In [1]:
```
import os
import sys

import altair as alt
import pandas as pd
import vega
alt.renderers.enable('mimetype')
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

sys.path.append(os.path.join(os.getcwd(),os.pardir))
from src.data.load_preprocess_data import load_and_preprocess_raw_complaints

data_path = os.path.join(os.pardir, "data", "raw", "complaints.csv")
if os.path.exists(data_path):
    complaints_df = load_and_preprocess_raw_complaints_data(data_path)
else:
    raise FileNotFoundError('Data is not downloaded or not in the correct fo
```

Inspection

- We can see that the interested target only has 768443 valid values, under which we want to trim the data frame to have null dispute responses removed.
- We can drop non-useful features like `zip_code` and `complaint_id`.
- It seems that we can process the `consumer_complaint_narrative` using NLP and other useful features using `OneHotEncoder` (apply binary encoding if necessary) since the unique values of most of the features are not too many.

In [2]:
```
unique_df = pd.DataFrame()
unique_df['columns'] = complaints_df.columns
unique_df['valid_count'] = complaints_df.count(axis=0).reset_index()[0]
unique_df['unique_count'] = complaints_df.nunique().reset_index()[0]
unique_df
```

Out[2]:

	columns	valid_count	unique_count
0	date_received	3122836	4018
1	product	3122836	18
2	sub_product	2887543	76
3	issue	3122836	165
4	sub_issue	2438461	221
5	consumer_complaint_narrative	1121913	979279
6	company_public_response	1359325	11
7	company	3122836	6579
8	state	3082743	63
9	zip_code	3082222	34463
10	tags	353109	3
11	consumer_consent_provided	2297533	4
12	submitted_via	3122836	7
13	date_sent_to_company	3122836	3967
14	company_response_to_consumer	3122832	8
15	timely_response	3122836	2
16	consumer_disputed	768440	2
17	complaint_id	3122836	3122836

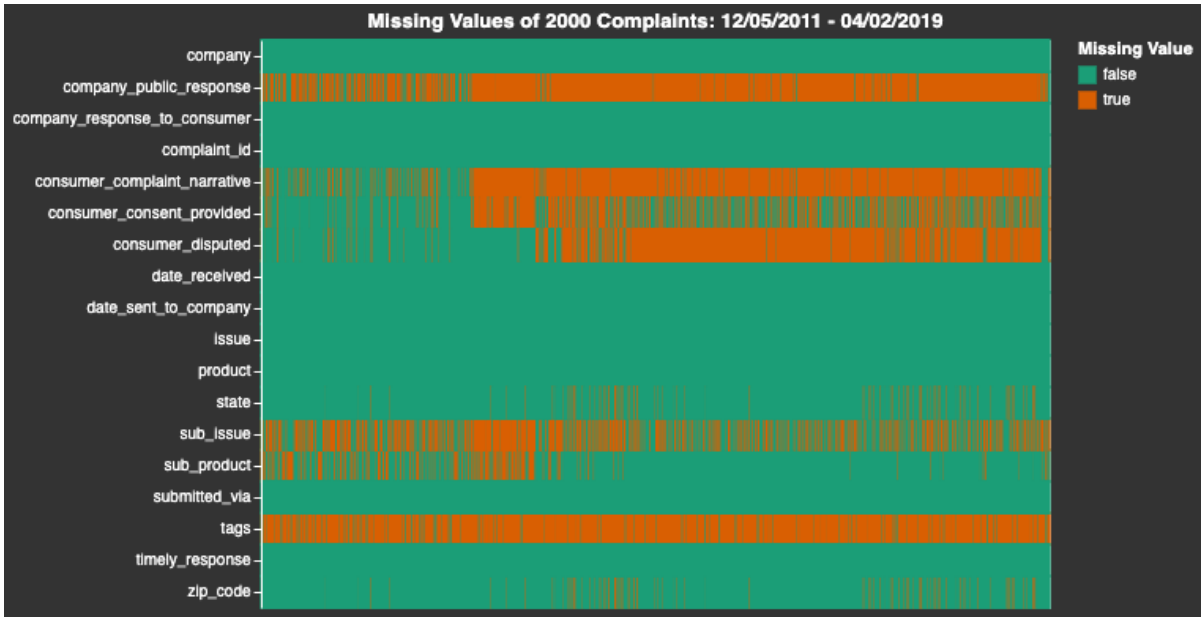
Missing Values

In [3]:

```
num_complaints = 2000
alt.data_transformers.enable('data_server')
na_val_df = complaints_df.tail(num_complaints).isna().reset_index().melt(
        id_vars='index'
    )
last_date = complaints_df.date_received.tail(num_complaints).max().strftime(
first_date = complaints_df.date_received.tail(num_complaints).min().strftime

missing_vals = alt.Chart(
    complaints_df.tail(num_complaints).isna().reset_index().melt(
        id_vars='index'
    ),
    title = f"Missing Values of {num_complaints} Complaints: {first_date} -
).mark_rect().encode(
    alt.X('index:O', axis=None),
    alt.Y('variable', title=None),
    alt.Color('value', title='Missing Value',scale=alt.Scale(scheme='dark2')
    alt.Stroke('value', scale=alt.Scale(scheme='dark2'))  # We set the strok
).properties(
    width=min(500, complaints_df.tail(num_complaints).shape[0])
```

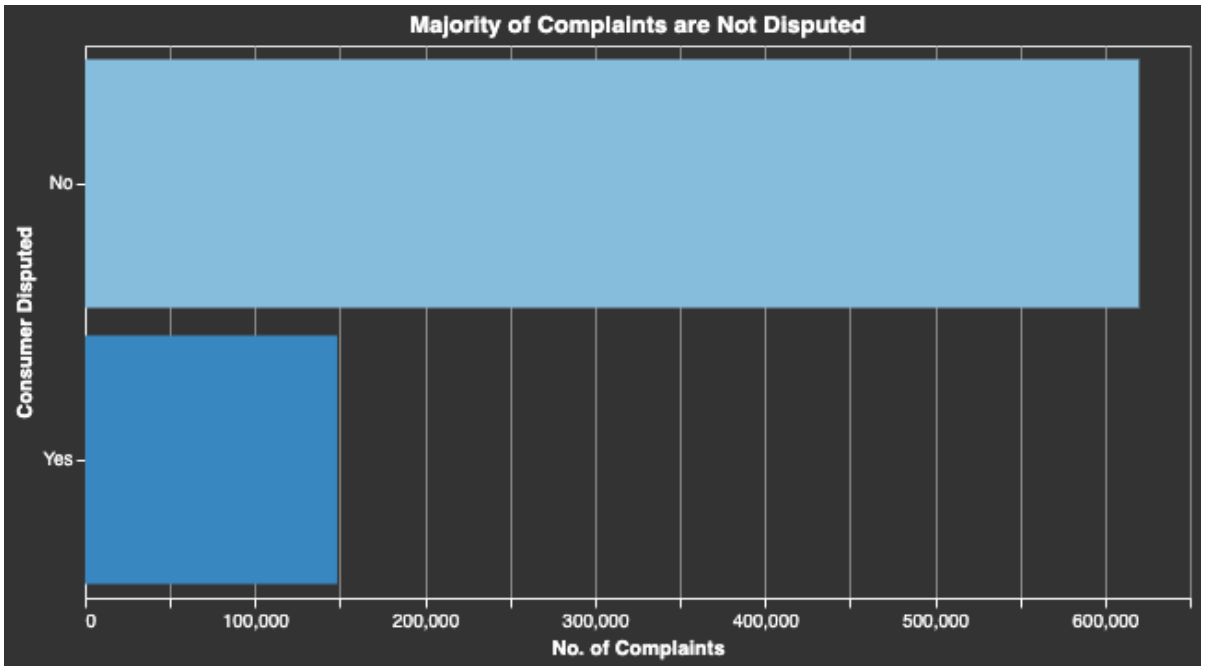
```
);
display(missing_vals)
```



- From the graph below, we see an imbalanced class, which we should take into account during later training of the model.

In [4]:
```
# complaints_df = complaints_df.query('not consumer_disputed.isnull()')
complaints_df.head()
target = pd.DataFrame(complaints_df.value_counts('consumer_disputed')).reset
target.columns = ['consumer_disputed','count']
alt.Chart(
    target,
    title = "Majority of Complaints are Not Disputed"
).mark_bar().encode(
    y=alt.Y('consumer_disputed:O',title = 'Consumer Disputed'),
    x=alt.X('count:Q',title = 'No. of Complaints'),
    color=alt.Color('consumer_disputed:O', legend=None),
).properties(
    width = 600,
    height = 300
)
```

Out[4]:



Wordcloud of Customer Review

Here we show two visualizations of what customers mentioned for their complaints in disputed/non-disputed classes.

In [5]:

```
stopwords = set(STOPWORDS)
disputed_words = str(complaints_df['issue'].loc[complaints_df['consumer_disp
undisputed_words = str(complaints_df['issue'].loc[complaints_df['consumer_di
print("\033[1mComparing the Most common words in issues between disputed and

plt.title("Most common Issue words for Disputed Consumers:")
wordcloud = WordCloud(background_color="white", collocations=False, colormap
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()

plt.title("Most common Issue words for undisputed Consumers:")
wordcloud = WordCloud(background_color="white", collocations=False, colormap
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

**Comparing the Most common words in issues between disputed and undisputed c
onsumers:**

Insights

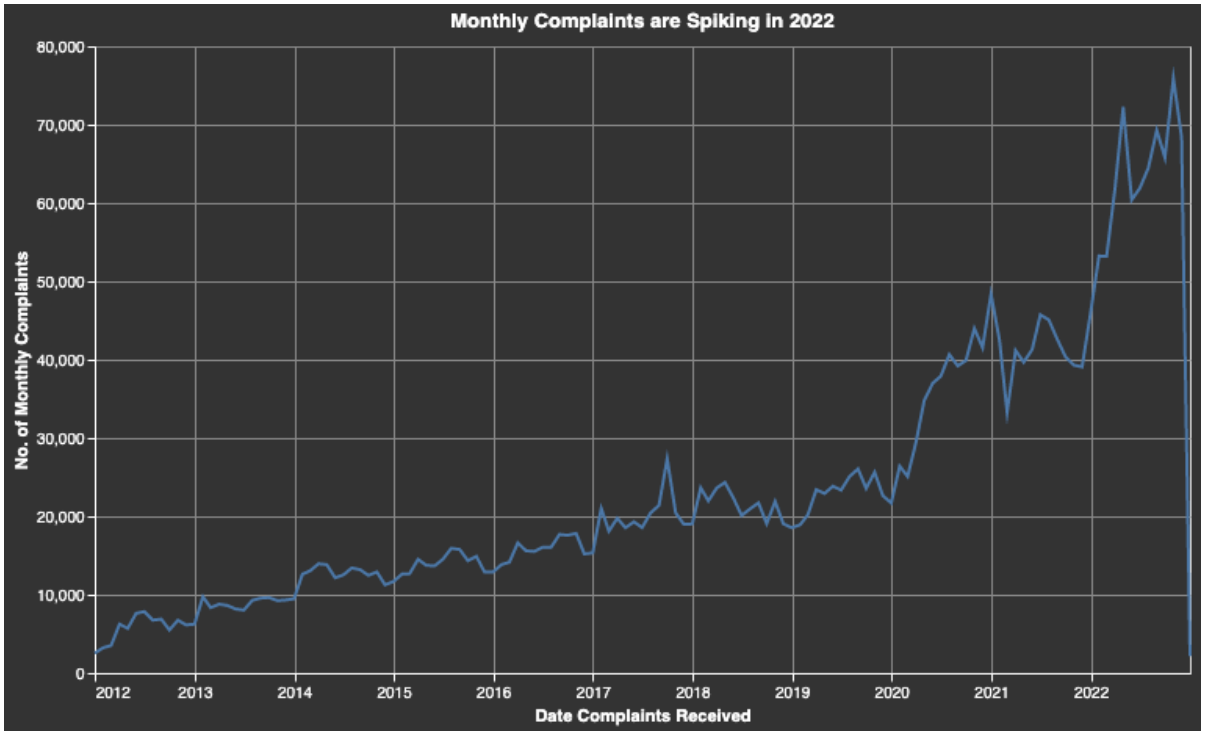
- from July 2022 to November no consumers were recorded as disputing a claim, potentially because they haven't been processed yet?
- what about older claims?

In [6]:
```
complaints_df = load_and_preprocess_raw_complaints_data(data_path)
num_complaints = complaints_df.resample("M", on="date_received").agg({'date_

alt.Chart(
    num_complaints,
    title = "Monthly Complaints are Spiking in 2022"
).mark_line().encode(
        x = alt.X('date_received:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints:Q", title = "No. of Monthly Complaints")
).properties(
    width = 700,
```

```
    height = 400
)
```

Out[6]:

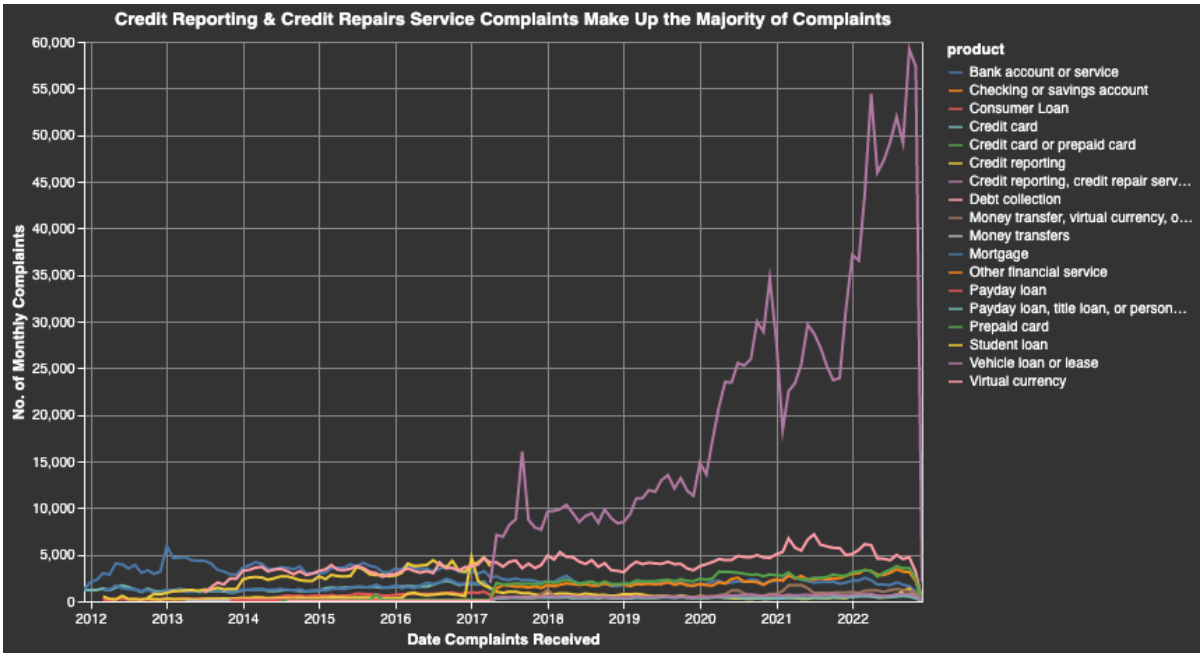


Complaints by Product

In [7]:

```
proc_complaints_df = complaints_df.copy(deep=True)
proc_complaints_df["year_month"] = proc_complaints_df.date_received.apply(la
alt.Chart(
    proc_complaints_df.groupby(['year_month', "product"], as_index=False).ag
        num_complaints = ("product", "size")
    ),
    title = "Credit Reporting & Credit Repairs Service Complaints Make Up th
).mark_line().encode(
        x = alt.X('year_month:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints", title = "No. of Monthly Complaints"),
    color = "product",
    tooltip="product"
).properties(
    width = 600,
    height = 400
).interactive()
```

Out[7]:



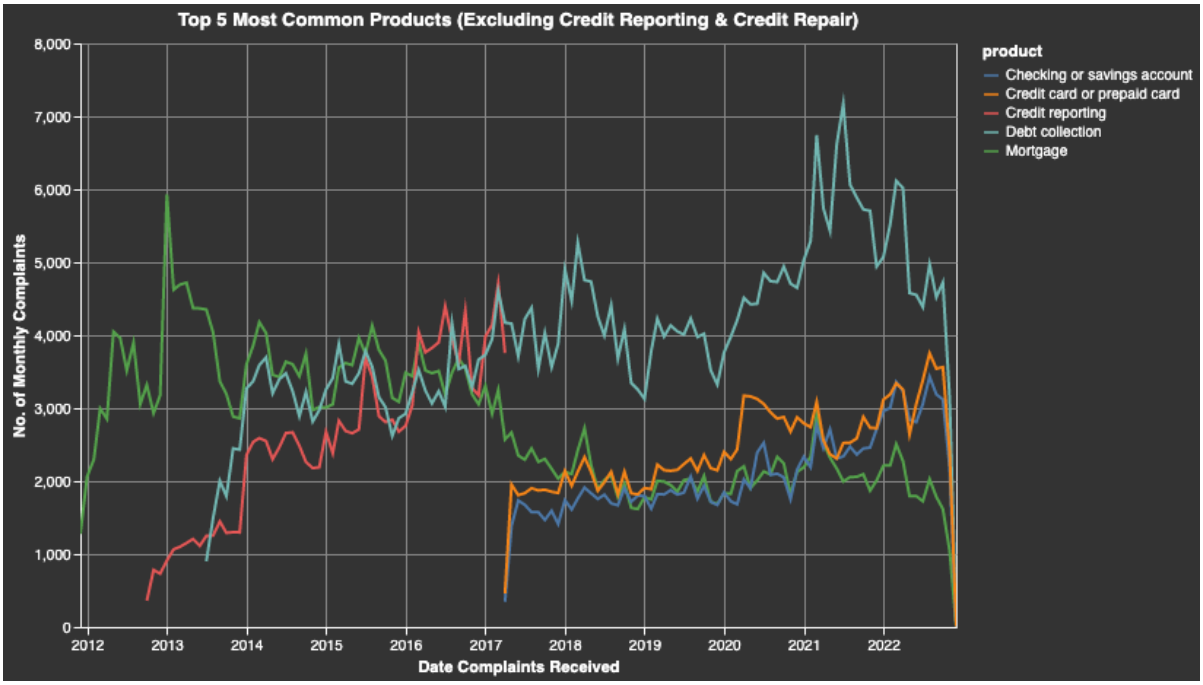
In [8]:

```
most_complaint_types = (
    proc_complaints_df.groupby(["product"], as_index=False)
    .agg(num_complaints = ("product", "size"))
    .sort_values(by = "num_complaints", ascending=False)
    .head(6)["product"]
    .to_list()
)

most_complaint_types = list(filter(lambda x: not x.startswith("Credit report

alt.Chart(
    proc_complaints_df.loc[proc_complaints_df["product"].isin(most_complaint
    num_complaints = ("product", "size")
    ),
    title = "Top 5 Most Common Products (Excluding Credit Reporting & Credit
).mark_line().encode(
        x = alt.X('year_month:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints", title = "No. of Monthly Complaints"),
    color = "product",
    tooltip="product"
).properties(
    width = 600,
    height = 400
).interactive()
```

Out[8]:

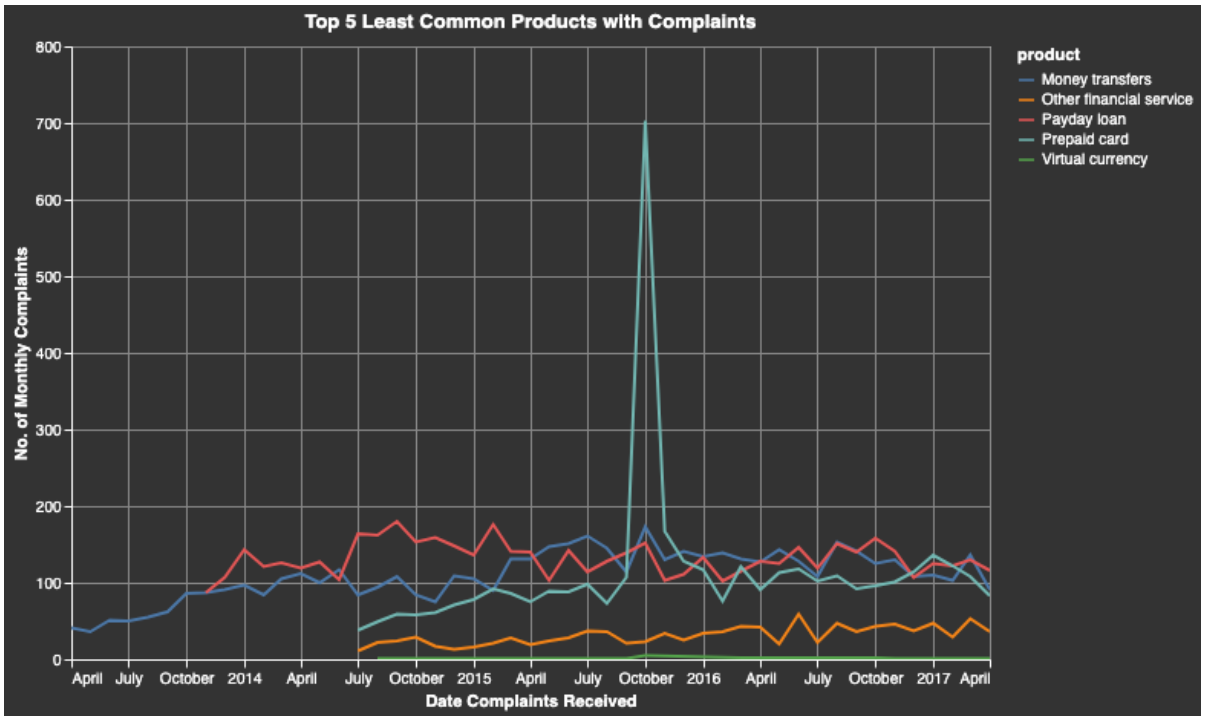


In [9]:

```
least_complaint_types = (
    proc_complaints_df.groupby(["product"], as_index=False)
    .agg(num_complaints = ("product", "size"))
    .sort_values(by = "num_complaints", ascending=True)
    .head(5)["product"]
    .to_list()
)

alt.Chart(
    proc_complaints_df.loc[proc_complaints_df["product"].isin(least_complain
    num_complaints = ("product", "size")
    ),
    title = "Top 5 Least Common Products with Complaints"
).mark_line().encode(
        x = alt.X('year_month:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints", title = "No. of Monthly Complaints"),
    color = "product",
    tooltip="product"
).properties(
    width = 600,
    height = 400
).interactive()
```

Out[9]:

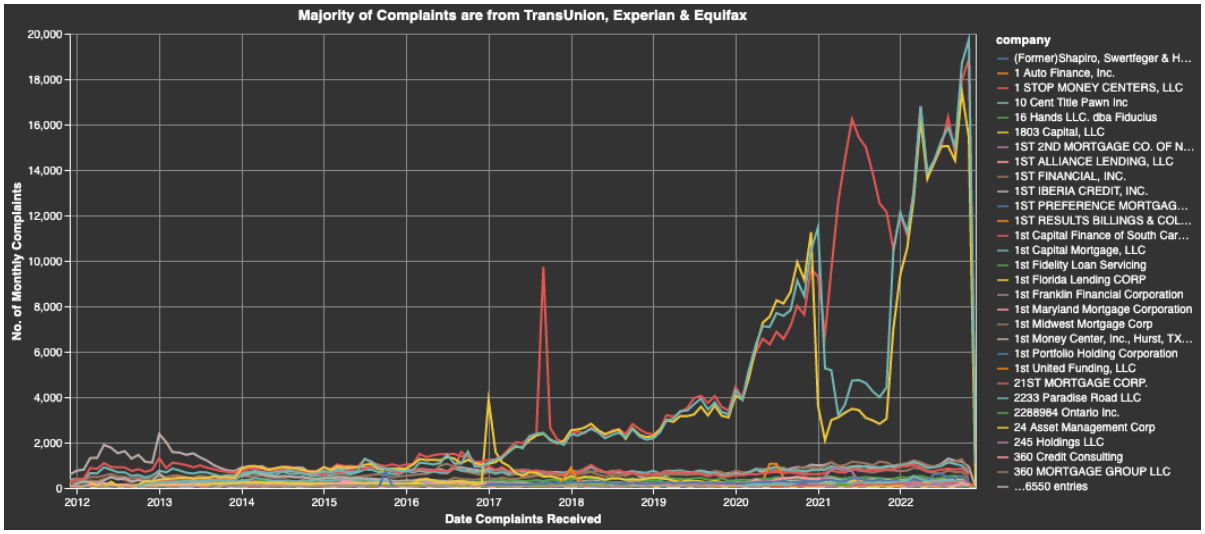


Complaints by Company

In [10]:

```
alt.Chart(
    proc_complaints_df.groupby(['year_month', "company"], as_index=False).ag
        num_complaints = ("company", "size")
    ),
    title = "Majority of Complaints are from TransUnion, Experian & Equifax"
).mark_line().encode(
        x = alt.X('year_month:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints", title = "No. of Monthly Complaints"),
    color = "company",
    tooltip="company"
).properties(
    width = 800,
    height = 400
).interactive()
```

Out[10]:



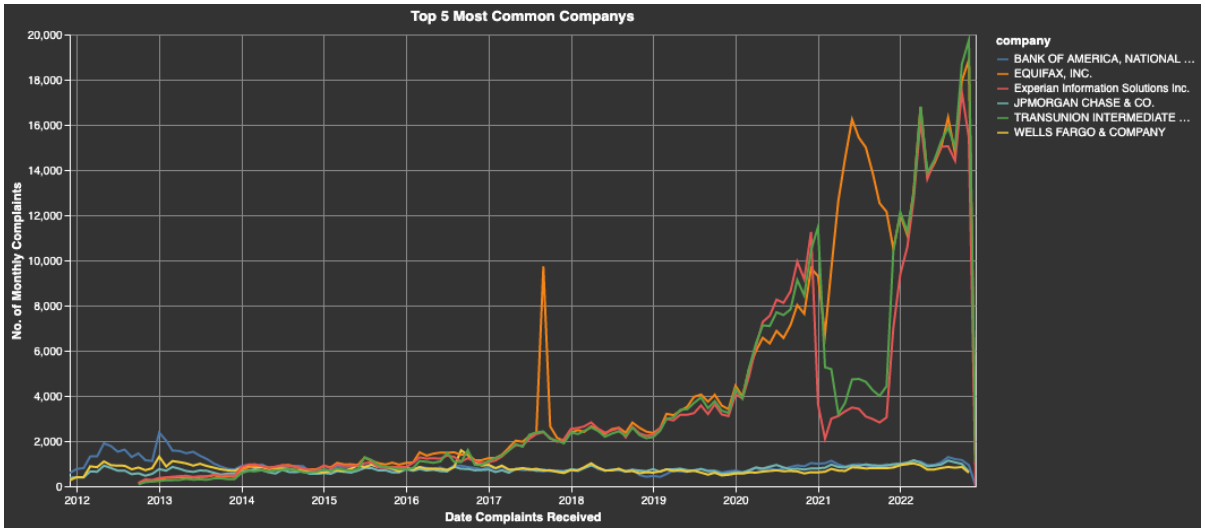
Insight: looks like majority of compalints come from 3 companies

Top 6 Companies by Number of Complaints

In [11]:
```
most_complaint_types = (
    proc_complaints_df.groupby(["company"], as_index=False)
    .agg(num_complaints = ("company", "size"))
    .sort_values(by = "num_complaints", ascending=False)
    .head(6)["company"]
    .to_list()
)

alt.Chart(
    proc_complaints_df.loc[proc_complaints_df["company"].isin(most_complaint
    num_complaints = ("company", "size")
    ),
    title = "Top 5 Most Common Companys"
).mark_line().encode(
        x = alt.X('year_month:T', title="Date Complaints Received"),
        y = alt.Y("num_complaints", title = "No. of Monthly Complaints"),
    color = "company",
    tooltip="company"
).properties(
    width = 800,
    height = 400
).interactive()
```

Out[11]:



Insight: Top 3 Companies with Most Complaints: Equifax, TransUnion, Experian