To visually represent the popularity and distribution of the content, we can create a heatmap based on the total number of times each CID is requested and the number of unique users that have made a request for it. This visualization can help us quickly identify which pieces of content are most in demand and how widely they are being accessed by different users.

In [1]:
```
import pandas as pd
import numpy as np
from scipy import stats
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from tqdm.notebook import tqdm
import re
pd.set_option('max_columns', None)
pd.options.display.max_colwidth = 100
from pandas.api.types import CategoricalDtype
```

In [2]:
```
df = pd.read_csv('data.csv', index_col=0)
df.shape
```

Out[2]:
```
(6643221, 4)
```

In [3]:
```
df7 = df[['cid','agent']].groupby('cid').agg(['count',pd.Series.nunique])
df7.columns = df7.columns.get_level_values(1)
df7 = df7.reset_index()
df7 = df7.rename(columns={"nunique": "unique"})
df7['count'] = df7['count'].astype(int)
df7['unique'] = df7['unique'].astype(int)
df7 = df7.sort_values(by=['count'], ascending=False)
df7.head()
```

Out[3]:

	cid	count	unique
245661	bafybeifbabckr4o6peetw7jkwmpxswibwrave7tz2ireq2g46drkhdfuk4	101717	696
238346	bafybeicktciicnjr7yyvkp4fa6dac5ohhlyxpgwzr3rd2jfgogtkowbjq4	91533	825
240635	bafybeidg3tjbadjq4ntbl4ayoyeendnxjk2zyrq63gfxecoorc7me2y5cq	68304	551
252238	bafybeihnsvb3hbcvbkxpmzrpbg6z5lhpbcswmlceapr6fqsssghpmbemt4	58057	641
191976	QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq	55810	154

In [4]:
```
df7['count_type'] = ''

def addCountType(l, r, name):
    df7.loc[(df7['count'] >= l) & (df7['count'] < r), 'count_type'] = name

addCountType(1, 2, '1')
addCountType(2, 10, '[2,10)')
addCountType(10, 100, '[10,100)')
addCountType(100, 1000, '[100,1000)')
addCountType(1000, 10000, '[1000,10000)')
addCountType(10000, 100000, '[10000,100000)')

df7[df7['count']>100000].shape[0]
```

Out[4]:
```
1
```

In [5]:
```
df7['unique_type'] = ''

def addUniqueType(l, r, name):
    df7.loc[(df7['unique'] >= l) & (df7['unique'] < r), 'unique_type'] = name

addUniqueType(1, 2, '1')
addUniqueType(2, 10, '[2,10)')
addUniqueType(10, 100, '[10,100)')
addUniqueType(100, 1000, '[100,1000)')

df7[df7['unique'] >= 1000].shape[0]
```

Out[5]:
```
3
```

In [6]:
```
df8 = df7[['count_type','unique_type','cid']].groupby(['count_type','unique_type']).agg('count')
df8 = df8.rename(columns={"cid": "count"})
df8 = df8.reset_index()
total = df8['count'].sum()
df8['percentage'] = df8['count']/total*100
df8['percentage'] = df8['percentage'].apply(lambda x:round(x,3))
```

In [7]:
```
df8.head()
```
Out[7]:

	count_type	unique_type	count	percentage
0		[100,1000)	1	0.000
1	1	1	163887	64.377
2	[10,100)	1	3358	1.319
3	[10,100)	[10,100)	2133	0.838
4	[10,100)	[2,10)	7268	2.855

In [8]:
```
df8.groupby('unique_type').agg('sum')
```
Out[8]:

	count	percentage
unique_type		
	3	0.001
1	206748	81.213
[10,100)	3303	1.297
[100,1000)	442	0.173
[2,10)	44077	17.314

In [9]:
```
cat_unique_order = CategoricalDtype(
    ['1', '[2,10)', '[10,100)', '[100,1000)'],
    ordered=True
)
df8['unique_type'] = df8['unique_type'].astype(cat_unique_order)
df8 = df8.sort_values('unique_type')

cat_count_order = CategoricalDtype(
    ['1', '[2,10)', '[10,100)', '[100,1000)','[1000,10000)','[10000,100000)'],
    ordered=True
)
df8['count_type'] = df8['count_type'].astype(cat_count_order)
df8 = df8.sort_values('count_type')
df8 = df8.dropna()
df8.head()
```
Out[9]:

	count_type	unique_type	count	percentage
1	1	1	163887	64.377
19	[2,10)	[2,10)	36159	14.204
18	[2,10)	1	39304	15.439
3	[10,100)	[10,100)	2133	0.838
4	[10,100)	[2,10)	7268	2.855

In [10]:
```
data = df8.pivot(index='unique_type', columns='count_type', values='count')
data = np.log(data)
data = data.fillna(0)
data.head()
```
Out[10]:

count_type	**1**	**[2,10)**	**[10,100)**	**[100,1000)**	**[1000,10000)**	**[10000,100000)**
unique_type						
1	12.006932	10.579082	8.119101	5.164786	3.178054	0.000000
[2,10)	0.000000	10.495681	8.891236	6.327937	4.430817	1.791759
[10,100)	0.000000	0.000000	7.665285	6.659294	5.953243	1.609438
[100,1000)	0.000000	0.000000	0.000000	4.110874	5.533389	4.844187

In [11]:
```
text = df8.pivot(index='unique_type', columns='count_type', values='percentage')

text = text.T
text['1'] = text['1'] .astype(str) + "%"
text['[2,10)'] = text['[2,10)'] .astype(str) + "%"
text['[10,100)'] = text['[10,100)'] .astype(str) + "%"
text['[100,1000)'] = text['[100,1000)'] .astype(str) + "%"

text = text.T
text = text.replace("nan%","0%")
text.head()
```

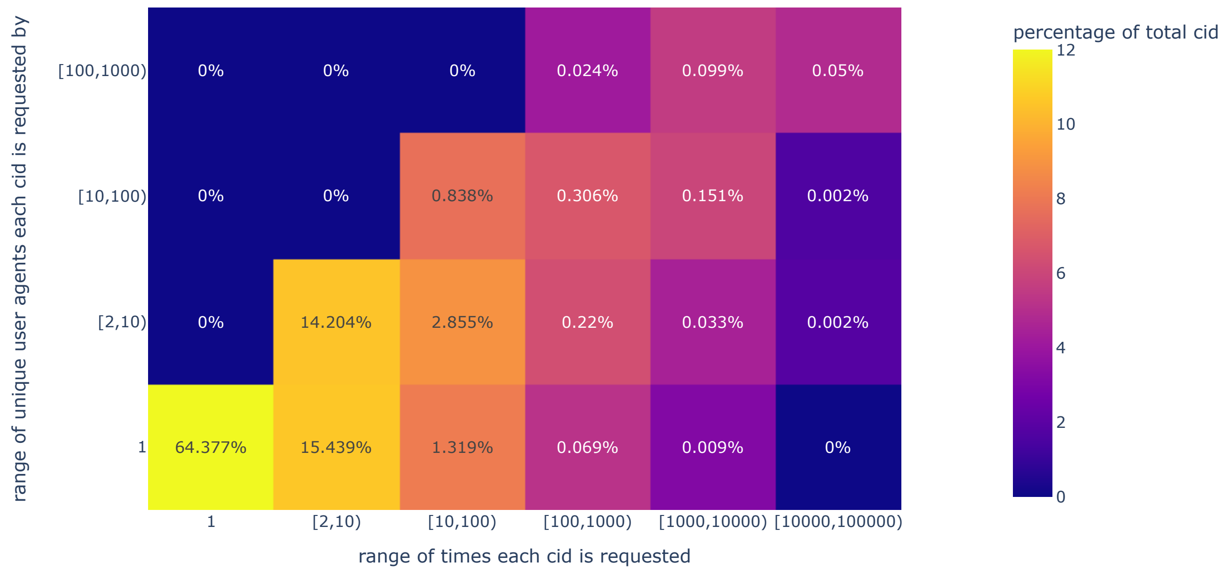
Out[11]:

count_type	1	[2,10)	[10,100)	[100,1000)	[1000,10000)	[10000,100000)
unique_type						
1	64.377%	15.439%	1.319%	0.069%	0.009%	0%
[2,10)	0%	14.204%	2.855%	0.22%	0.033%	0.002%
[10,100)	0%	0%	0.838%	0.306%	0.151%	0.002%
[100,1000)	0%	0%	0%	0.024%	0.099%	0.05%

In [12]:
```
fig = px.imshow(data,
                origin='lower',
               labels=dict(color="percentage of total cid"), text_auto=True)
fig = fig.update_traces(text=text, texttemplate="%{text}", hovertemplate=None)

fig.update_xaxes(side='bottom')
fig.update_xaxes(title="range of times each cid is requested")
fig.update_yaxes(title="range of unique user agents each cid is requested by")

fig.show()
```



In [13]:
```
data = df7[df7['count']>100]
data = data.reset_index().drop(['index'],axis=1).reset_index()
data['idx_percentage'] = data['index']/data.shape[0]
data.head()
```

Out[13]:

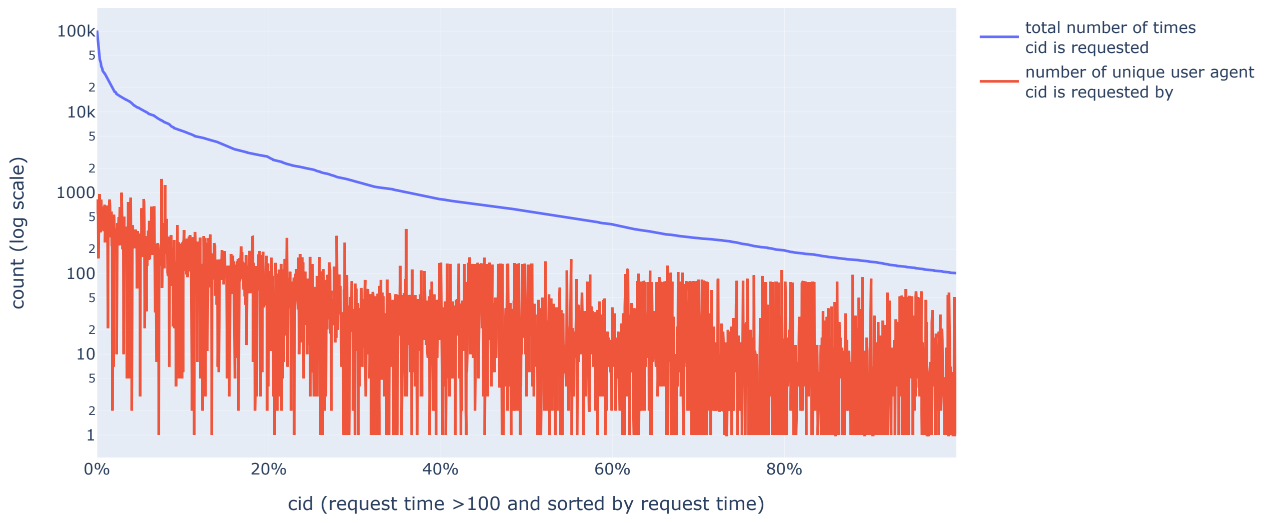
	index	cid	count	unique	count_type	unique_type	idx_percentage
0	0	bafybeifbabckr4o6peetw7jkwmpxswibwrave7tz2ireq2g46drkhdfuk4	101717	696		[100,1000)	0.000000
1	1	bafybeicktciicnjr7yyvkp4fa6dac5ohhlyxpgwzr3rd2jfgogtkowbjq4	91533	825	[10000,100000)	[100,1000)	0.000408
2	2	bafybeidg3tjbadjq4ntbl4ayoyeendnxjk2zyrq63gfxecoorc7me2y5cq	68304	551	[10000,100000)	[100,1000)	0.000817
3	3	bafybeihnsvb3hbcvbkxpmzrpbg6z5lhpbcswmlceapr6fqsssghpmbemt4	58057	641	[10000,100000)	[100,1000)	0.001225
4	4	QmeSjSinHpPnmXmspMjwiXyN6zS4E9zccariGR3jxcaWtq	55810	154	[10000,100000)	[100,1000)	0.001633

```
In [14]: fig = go.Figure([
             go.Scatter(
                 name='total number of times<br>cid is requested',
                 mode='lines',
                 x=data['idx_percentage'],
                 y=data['count']
             ),
             go.Scatter(
                 name='number of unique user agent<br>cid is requested by',
                 mode='lines',
                 x=data['idx_percentage'],
                 y=data['unique']
             )
         ])

         # fig.update_xaxes(visible=False, showticklabels=False)
         fig.update_xaxes(tickformat = ',.0%')
         fig.update_yaxes(type="log")

         fig.update_layout(xaxis_title='cid (request time >100 and sorted by request time)', yaxis_title='count (log scale)')

         fig.show()
```


100k
10k
count (log scale)
cid (request time >100 and sorted by request time)
total number of times cid is requested
number of unique user agent cid is requested by