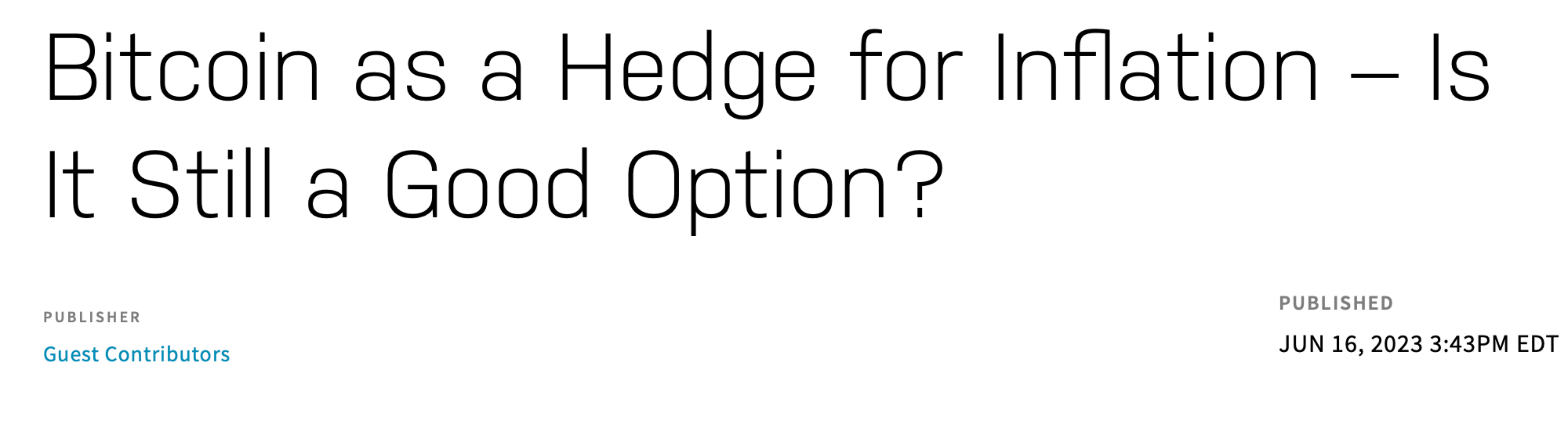
Building ETL Pipelines 2



Bitcoin as a Hedge for Inflation – Is It Still a Good Option?

PUBLISHER

Guest Contributors

PUBLISHED

JUN 16, 2023 3:43PM EDT

https://www.nasdaq.com/articles/bitcoin-as-a-hedge-for-inflation-is-it-still-a-good-option#:~:text=Bitcoin has potential as an,add a level of risk.

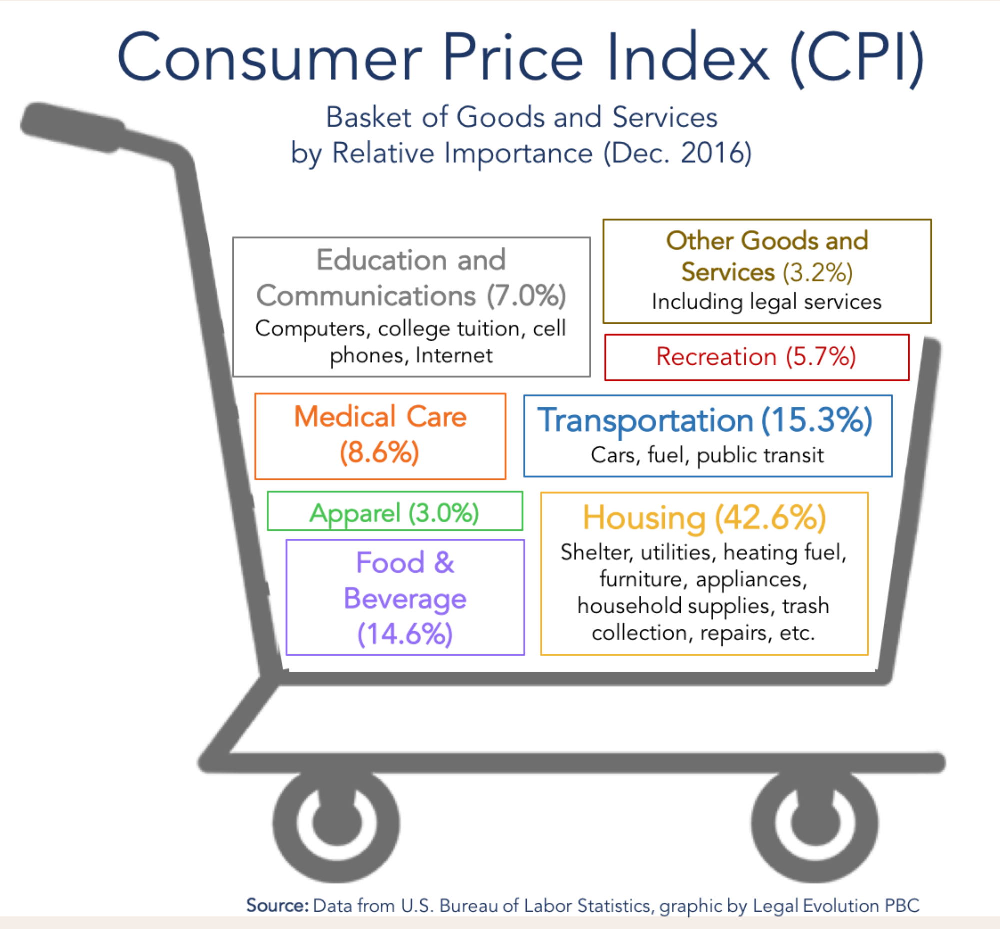
Bitcoin as a hedge for inflation?

Build data pipeline:

- **Bitcoin price**
- **Inflation data**
  - Consumer Price Index (CPI)
  - Global Price Index of all commodities (GPI)

Compare bitcoin price with the economic indicators

Price Index



Consumer Price Index (CPI)

Basket of Goods and Services
by Relative Importance (Dec. 2016)

Education and Communications (7.0%)
Computers, college tuition, cell phones, Internet
Other Goods and Services (3.2%)
Including legal services
Recreation (5.7%)
Medical Care (8.6%)
Transportation (15.3%)
Cars, fuel, public transit
Apparel (3.0%)
Food & Beverage (14.6%)
Housing (42.6%)
Shelter, utilities, heating fuel, furniture, appliances, household supplies, trash collection, repairs, etc.

Source: Data from U.S. Bureau of Labor Statistics, graphic by Legal Evolution PBC

ETL development process

1. **Design the database schema**
2. **Extract data from the source**
    - Identify endpoint and query parameters from API documentation
    - Request and get response from API
3. **Transform the data**
    - Select and clean the relevant values from the API response
    - Prepare the data for database loading
4. **Load the data into the database**
    - Create the table(s) based on the schema
    - Insert the transformed data into the table(s)

Design the database schema

Create additional tables in the `coins.db` database

Table: `cpi`

Column	Type
date	TEXT
cpi	NUMERIC

Table: `gpi`

Column	Type
date	TEXT
gpi	NUMERIC

🖥️ Extract data from the source

https://www.alphavantage.co/documentation/#economic-indicators

Extract CPI and GPI from Alpha Vantage

- Endpoint?
- Query parameters?

`extract_indicators()`

```python
def extract_indicators(function, apikey):
    """Extract economic indicator data from Alpha Vantage API

    Args:
        function (str): The function parameter for the economic indicator to extract (e.g., 'CPI')
        apikey (str): Your Alpha Vantage API key
    Returns:
        dict: The JSON response from the API
    """
    ...
```

transform_indicators()

```python
def transform_indicators(response):
    """Transform the API response into a list of tuples
    Args:
        response (dict): The JSON response from the API
    Returns:
        list: A list of tuples containing (date, value)
    """
    ...
```

load_indicators()

```python
def load_indicators(conn, table, data):
    """Load the transformed data into the database
    Args:
        conn (sqlite3.Connection): The database connection
        table (str): The name of the table to load data into
        data (list): A list of tuples containing (date, value)
    """
    ...
```

`indicators_etl()`

```python
def indicators_etl(conn, table, function, apikey):
    """ETL process for economic indicators
    Args:
        conn (sqlite3.Connection): The database connection
        table (str): The name of the table to load data into
        function (str): The function parameter for the economic indicator to extract (e.g., 'CPI')
        apikey (str): Your Alpha Vantage API key

    Returns:
        None
    """
    ...
```

Read Bitcoin price, CPI, and GPI from the database and create DataFrames

1. Connect to the `coins.db` database
2. Read data from the `coins`, `cpi`, and `gpi` tables into Pandas DataFrames

Calculate monthly returns

Aggregation:

- Aggregate daily data to monthly data
- Aggregate monthly data to get return rates

Handling missing values:

- How to handle missing values?

More aggregate functions

- `last` (`first`): last (first) value in a group
- `nth`: nth value in a group
- `diff`: difference from the previous value
- `pct_change`: percentage change from the previous value
- `nunique`: number of unique values in a group

  …

---

https://pandas.pydata.org/pandas-docs/stable/reference/groupby.html#aggregation

Aggregate daily Bitcoin price to monthly

Calculate monthly returns of Bitcoin, CPI, and GPI

GPI data: `"."` is not a number

	date	gpi
0	1992-01-01	.
1	1992-02-01	.
2	1992-03-01	.
3	1992-04-01	.
4	1992-05-01	.
...	...	...
376	2023-05-01	157.134002
377	2023-06-01	154.069142
378	2023-07-01	157.908799

```
# error
# can't calculate pct_change with a dot
rate = gpi_df.agg({"gpi": "pct_change"})

# gpi data type is object, not float
gpi_df['gpi'].dtype
# dtype('O')

cpi_df['cpi'].dtype
# dtype('float64')
```

Convert GPI column to numeric

`pd.to_numeric()` : convert argument to a numeric type

```
print(gpi_df['gpi'].dtype)      # dtype('O')

# Convert gpi column to numeric
# errors='coerce' replaces non-numeric values with missing value (NaN)
gpi_df['gpi']=pd.to_numeric(gpi_df['gpi'], errors='coerce')

print(gpi_df['gpi'].dtype)      # dtype('float64')
```

Data types for missing values

Types:

- `None`: Python's built-in missing value
- `pd.NA`: Pandas's missing value
- `np.nan`: Numpy's missing value

Advantages:

- Compatible with numerical data types
- Numerical operations supported
- Easy missing value detection and handling

```
df = pd.DataFrame({
    'a': [1, 2, None],
    'b': [1, 2, pd.NA],
    'c': [1, 2, np.nan]
})
# outputs
#     a     b    c
# 0  1.0  1    1.0
# 1  2.0  2    2.0
# 2  NaN  <NA> NaN

df['a'].dtype
# dtype('float64')
```

GPI data after replacing `.` with missing value

	date	gpi
0	1992-01-01	NaN
1	1992-02-01	NaN
2	1992-03-01	NaN
3	1992-04-01	NaN
4	1992-05-01	NaN
...	...	...
376	2023-05-01	157.134002
377	2023-06-01	154.069142
378	2023-07-01	157.908799

Handling missing data

- `isna()`: returns `True` if the value is missing, `False` otherwise
- `notna()`: returns `True` if the value is not missing, `False` otherwise
- `dropna()`: **drop rows with missing data**

```
# Count missing data in each column
gpi_df.isna().agg('sum')

# Count non-missing data in each column
gpi_df.notna().agg('sum')

# Drop rows with missing data and assign to gpi_df
gpi_df = gpi_df.dropna()

# Drop rows with missing data and assign to gpi_df
gpi_df = gpi_df.dropna(subset=["gpi"])
```

Missing data imputation

Year	Firm ID	Stock Price	Revenue	Earnings	Total Assets
2015	XYZ	85.50	1000	120	5000
2016	XYZ	90.00	1050	NaN	5200
2017	XYZ	NaN	1075	125	NaN
2018	XYZ	NaN	1100	130	5400
2019	XYZ	80.25	1150	NaN	5600
2020	XYZ	100.00	NaN	140	5800

Missing data imputation

```
# Fill missing data with 0
gpi_df = gpi_df.fillna(0)

# Fill missing data with the previous value (forward fill)
gpi_df = gpi_df.fillna(method="ffill")

# Fill missing data with the next value (backward fill)
gpi_df = gpi_df.fillna(method="bfill")

# Fill missing data with linear interpolation
gpi_df = gpi_df.interpolate(method="linear")
```

`merge` Bitcoin price and economic indicators

Chaining Pandas methods

```
df = coins_monthly_df.merge(cpi_df, on="month").merge(gpi_df, on="month").dropna()[cols]
```

```
df = (
    coins_monthly_df                  # coins_monthly_df
    .merge(cpi_df, on="month")  # merge with cpi_df
    .merge(gpi_df, on="month")  # merge with gpi_df
    .dropna()                         # drop rows with missing data
    [cols]                            # select columns
)
```

Is Bitcoin a hedge for inflation?

- Visualization
- Correlation
- Regression
- Ask ChatGPT

Plotly Express Syntax

```python
import plotly.express as px

fig = px.scatter(df, x="age", y="height")
fig.show()
```

https://plotly.com/python/plotly-express/
https://plotly.com/python/px-arguments/

Line plot

```
# line plot for monthly return
fig = px.line(df, x="month", y="return")
fig.show()
```

```
# line plot for monthly return and inflation rate
fig = px.line(df, x="month", y=["return", "cpi_change"])
fig.show()
```

Heatmap for correlation

```
# correlation matrix
corr = df.corr()

# heatmap
fig = px.imshow(corr, color_continuous_scale="Redor")
fig.show()
```

---

color scale: https://plotly.com/python/builtin-colorscales/

Scatter plot with regression line

```
fig = px.scatter(
    df,
    x="cpi_change",
    y="return",
    trendline="ols"
)
fig.show()
```

Regression using `statsmodels`

```
import statsmodels.api as sm

X = df["cpi_change"]
y = df["return"]
X = sm.add_constant(X)
model = sm.OLS(y, X).fit()
model.summary()
```

https://www.statsmodels.org/stable/index.html

Ask ChatGPT

```
from openai import OpenAI

prompt = f"""
    Is Bitcoin a good hedge for inflation? Use the following data to support your answer.
    bitoin returns: {bitcoin_returns}
    CPI rates: {inflation_rates}
    GPI rates: {gpi_rates}
    Provide a detailed analysis including correlation and regression results.
"""

client = OpenAI(api_key="your-api-key")
response = client.chat.completions.create(
    model="gpt-5-nano",
    messages=[
        {"role": "system", "content": "You are a data analyst."},
        {"role": "user", "content": prompt}
    ]
)
print(response.choices[0].message.content)
```