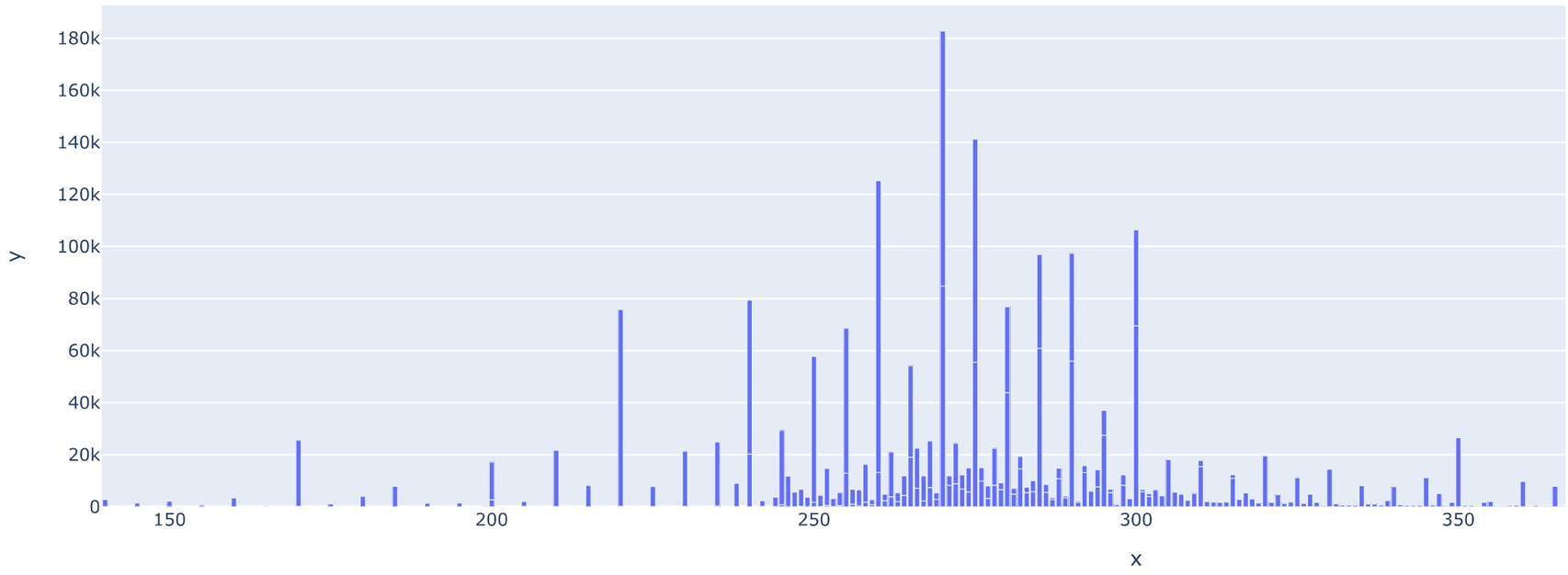
```
34
```

```
1 ticker = 'QQQ'
```

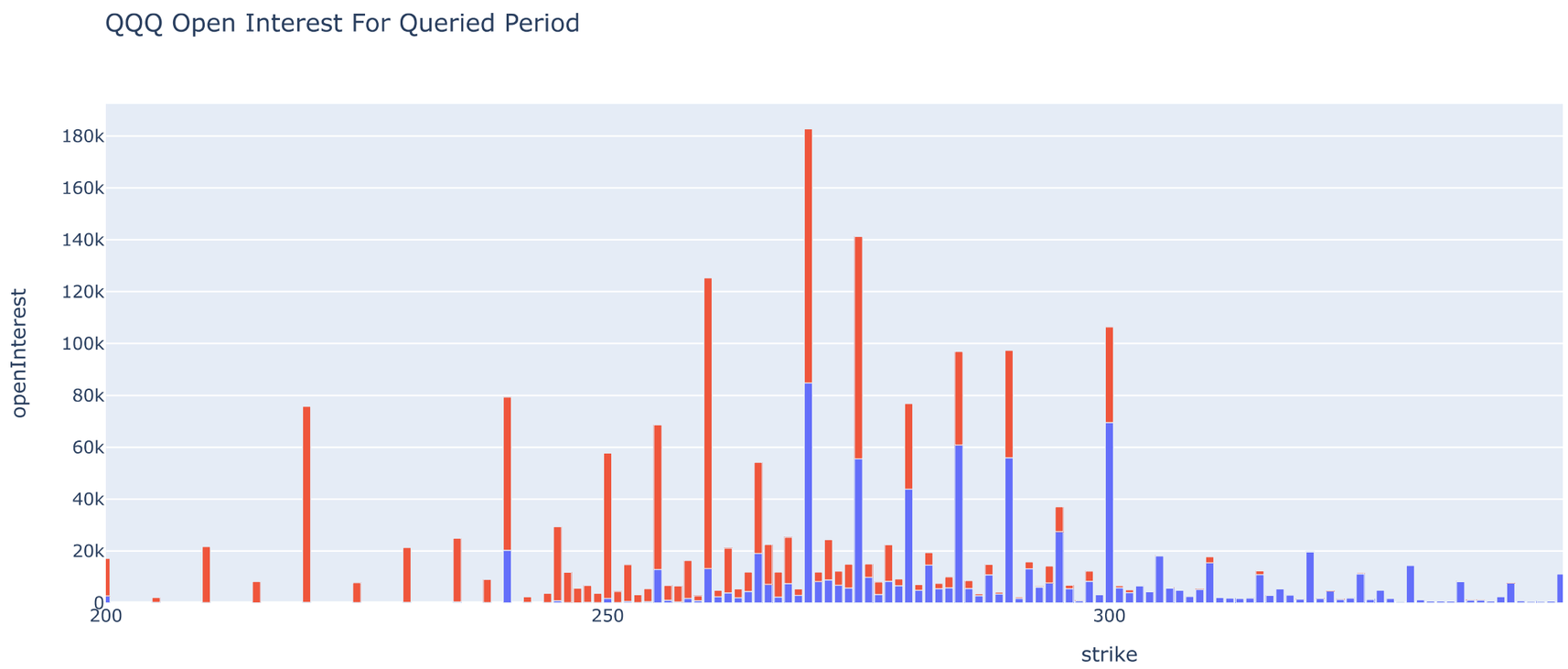
```
1 options_info = pd.DataFrame(options_chain(ticker))
2 date_set = []
3 date_set = set(options_info['expirationDate'])
4 date_set
```

```
1 time_frame = '2022-11-19'
```

```
1 import plotly.express as px
2
3 options_info_t = options_info
4 options_info_t = options_info[options_info.expirationDate == time_frame]
5 # options_info_t = options_info_t[options_info_t.CALL != True]
6
7 fig = px.bar(x = options_info_t['strike'],y=options_info_t['openInterest'],title='Put Open Interest')
8 # fig.update_xaxes(range=[400,480])
9 # fig.update_xaxes(range=[330,420])
10 fig.show()
```



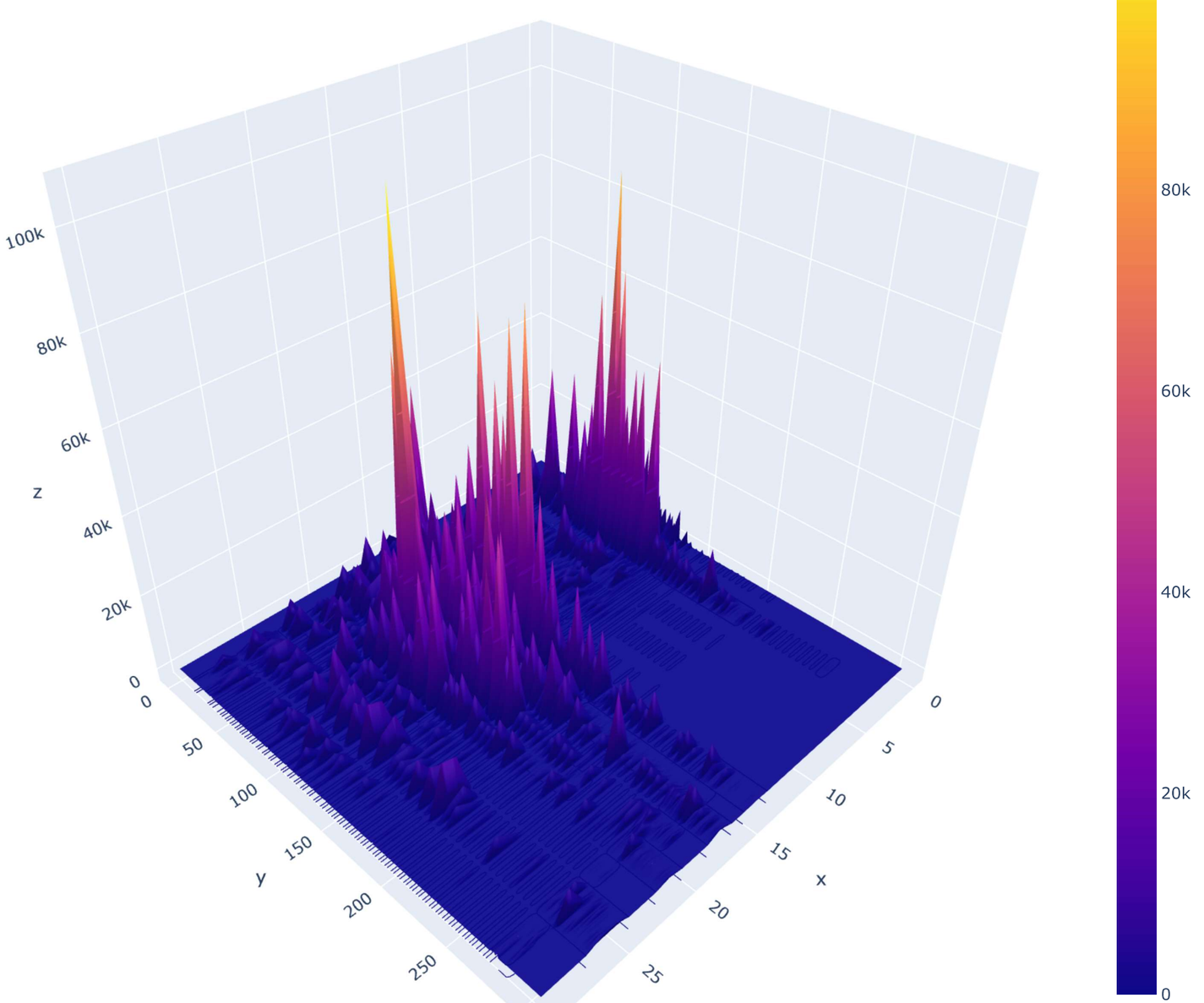
```
import plotly.express as px
options_info_t = options_info[options_info.expirationDate == time_frame]
openInterest_plot = px.bar(options_info_t,x = 'strike',y='openInterest',title='QQQ Open Interest For Queried Period', color="CALL")
openInterest_plot.update_xaxes(range=[200,400])
openInterest_plot.show()
```




```
options_pivot_table1 = pd.pivot_table(data=options_info,values='openInterest',index='strike',columns='expirationDate',fill_value=0)
```

```
import plotly.graph_objects as go

import pandas as pd

#Read data from a csv
#z_data = pd.read_csv('/Options Volatility Surface Data.csv')
#z_data=z_data.set_index('Unnamed: 0')
z_data=options_pivot_table1

fig = go.Figure(data=[go.Surface(z=z_data.values)])
fig.update_traces(contours_z=dict(show=True, usecolormap=True,
                                  highlightcolor="limegreen", project_z=True))
fig.update_layout(title='Open Interest', autosize=False,width=1000, height=1000)

fig.show()
```

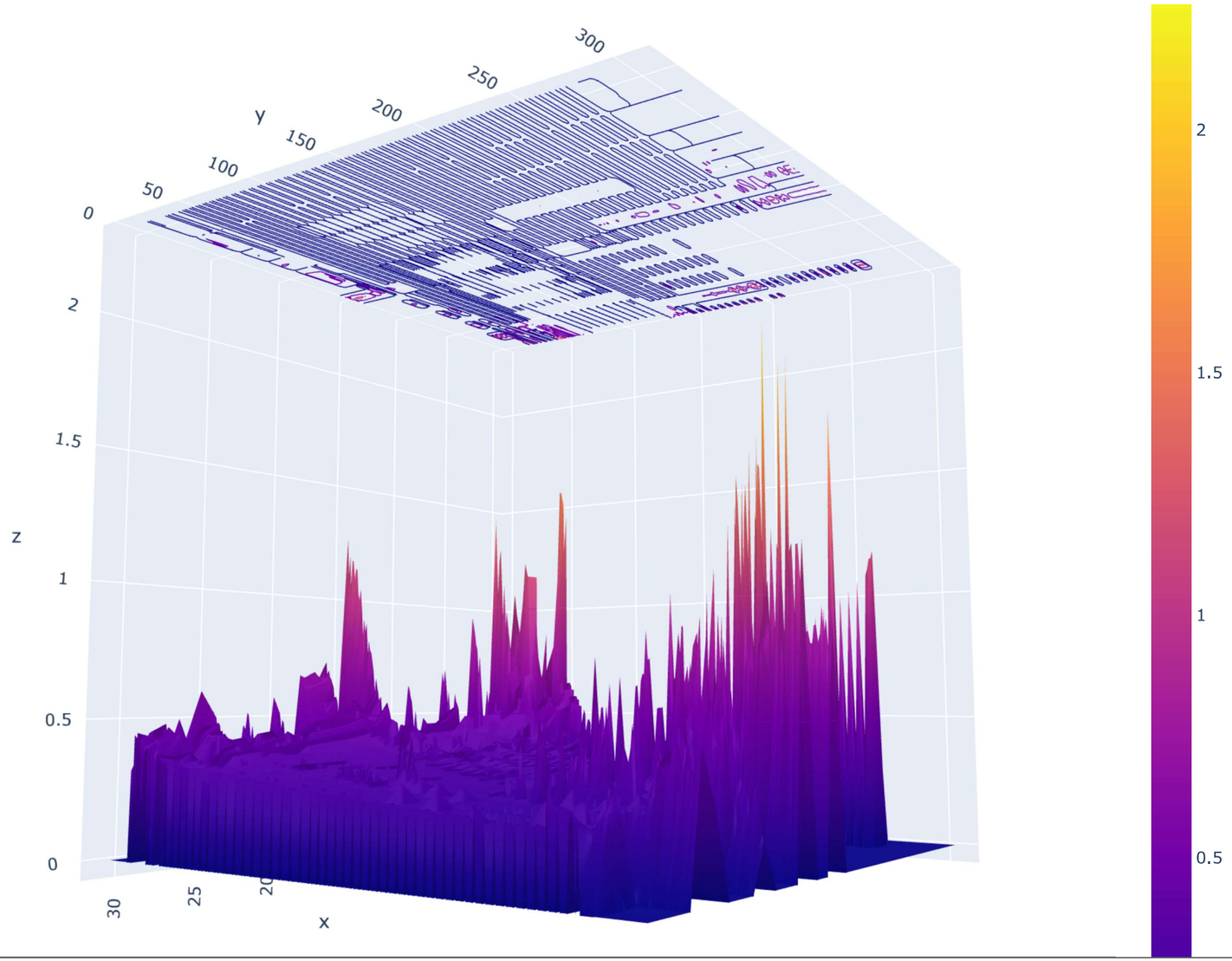
Open Interest



```
options_pivot_table2 = pd.pivot_table(data=options_info,values='impliedVolatility',index='strike',columns='expirationDate',fill_value=0)
```

```
import plotly.graph_objects as go

import pandas as pd

# Read data from a csv
# z_data = pd.read_csv('/Options Volatility Surface Data.csv')
# z_data=z_data.set_index('Unnamed: 0')
z_data=options_pivot_table2

fig = go.Figure(data=[go.Surface(z=z_data.values)])
fig.update_traces(contours_z=dict(show=True, usecolormap=True,
                                  highlightcolor="limegreen", project_z=True))
fig.update_layout(title='Implied Volatility', autosize=False,width=1000, height=1000)

fig.show()
```

Implied Volatility



✓ 0s completed at 1:22 PM