

```
In [1]: import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
from sklearn.stats import boxcox
from sklearn.preprocessing import RobustScaler
import missingno as mso


In [2]: pipeline_df = pd.read_csv(r'C:\Users\vmotd\OneDrive\Documents\Boot_Camp_Files\Shiny_Project\Shiny_Project\Pipeline_I
ncidents.csv')

Assessing missingness patterns in pipeline dataframe and handling them accordingly

In [3]: missing_data = pipeline_df.columns[pipeline_df.isnull().any()].tolist()

mso.matrix(pipeline_df[missing_data], labels = True)

Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x2073996c889>
```



Imputing 0 for incidents where NaN represents that no injuries/fatalities occurred

```
In [4]: def imputation():
    pipeline_df.loc[:, ['Public Evacuations', 'Other Costs']] = pipeline_df.loc[:,
    pipeline_df.loc[:,
        'Pipeline/Facility Name': 'Accident State'] = pipeline_df.loc[:,
    'known')
    pipeline_df.loc[:, ['Intentional Release (Barrels)']] = pipeline_df.loc[:,
    pipeline_df['Pipeline Shutdown'] = pipeline_df['Pipeline Shutdown'].fillna(0)
    shutdown_null = pipeline_df.loc[:, ['Shutdown Date/Time']].fillna(0)
    pipeline_df.loc[shutdown_null, ['Restart Date/Time']] = pipeline_df['Restart Date/Time'].isna().index
    shutdown_null = pipeline_df.loc[:, ['Shutdown Date/Time']]
    pipeline_df.loc[shutdown_null, ['Restart Date/Time']] = pipeline_df['Restart Date/Time'].isna().index
    pipeline_df.loc[shutdown_null, ['Shutdown Date/Time', 'Restart Date/Time']] = pipeline_df.loc[shutdown_null,
    return pipeline_df

In [5]: pipeline_df = imputation()

In [6]: pipeline_df.loc[:, ['Accident Date/Time', 'Shutdown Date/Time',
    'Restart Date/Time']] = pipeline_df.loc[:, ['Accident Date/Time', 'Shutdown Date/Time',
    'Restart Date/Time']].apply(lambda x: pd.to_datetime(x), axis = 1)

In [7]: pipeline_df = pipeline_df.set_index('Accident Date/Time')

In [8]: monthly_incidents = pd.DataFrame(pipeline_df.groupby(pd.Grouper(freq = 'm'))['Report Number'].count())

In [9]: monthly_incidents['year'] = monthly_incidents.index.year

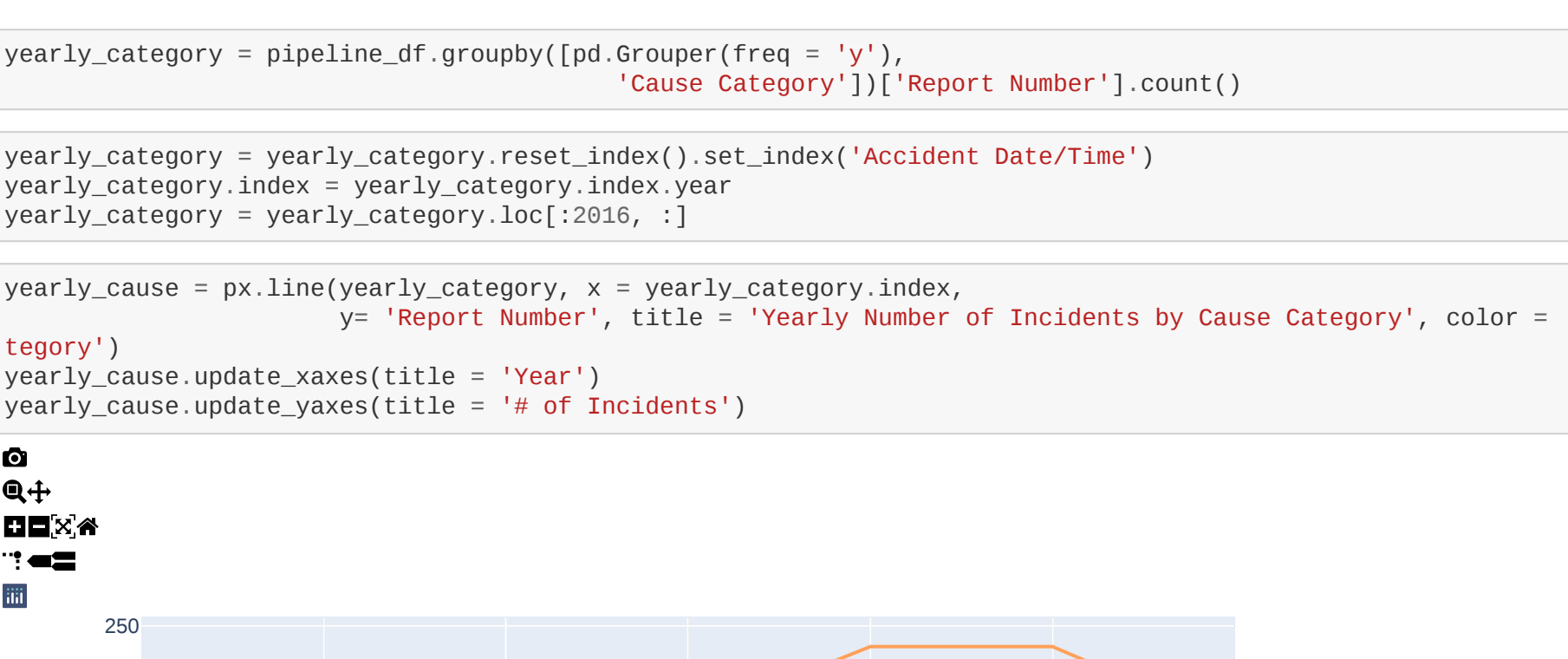
In [10]: monthly_incidents.columns = ['Incidents_num', 'year']

In [11]: yearly_incidents = pd.DataFrame(pipeline_df.groupby(pd.Grouper(freq = 'y'))['Report Number'].count())

In [12]: yearly_incidents.index = yearly_incidents.index.year
yearly_incidents = yearly_incidents.loc['2010': '2016',:]

In [13]: yearly_incidents.columns = ['num_incidents', 'year']

In [14]: yearly_graph = px.line(yearly_incidents, x = yearly_incidents.index,
    y = 'num_incidents', title = 'Yearly Number of Incidents')
yearly_graph.update_xaxes(title = 'Year')
yearly_graph.update_yaxes(title = 'n of Incidents')
```



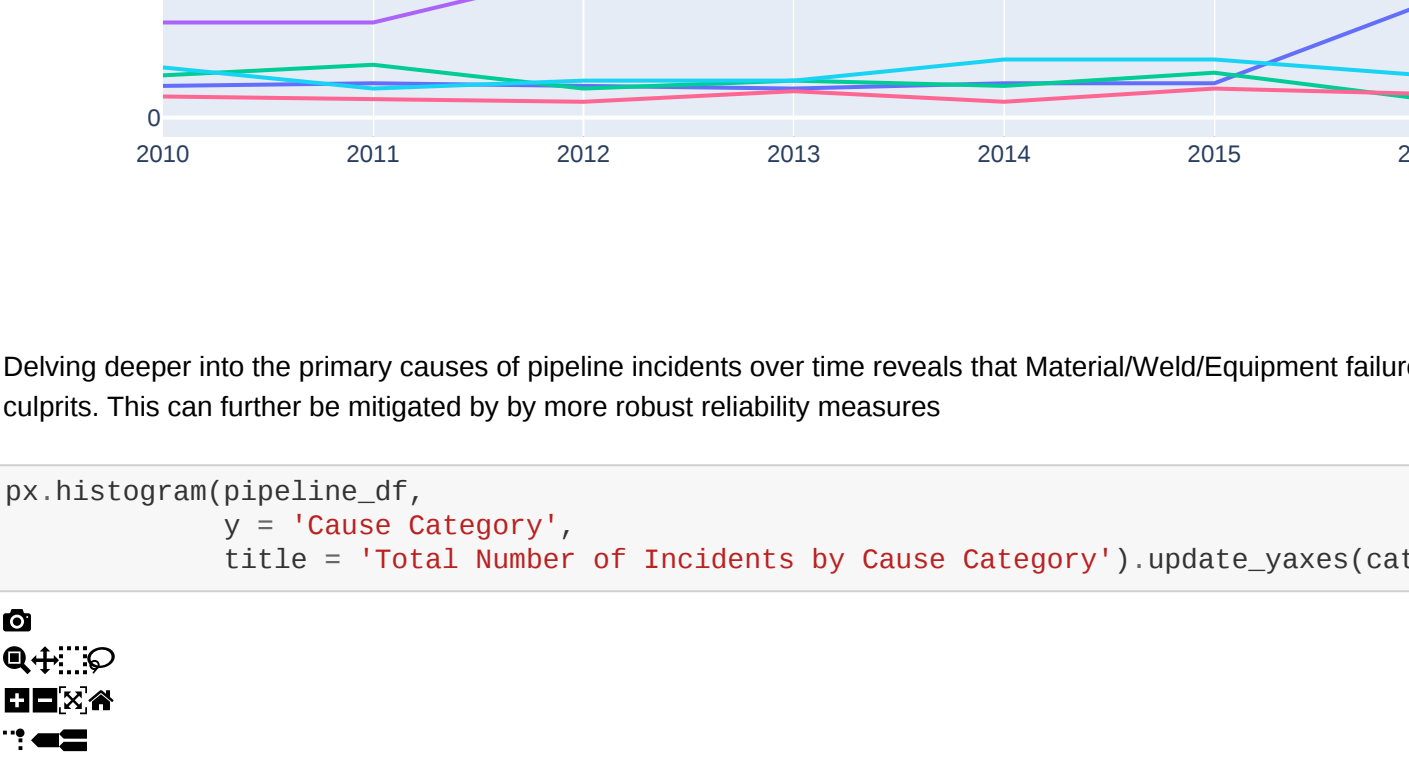
The trend of incidents shows an exponential upwards trend between 2011 and 2014. From 2014 to 2016, there is a slight upwards linear trend followed by a decreasing linear trend in 2016. This may suggest the implementation of more robust reliability systems such as predictive maintenance or leak detection systems.

What are the growth trends of accident categories

```
In [15]: yearly_category = pipeline_df.groupby([pd.Grouper(freq = 'y'),
    'Cause Category'])['Report Number'].count()

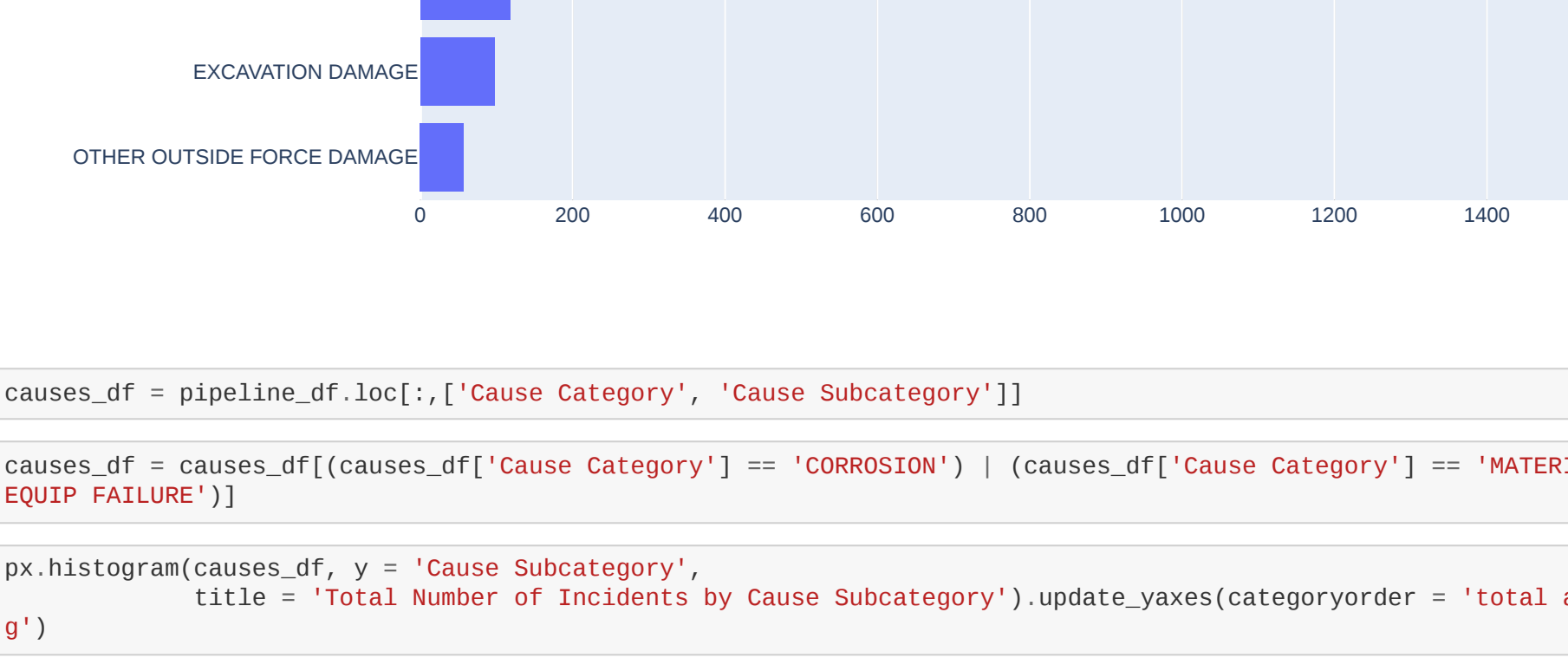
In [16]: yearly_category = yearly_category.reset_index().set_index('Accident Date/Time')
yearly_category.index = yearly_category.index.year
yearly_category = yearly_category.loc['2010',:]

In [17]: yearly_cause = px.line(yearly_category, x = yearly_category.index,
    y = 'Report Number', title = 'Yearly Number of Incidents by Cause Category', color = 'Cause Ca
teory')
yearly_cause.update_xaxes(title = 'Year')
yearly_cause.update_yaxes(title = 'n of Incidents')
```



Diving deeper into the primary causes of pipeline incidents over time reveals that Material/Weld/Equipment failures followed by corrosion are the primary culprits. This can further be mitigated by more robust reliability measures

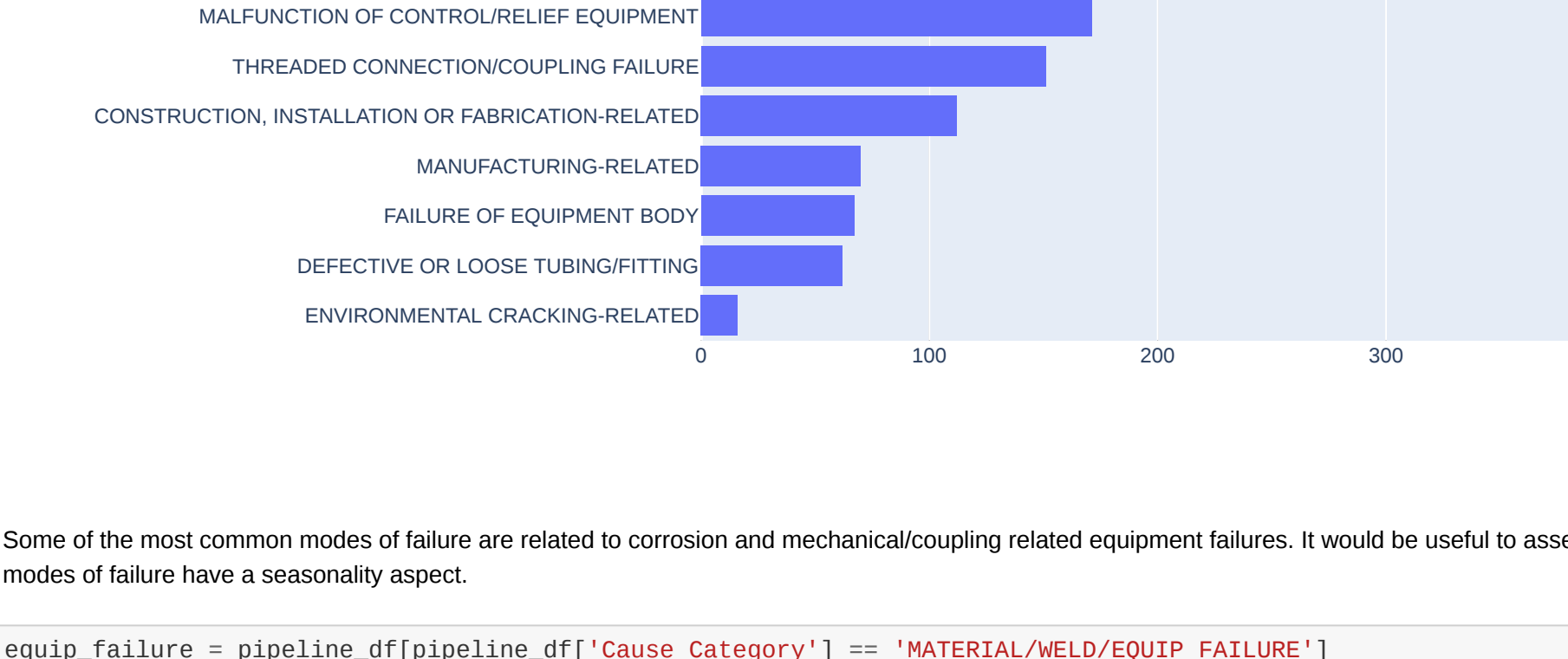
```
In [18]: px.histogram(pipeline_df,
    y = 'Cause Category',
    title = 'Total Number of Incidents by Cause Category').update_yaxes(categoryorder = 'total ascending')
```



```
In [19]: causes_df = pipeline_df.loc[:, ['Cause Category', 'Cause Subcategory']]

In [20]: causes_df = causes_df[(causes_df['Cause Category'] == 'CORROSION') | (causes_df['Cause Category'] == 'MATERIAL/WELD/
EQUIP FAILURE')]

In [21]: px.histogram(causes_df, y = 'Cause Subcategory',
    title = 'Total Number of Incidents by Cause Subcategory').update_yaxes(categoryorder = 'total ascendin
g')
```



Some of the most common modes of failure are related to corrosion and mechanical/coupling related equipment failures. It would be useful to assess if the modes of failure have a seasonality aspect.

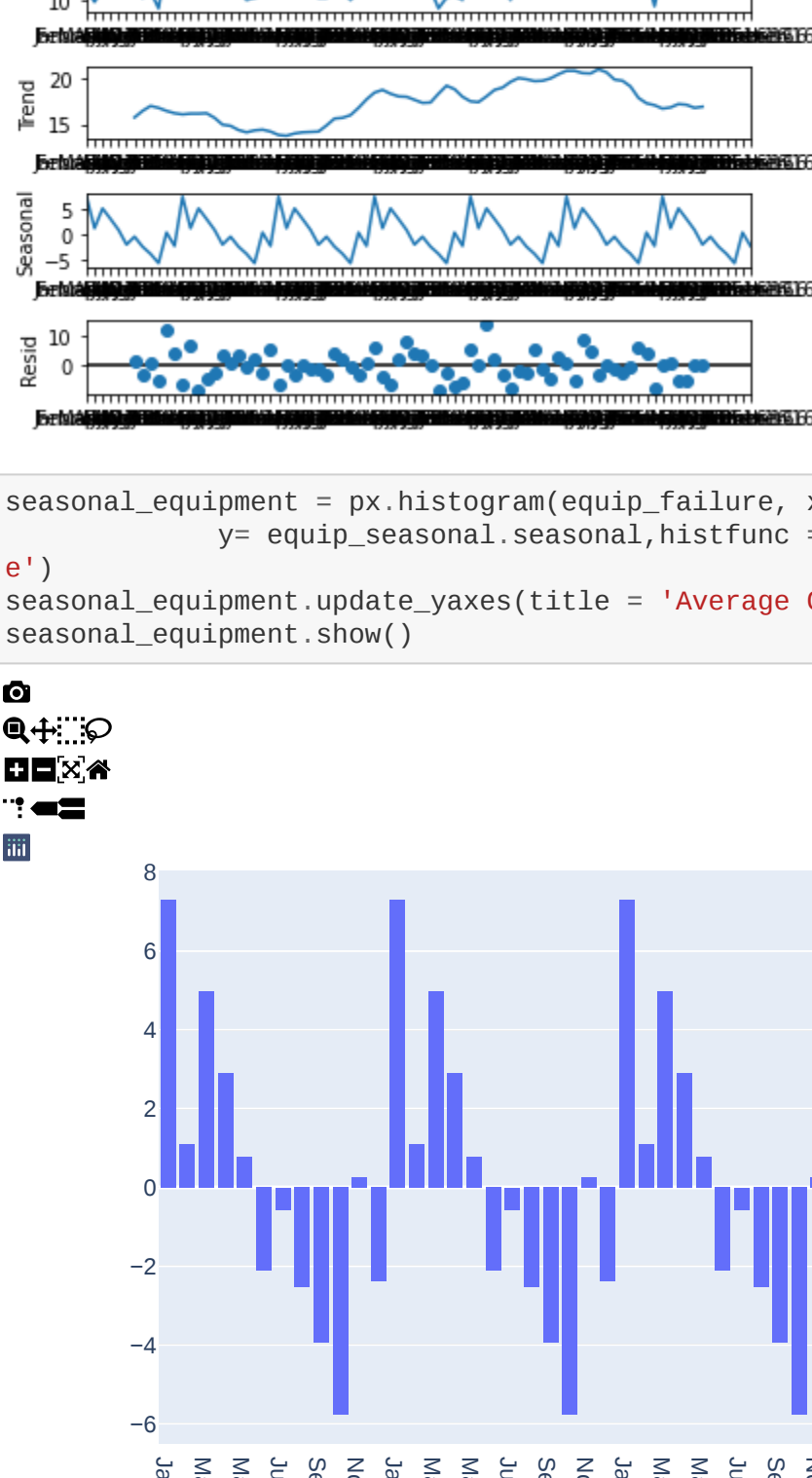
```
In [22]: equip_failure = pipeline_df[pipeline_df['Cause Category'] == 'MATERIAL/WELD/EQUIP FAILURE']

In [23]: equip_failure = equip_failure.groupby([pd.Grouper(freq = 'm')])['Report Number'].count()

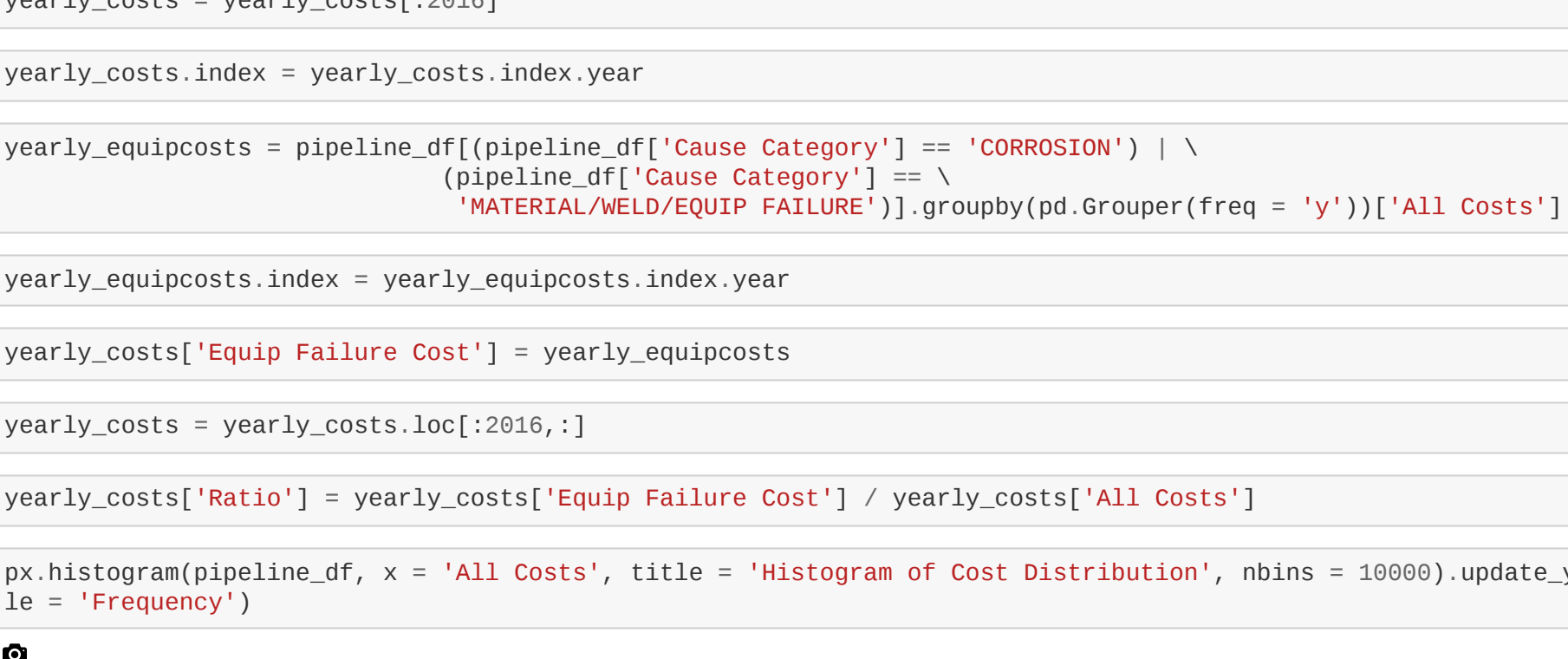
In [24]: equip_failure.index = equip_failure.index.strftime('%B-%y')

In [25]: equip_seasonal = seasonal_decompose(equip_failure, model = 'Multiplicative', period = 12)
equip_seasonal.plot()
```

```
Out[25]:
```



```
In [26]: seasonal_equipment = px.histogram(equip_failure,
    y = equip_seasonal.seasonal, histfunc = 'avg', nbins = 120, title = 'Seasonality of Equipment/Seal Failur
e')
seasonal_equipment.update_yaxes(title = 'Average Change in Incidents by Month')
seasonal_equipment.show()
```



It would appear that most equipment related failures occur during the transition from winter to spring. This could be caused by change in temperature between seasons. Mechanical equipments contracts in the winter and expands with heat. As the equipment stabilizes to temperature, the incidence rate decreases.

```
In [27]: yearly_costs = pd.DataFrame(pipeline_df.groupby([pd.Grouper(freq = 'y')])['All Costs'].sum())

In [28]: yearly_costs = yearly_costs['2010']

In [29]: yearly_costs.index = yearly_costs.index.year

In [30]: yearly_equipcosts = pipeline_df[(pipeline_df['Cause Category'] == 'CORROSION') | \
    (pipeline_df['Cause Category'] == \
    'MATERIAL/WELD/EQUIP FAILURE')].groupby(pd.Grouper(freq = 'y'))['All Costs'].sum()

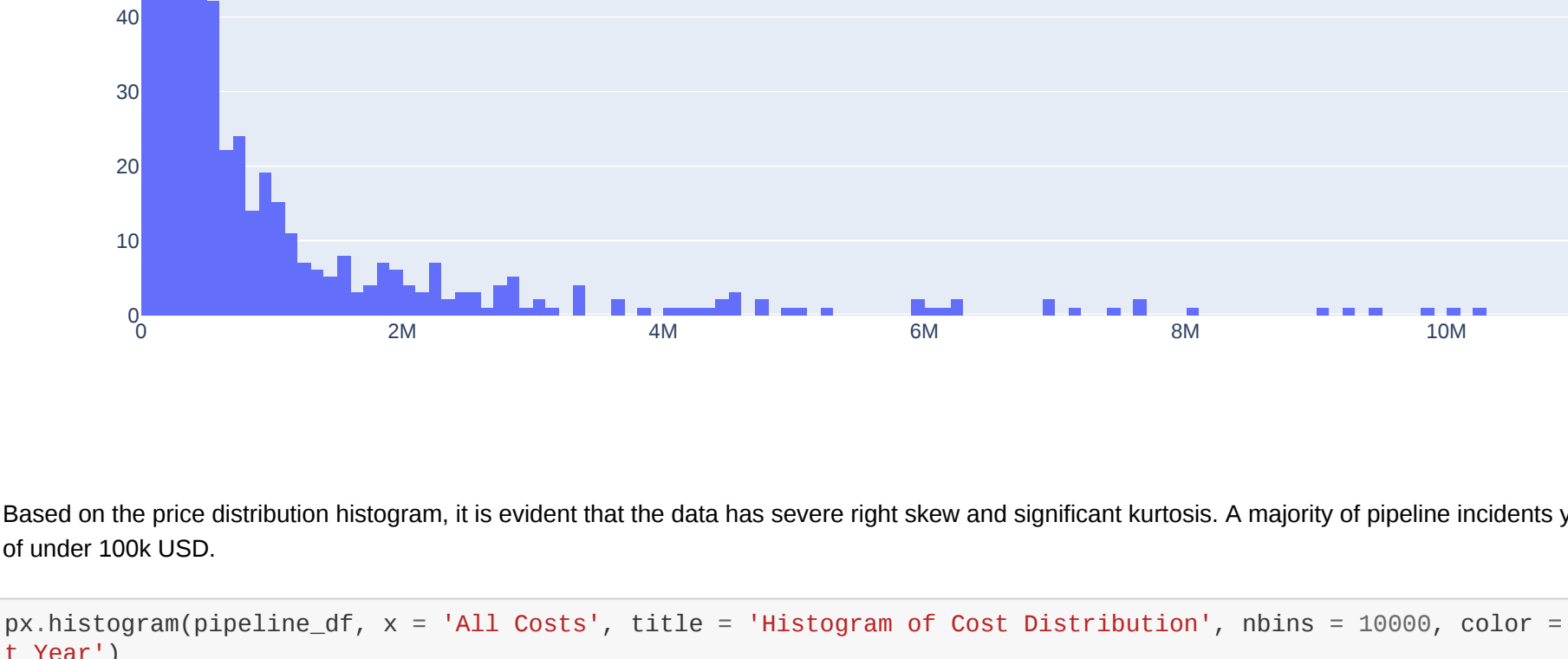
In [31]: yearly_equipcosts.index = yearly_equipcosts.index.year

In [32]: yearly_costs['Equip Failure Cost'] = yearly_equipcosts

In [33]: yearly_costs = yearly_costs.loc['2010',:]

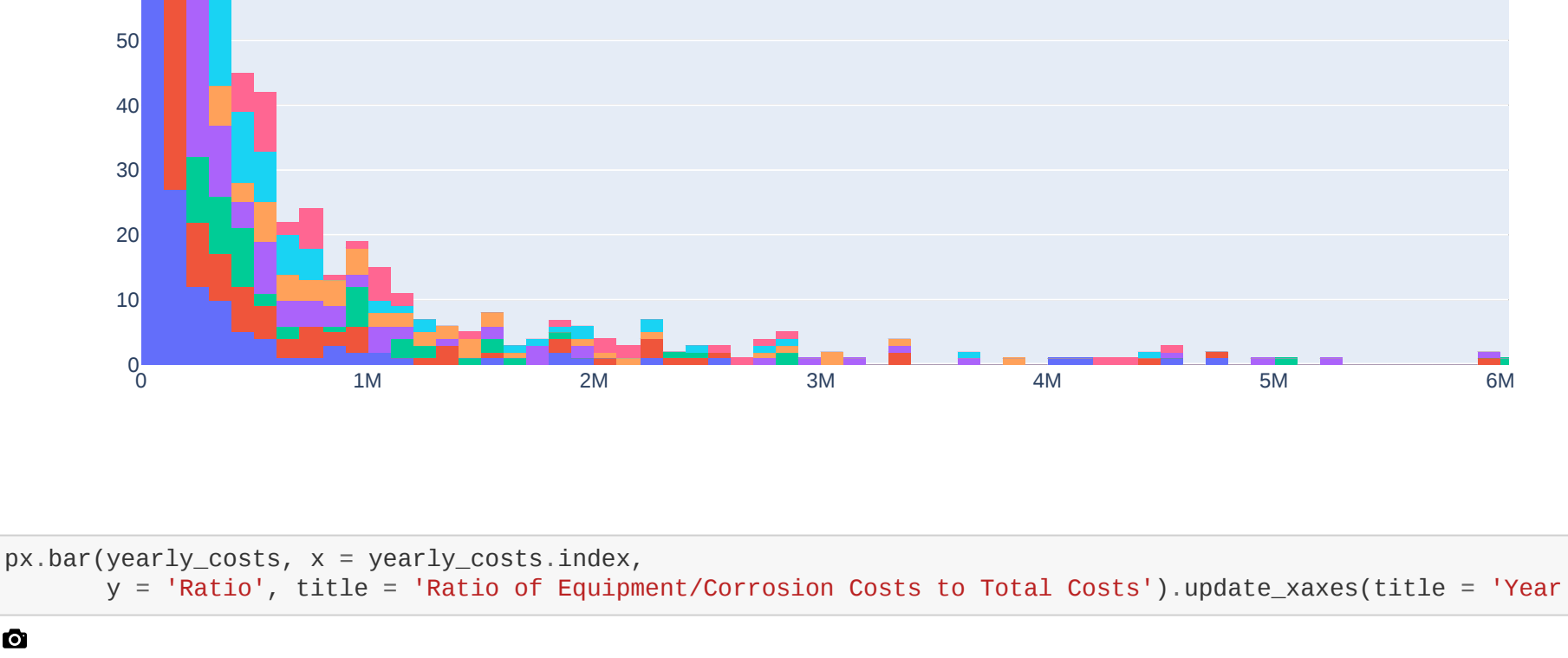
In [34]: yearly_costs['Ratio'] = yearly_costs['Equip Failure Cost'] / yearly_costs['All Costs']

In [35]: px.histogram(pipeline_df, x = 'All Costs', title = 'Histogram of Cost Distribution', nbins = 10000).update_yaxes(tit
le = 'Frequency')
```

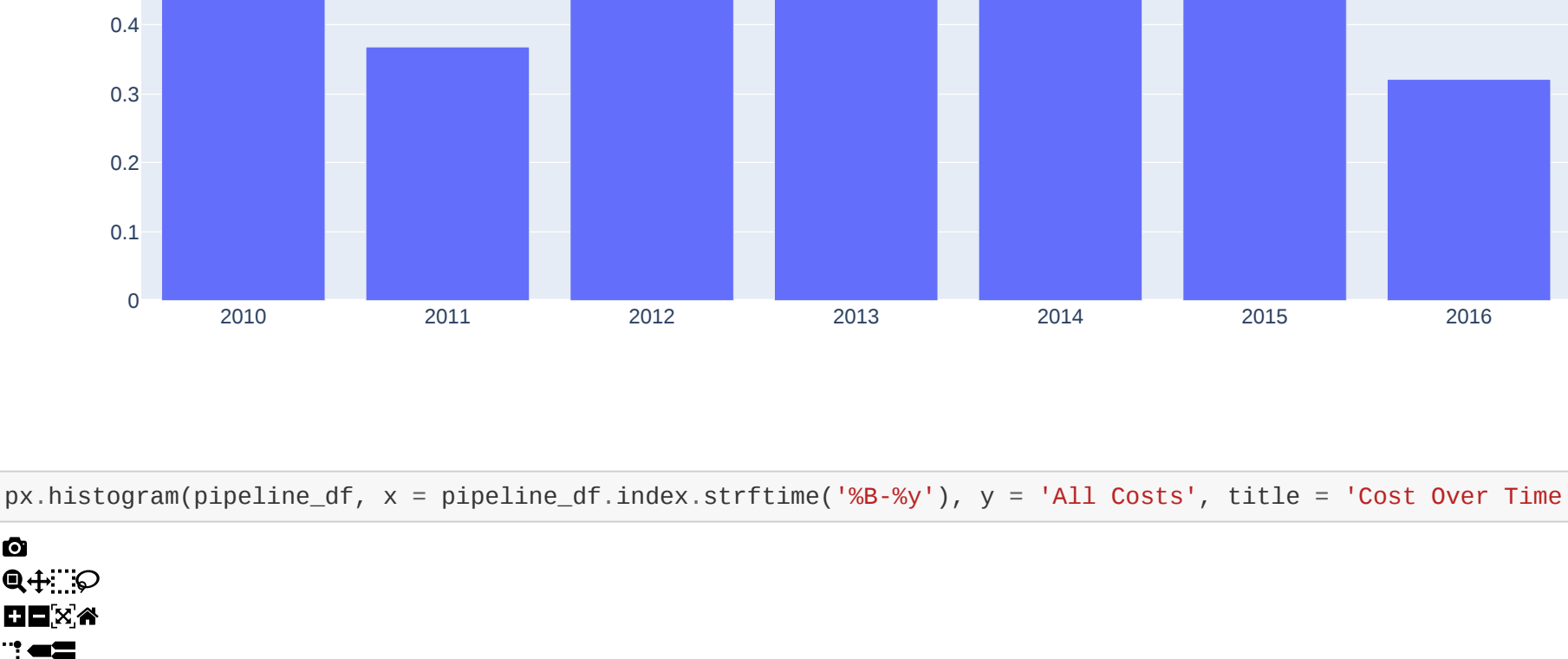


Based on the price distribution histogram, it is evident that the data has severe right skew and significant kurtosis. A majority of pipeline incidents yield losses of under 100k USD.

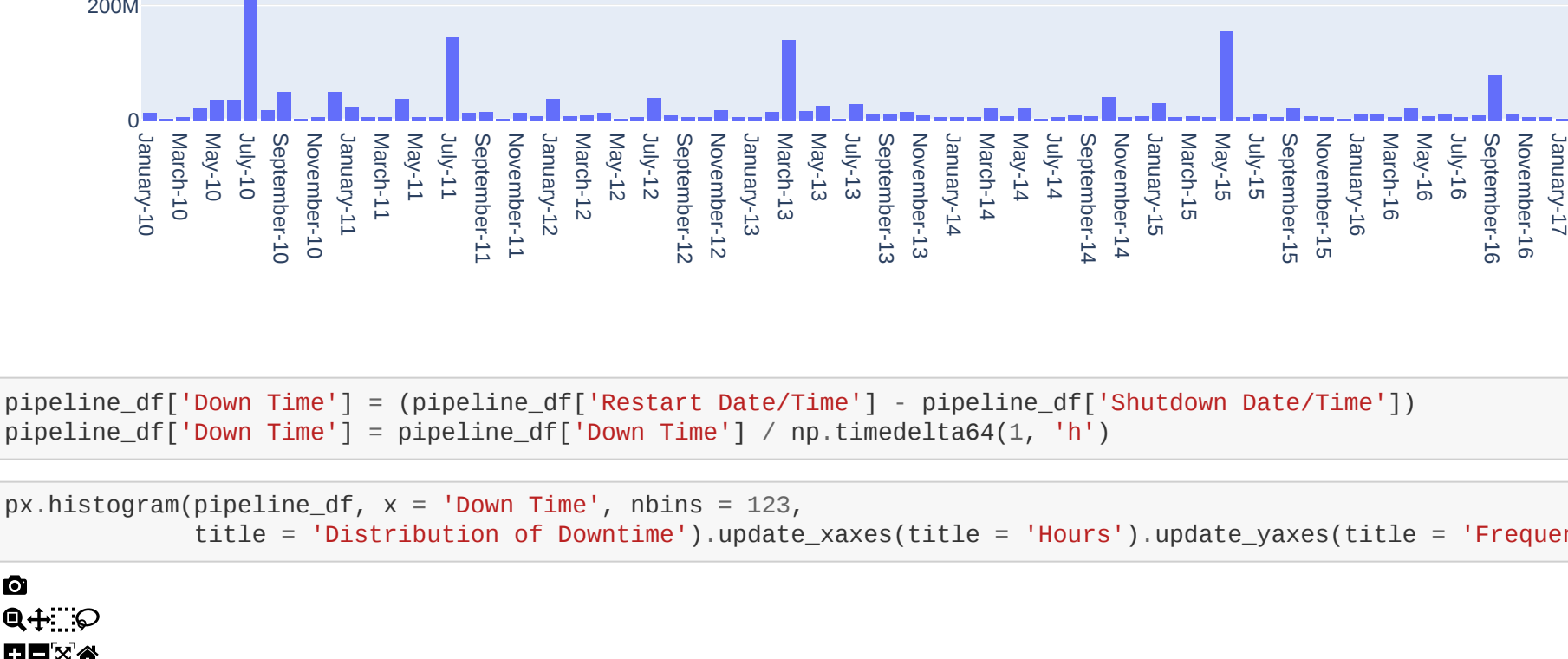
```
In [36]: px.histogram(pipeline_df, x = 'All Costs', title = 'Histogram of Cost Distribution', nbins = 10000, color = 'Acciden
t Year')
```



```
In [37]: px.bar(yearly_costs, x = yearly_costs.index,
    y = 'Ratio', title = 'Ratio of Equipment/Corrosion Costs to Total Costs').update_xaxes(title = 'Year')
```

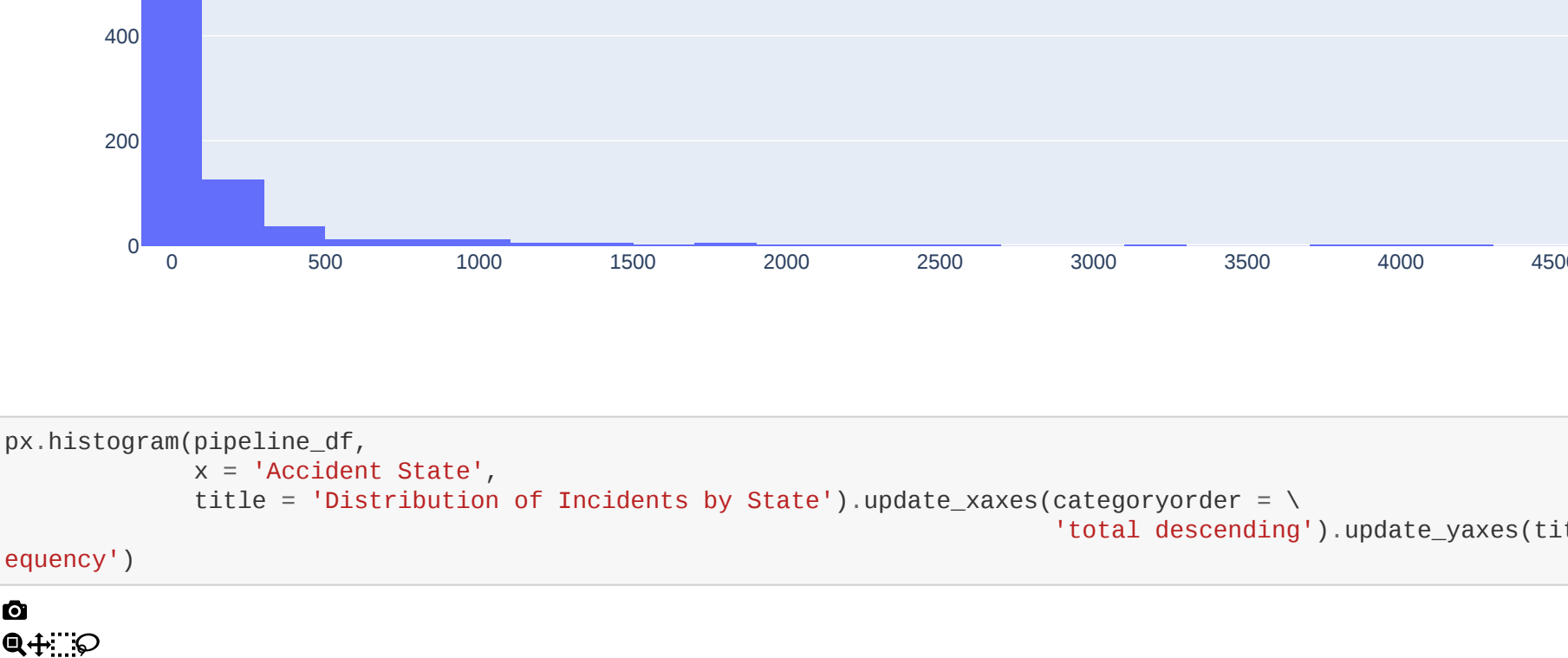


```
In [38]: px.histogram(pipeline_df, x = pipeline_df.index.strftime('%B-%y'), y = 'All Costs', title = 'Cost Over Time')
```

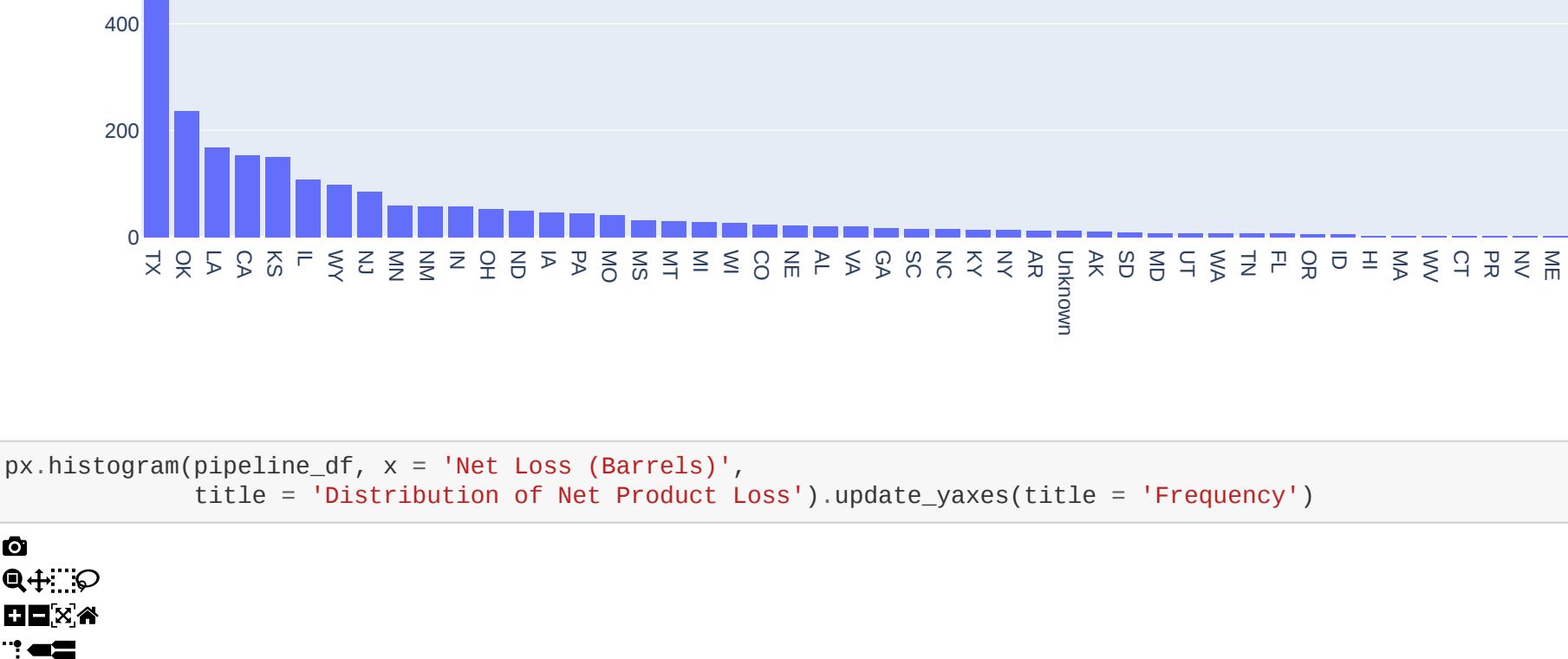


```
In [39]: pipeline_df['Down Time'] = (pipeline_df['Restart Date/Time'] - pipeline_df['Shutdown Date/Time'])
pipeline_df['Down Time'] = pipeline_df['Down Time'].dt.timedelta64(1, 'h')

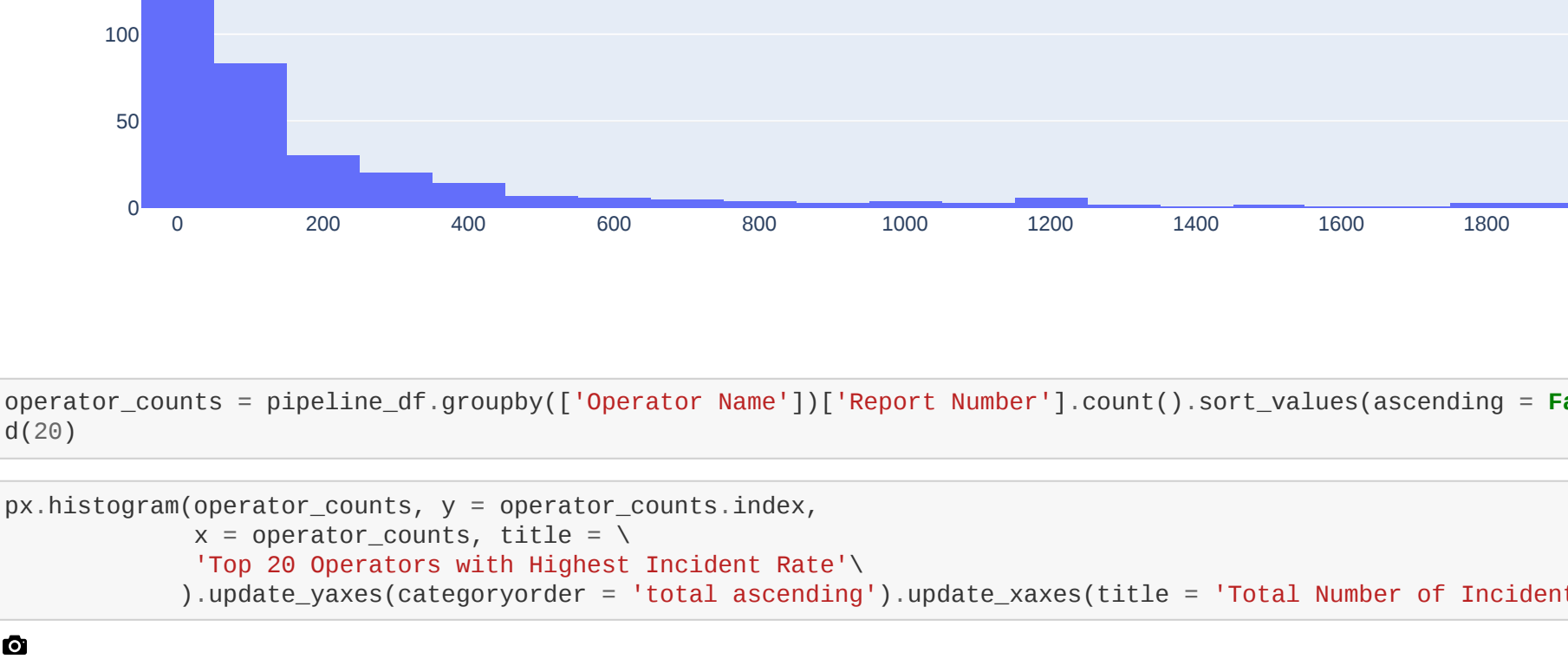
In [40]: px.histogram(pipeline_df, x = 'Down Time', nbins = 120,
    title = 'Distribution of Down Time').update_xaxes(title = 'Hours').update_yaxes(title = 'Frequency')
```



```
In [41]: px.histogram(pipeline_df,
    x = 'Accident State',
    title = 'Distribution of Incidents by State').update_yaxes(categoryorder = \
    'total ascending').update_yaxes(title = 'Fr
equency')
```

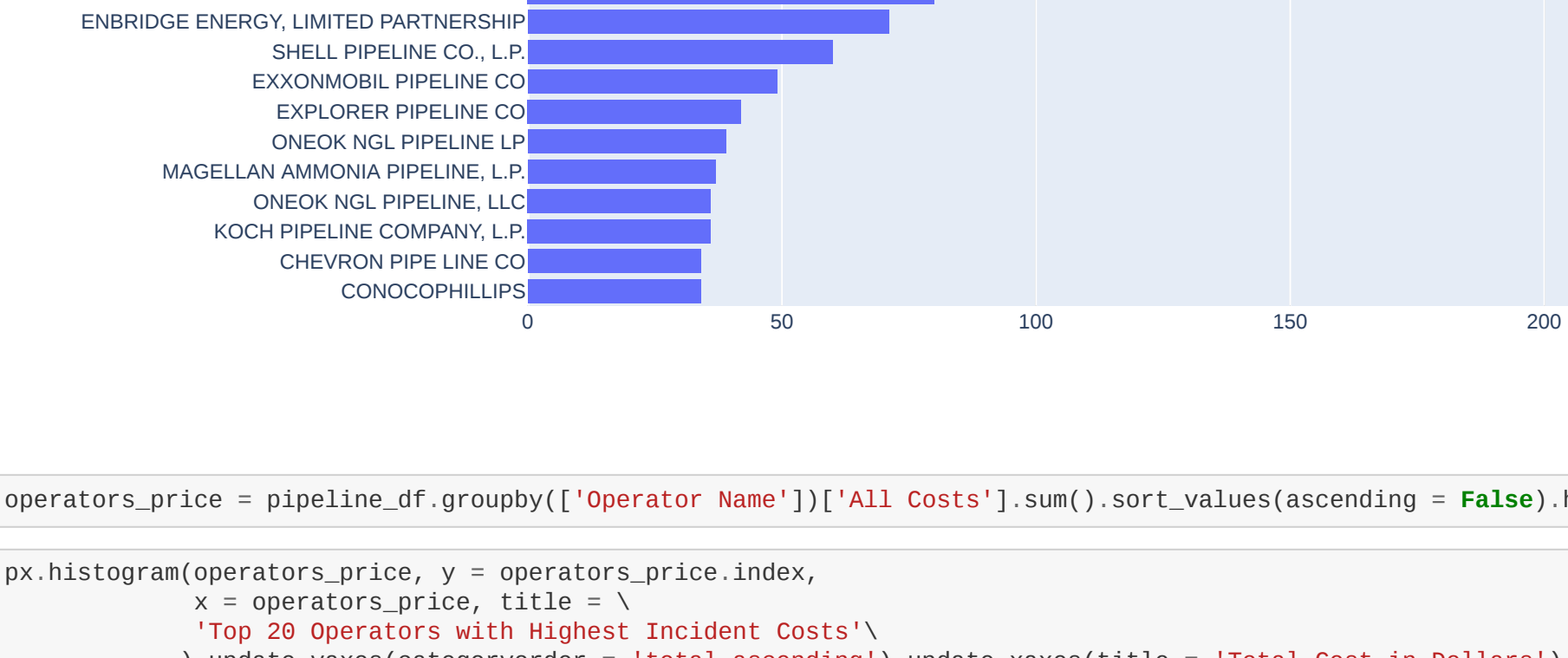


```
In [42]: px.histogram(pipeline_df, x = 'Net Loss (Barrels)',
    title = 'Distribution of Net Product Loss').update_yaxes(title = 'Frequency')
```



```
In [43]: operator_counts = pipeline_df.groupby(['Operator Name'])['Report Number'].count().sort_values(ascending = False).hea
d(20)

In [44]: px.histogram(operator_counts, y = operator_counts.index,
    x = operator_counts, title = \
    'Top 20 Operators with Highest Incident Rate'
    ).update_yaxes(categoryorder = 'total ascending').update_xaxes(title = 'Total Number of Incidents')
```



```
In [45]: operators_price = pipeline_df.groupby(['Operator Name'])['All Costs'].sum().sort_values(ascending = False).head(20)

In [46]: px.histogram(operators_price, y = operators_price.index,
    x = operators_price, title = \
    'Top 20 Operators with Highest Incident Costs'
    ).update_yaxes(categoryorder = 'total ascending').update_xaxes(title = 'Total Cost in Dollars')
```

