|  |  |
| --- | --- |
| **CAT 2 PROJECT** | **OSS ANALYTICS** |
| **20.02.2024** |

**AIM**

To leverage open-source software (OSS) insights for enhanced analytics, fostering data-driven decision-making among developers, contributors, and stakeholders in the OSS community.

**PROBLEM STATEMENT**

The explosive growth of OSS has made it challenging to track contributions, issues, and trends effectively. Without detailed analytics, it's difficult to understand community engagement and the performance of various projects. This project seeks to address these challenges by systematically harvesting and analyzing OSS data.

**OBJECTIVE**

The objective of this project is to extract, process, and analyze data from the OSS Insight API, aiming to uncover trends, distribution, and other valuable metrics related to repositories, contributors, and issue creators. Insights such as top contributing countries, most used languages, and trending repositories can be derived to inform community and development strategies.

**WORKFLOW**

A diagram of a application

Description automatically generated

Data Fetching: Utilize the OSS Insight API to extract data related to trending repositories, collections, and contributors' geographical distribution.

Data Ingestion: Insert the retrieved data into Elasticsearch to enable complex searches and data retrieval.

Exploratory Data Analysis (EDA): Conduct EDA through statistical computations and visualizations to unwrap the underlying patterns and insights within the data.

Reporting: Using the Kibana to compile the analytical findings into comprehensible reports for stakeholders.

**CODE**

The provided code snippets depict the essential components of the EDA process, from fetching data using the API to visualizing it with libraries like pandas, matplotlib, and seaborn.

**EDA**

import pandas as pd  
 import matplotlib.pyplot as plt  
 import seaborn as sns  
 from ast import literal\_eval

import requests  
def fetch\_data\_from\_api(url):  
 headers = {'Accept': 'application/json'}  
 response = requests.get(url, headers=headers)  
 if response.status\_code == 200:  
 return response.json()  
 else:  
 print("Failed to fetch data:", response.status\_code)  
 return None

# Define a function to perform EDA  
def perform\_eda(json\_data):  
 # Load the JSON data into a pandas DataFrame  
 df = pd.DataFrame(json\_data['data']['rows'])  
  
 # Inspect the data  
 print(df.head()) # Print the first few rows  
 print(df.info()) # Get a concise summary of the DataFrame  
  
 # Clean the data dynamically based on data type inferred from the first row  
 for column in df.columns:  
 if pd.api.types.is\_numeric\_dtype(df[column]):  
 df[column] = pd.to\_numeric(df[column], errors='coerce').fillna(0)  
 else:  
 df[column] = df[column].fillna('Unknown').astype(str)  
  
 # Perform statistical analysis  
 print(df.describe(include='all')) # Get the summary statistics for all columns  
  
 # Visualizations and further analysis would go here  
 # ...  
  
 return df # Return the cleaned DataFrame for further use if needed

df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/trends/repos/")  
df = perform\_eda(df)

repo\_id repo\_name primary\_language \  
0 681372871 microsoft/garnet C#   
1 773286980 xai-org/grok-1 Python   
2 741790915 OpenInterpreter/01 Python   
3 775250190 stitionai/devika Python   
4 661954287 mshumer/gpt-prompt-engineer Jupyter Notebook   
  
 description stars forks \  
0 Garnet is a remote cache-store from Microsoft ... 1613 80   
1 Grok open release 1348 263   
2 The open-source language model computer 799 72   
3 Devika is an Agentic AI Software Engineer that... 782 111   
4 669 35   
  
 pull\_requests pushes total\_score \  
0 9 21 6210.0914   
1 4 5404.7856   
2 11 25 3651.9766   
3 7 4 3368.1381   
4 3 2866.1857   
  
 contributor\_logins collection\_names   
0 badrishc,TalZaccai,darrenge,yrajas,pradeepyada...   
1 SleepTheGod,menyunlei,yh-yao,chenyzh28,qinjian...   
2 KillianLucas,tyfiero,MikeBirdTech,Maclean-D,my...   
3 rohittp0,mufeedvh,123vivekr,NorbiXD,alfi4000   
4 mshumer,mattdaniell   
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 100 entries, 0 to 99  
Data columns (total 11 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 repo\_id 100 non-null object  
 1 repo\_name 100 non-null object  
 2 primary\_language 100 non-null object  
 3 description 100 non-null object  
 4 stars 100 non-null object  
 5 forks 100 non-null object  
 6 pull\_requests 100 non-null object  
 7 pushes 100 non-null object  
 8 total\_score 100 non-null object  
 9 contributor\_logins 100 non-null object  
 10 collection\_names 100 non-null object  
dtypes: object(11)  
memory usage: 8.7+ KB  
None  
 repo\_id repo\_name primary\_language description stars forks \  
count 100 100 100 100 100 100   
unique 100 100 18 96 81 41   
top 681372871 microsoft/garnet Python 62 0   
freq 1 1 35 5 5 11   
  
 pull\_requests pushes total\_score contributor\_logins collection\_names   
count 100 100 100 100 100   
unique 14 23 100 84 7   
top 6210.0914   
freq 51 38 1 17 94

# Trending repos  
  
# Distribution of repository stars  
plt.figure(figsize=(12, 6))  
sns.histplot(df['stars'], bins=10, kde=False)  
plt.title('Distribution of Stars in Trending Repositories')  
plt.xlabel('Stars')  
plt.ylabel('Number of Repositories')  
plt.show()  
  
# Distribution of repository forks  
plt.figure(figsize=(12, 6))  
sns.histplot(df['forks'], bins=10, kde=False)  
plt.title('Distribution of Forks in Trending Repositories')  
plt.xlabel('Forks')  
plt.ylabel('Number of Repositories')  
plt.show()  
  
# Top Primary Languages Used  
plt.figure(figsize=(12, 6))  
sns.countplot(data=df, y='primary\_language', order=df['primary\_language'].value\_counts().index)  
plt.title('Top Primary Languages in Trending Repositories')  
plt.xlabel('Number of Repositories')  
plt.ylabel('Primary Language')  
plt.show()  
  
# Top contributors  
top\_contributors = df.explode('contributor\_logins')['contributor\_logins'].value\_counts().head(10)  
  
plt.figure(figsize=(12, 6))  
sns.barplot(x=top\_contributors.values, y=top\_contributors.index)  
plt.title('Top 10 Contributors in Trending Repositories')  
plt.xlabel('Number of Contributions')  
plt.ylabel('Contributor Logins')  
plt.show()

A graph with blue bars

Description automatically generated with medium confidence

A graph of blue bars

Description automatically generated with medium confidence

A graph with blue bars

Description automatically generated with medium confidence

A graph of a bar graph

Description automatically generated

# Collections  
df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/collections/")  
df = perform\_eda(df)

id name  
0 1 Static Site Generator  
1 2 Open Source Database  
2 10001 CSS Framework  
3 10002 Google Analytics Alternative  
4 10003 Low Code Development Tool  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 90 entries, 0 to 89  
Data columns (total 2 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 id 90 non-null object  
 1 name 90 non-null object  
dtypes: object(2)  
memory usage: 1.5+ KB  
None  
 id name  
count 90 90  
unique 90 90  
top 1 Static Site Generator  
freq 1 1

# Step 5: Perform statistical analysis  
# For this dataset, since it's categorical, we'll just count the occurrences of each collection  
collections\_count = df['name'].value\_counts()  
  
# Step 6: Visualize the data  
# Example: Create a bar plot for the collection counts  
plt.figure(figsize=(14, 8))  
collections\_count.plot(kind='bar')  
plt.title('Frequency of Each Collection')  
plt.xlabel('Collection Name')  
plt.ylabel('Frequency')  
plt.xticks(rotation=90) # Rotate x-axis labels for better readability  
plt.show()

A graph with blue and black text

Description automatically generated

# Hot collection data  
df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/collections/hot/")  
df = perform\_eda(df)

id name repos repo\_id repo\_name \  
0 2 Open Source Database 39 208728772 milvus-io/milvus   
1 2 Open Source Database 39 60246359 clickhouse/clickhouse   
2 2 Open Source Database 39 138754790 duckdb/duckdb   
3 10005 Javascript Framework 32 10270250 facebook/react   
4 10005 Javascript Framework 32 11730342 vuejs/vue   
  
 repo\_current\_period\_rank repo\_past\_period\_rank repo\_rank\_changes   
0 1 5 4   
1 3 2 -1   
2 2 1 -1   
3 1 1 0   
4 3 4 1   
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 60 entries, 0 to 59  
Data columns (total 8 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 id 60 non-null object  
 1 name 60 non-null object  
 2 repos 60 non-null object  
 3 repo\_id 60 non-null object  
 4 repo\_name 60 non-null object  
 5 repo\_current\_period\_rank 60 non-null object  
 6 repo\_past\_period\_rank 60 non-null object  
 7 repo\_rank\_changes 60 non-null object  
dtypes: object(8)  
memory usage: 3.9+ KB  
None  
 id name repos repo\_id repo\_name \  
count 60 60 60 60 60   
unique 20 20 16 55 56   
top 2 Open Source Database 36 331293626 refinedev/refine   
freq 3 3 6 3 3   
  
 repo\_current\_period\_rank repo\_past\_period\_rank repo\_rank\_changes   
count 60 60 60   
unique 3 8 7   
top 1 1 0   
freq 20 19 31

# Visualization 1: Bar Plot of Collections by Number of Repos  
plt.figure(figsize=(14, 7))  
collections\_by\_repos = df.groupby('name')['repos'].sum().sort\_values(ascending=False)  
sns.barplot(x=collections\_by\_repos.values, y=collections\_by\_repos.index)  
plt.title('Number of Repositories in Each Collection')  
plt.xlabel('Total Number of Repositories')  
plt.ylabel('Collection Name')  
plt.show()  
  
# Visualization 2: Histogram of Current and Past Period Ranks  
plt.figure(figsize=(14, 7))  
sns.histplot(df['repo\_current\_period\_rank'], color='blue', label='Current Period Rank', kde=True)  
sns.histplot(df['repo\_past\_period\_rank'], color='orange', label='Past Period Rank', kde=True)  
plt.title('Distribution of Repository Ranks')  
plt.xlabel('Rank')  
plt.ylabel('Frequency')  
plt.legend()  
plt.show()  
  
# Visualization 3: Bar Plot of Top Repositories by Rank Changes  
plt.figure(figsize=(14, 7))  
top\_rank\_changes = df[df['repo\_rank\_changes'] != 0].sort\_values(by='repo\_rank\_changes', ascending=False).head(10)  
sns.barplot(x='repo\_rank\_changes', y='repo\_name', data=top\_rank\_changes, palette='viridis')  
plt.title('Top 10 Repositories by Rank Changes')  
plt.xlabel('Rank Changes')  
plt.ylabel('Repository Name')  
plt.show()  
  
# Visualization 4: Scatter Plot of Current vs. Past Ranks  
plt.figure(figsize=(10, 6))  
sns.scatterplot(data=df, x='repo\_past\_period\_rank', y='repo\_current\_period\_rank', hue='name', style='name', s=100)  
plt.title('Comparison of Current vs. Past Period Ranks')  
plt.xlabel('Past Period Rank')  
plt.ylabel('Current Period Rank')  
plt.legend(title='Collection Name', bbox\_to\_anchor=(1.05, 1), loc='upper left')  
plt.grid(True)  
plt.show()

A graph showing a bar graph

Description automatically generated

A graph of a bar graph

Description automatically generated

/tmp/ipykernel\_5441/1816133568.py:23: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.barplot(x='repo\_rank\_changes', y='repo\_name', data=top\_rank\_changes, palette='viridis')

A graph showing a graph of a graph

Description automatically generated with medium confidence

A graph with colored dots

Description automatically generated

# Issue creator   
df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/repos/pingcap/tidb/issue\_creators/")  
df = perform\_eda(df)

id login name issues first\_issue\_opened\_at  
0 3427324 hawkingrei Weizhen Wang 593 2021-05-28 09:41:34  
1 18818196 tisonkun tison 407 2021-02-20 02:02:25  
2 1420062 tiancaiamao 389 2016-07-01 13:40:58  
3 4242506 zimulala Lynn 360 2015-09-23 12:40:05  
4 17380469 wjhuang2016 wjHuang 306 2019-03-12 13:44:15  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 30 entries, 0 to 29  
Data columns (total 5 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 id 30 non-null object  
 1 login 30 non-null object  
 2 name 30 non-null object  
 3 issues 30 non-null object  
 4 first\_issue\_opened\_at 30 non-null object  
dtypes: object(5)  
memory usage: 1.3+ KB  
None  
 id login name issues first\_issue\_opened\_at  
count 30 30 30 30 30  
unique 30 30 24 28 30  
top 3427324 hawkingrei 254 2021-05-28 09:41:34  
freq 1 1 7 2 1

# Convert 'issues' to integer and 'first\_issue\_opened\_at' to datetime  
df['issues'] = df['issues'].astype(int)  
df['first\_issue\_opened\_at'] = pd.to\_datetime(df['first\_issue\_opened\_at'])  
  
# Visualization 1: Bar Plot of Top Contributors by Issue Count  
plt.figure(figsize=(12, 8))  
top\_contributors = df.nlargest(10, 'issues')  
sns.barplot(x='issues', y='login', data=top\_contributors, palette='viridis')  
plt.title('Top 10 Contributors by Issue Count')  
plt.xlabel('Issue Count')  
plt.ylabel('User Login')  
plt.show()  
  
# Visualization 2: Timeline of First Issue Opened Date for Each User  
plt.figure(figsize=(12, 8))  
df = df.sort\_values('first\_issue\_opened\_at')  
sns.scatterplot(data=df, x='first\_issue\_opened\_at', y='login', size='issues', hue='issues', palette='coolwarm', legend=False)  
plt.title('Timeline of First Issue Opened by Each User')  
plt.xlabel('Date of First Issue Opened')  
plt.ylabel('User Login')  
plt.xticks(rotation=45)  
plt.tight\_layout()  
plt.show()

/tmp/ipykernel\_5441/2959188873.py:8: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.barplot(x='issues', y='login', data=top\_contributors, palette='viridis')

A graph of a bar chart

Description automatically generated with medium confidence

A graph with blue dots

Description automatically generated

# Countries/Regions of Issue Creators  
df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/repos/pingcap/tidb/issue\_creators/countries/")  
df = perform\_eda(df)

country\_code issue\_creators percentage  
0 CN 455 0.7712  
1 US 39 0.0661  
2 IN 15 0.0254  
3 CA 8 0.0136  
4 SG 7 0.0119  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 32 entries, 0 to 31  
Data columns (total 3 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 country\_code 32 non-null object  
 1 issue\_creators 32 non-null object  
 2 percentage 32 non-null object  
dtypes: object(3)  
memory usage: 896.0+ bytes  
None  
 country\_code issue\_creators percentage  
count 32 32 32  
unique 32 10 10  
top CN 1 0.0017  
freq 1 12 12

# Convert 'issue\_creators' to integer and 'percentage' to float  
df['issue\_creators'] = df['issue\_creators'].astype(int)  
df['percentage'] = df['percentage'].astype(float)  
  
# Visualization 1: Bar Plot of Issue Creators by Country  
plt.figure(figsize=(14, 8))  
sns.barplot(x='country\_code', y='issue\_creators', data=df, palette='viridis')  
plt.title('Number of Issue Creators by Country')  
plt.xlabel('Country Code')  
plt.ylabel('Number of Issue Creators')  
plt.xticks(rotation=90)  
plt.show()  
  
# Visualization 2: Pie Chart of Issue Creator Percentages by Country  
plt.figure(figsize=(10, 10))  
df\_sorted = df.sort\_values('percentage', ascending=False)  
plt.pie(df\_sorted['percentage'], labels=df\_sorted['country\_code'], autopct='%1.1f%%', startangle=140)  
plt.title('Issue Creator Percentages by Country')  
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.  
plt.show()

/tmp/ipykernel\_5441/2510227193.py:7: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.barplot(x='country\_code', y='issue\_creators', data=df, palette='viridis')

A graph with numbers and a line

Description automatically generated with medium confidence

A pie chart with numbers and a number

Description automatically generated

# Organizations of Star Gazers  
df = fetch\_data\_from\_api("https://api.ossinsight.io/v1/repos/pingcap/tidb/issue\_creators/organizations/")  
df = perform\_eda(df)

issue\_creators org\_name percentage  
0 116 pingcap 0.2755  
1 10 tencent 0.0238  
2 7 bytedance 0.0166  
3 7 alibaba 0.0166  
4 3 linkedin 0.0071  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 263 entries, 0 to 262  
Data columns (total 3 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 issue\_creators 263 non-null object  
 1 org\_name 263 non-null object  
 2 percentage 263 non-null object  
dtypes: object(3)  
memory usage: 6.3+ KB  
None  
 issue\_creators org\_name percentage  
count 263 263 263  
unique 6 263 6  
top 1 pingcap 0.0024  
freq 239 1 23

**OUTPUT**

The output consists of a series of visualizations representing various analytical dimensions, such as the distribution of stars and forks in trending repositories, primary languages used, as well as geographical distribution and temporal analysis of issues.

A screenshot of a computer

Description automatically generated

**CONCLUSION**

The project demonstrates the importance of OSS analytics in understanding the rapidly evolving landscape of open-source development. The outcomes aid in identifying key trends, contributors' behavior, and global participation, ultimately leading to strategic insights and data-driven decisions within the OSS ecosystem.