Financial Data Insights

Catherine Sanso & Nicholas Sanso
Aug. 25, 2023

Problem Statement

- 4,644 U.S. exchange-based companies
- Financial Statements:
  - IS, BS, SCF
  - Unusual Relationships/patterns
- Implement Unsupervised Learning
  - K-Means, DBSCAN
- Baseline

WORK FLOW

Indicates in which section of the Jupyter Notebook this info can be found 2

Data Wrangling → Clean/Merge → EDA → Model → Analysis

GICS DATASET (2)

- Imported dataset (uncleaned) as csv titled "gics.csv"
- Source: Stock Market MBA
- Dataframe: df_g
- Cleaned dataset saved to csv output titled "gics_cleaned"

SEC DATASET (2)

- Imported dataset (uncleaned) as csv titled "sec_ciksv"
- Source: SEC
- Dataframe: df_sec
- Cleaned dataset saved to csv output titled "gics_cleaned"

SECTOR MV DATASET (3)

- Imported dataset (uncleaned) as csv titled "gics_sector_chart.csv"
- Source: GICS
- Dataframe: df_mv
- Cleaned dataset saved to csv output titled "mv_cleaned"

FIN. STAT DATABASE (4)

- Dataset not imported, but scraped
- Source: Polygon.io
- Dataframe: df_poly
- Cleaned dataset saved to csv output titled "polygon_io_output_ cleaned"

CONCATENATED DATAFRAMES: SEC + GICS (3)

- Dataframe renamed "df_sec_g"
- Cleaned as an interim file

CONCATENATED DATAFRAMES: (SEC + GICS) + MV (3)

- Dataframe renamed "df_sec_g_mv"
- Saved as "sec_gics_mv_cleaned.csv"

CONCATENATED DATAFRAMES: (SEC + GICS + MV) + FIN STATS (6)

- Dataframe renamed "df"
- Saved as "df_cleaned.csv"

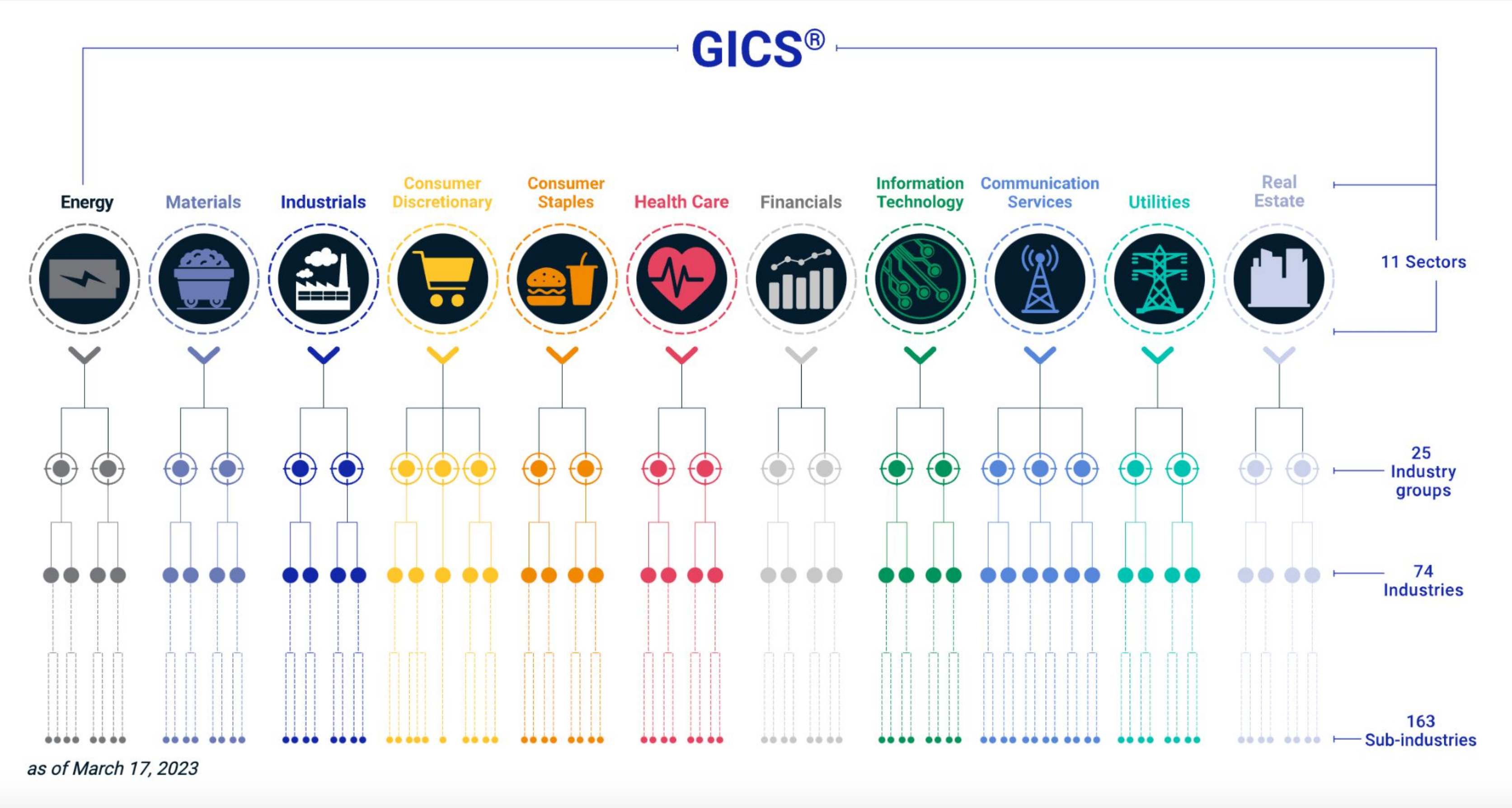
K-MEANS CLUSTERING (9)

- Unsupervised Learning Technique #1 performed on model

DBSCAN (9)

- Unsupervised Learning Technique #2 performed on model

Preprocessing



polygon.io

1

```
statements
```


```
[StockFinancial(cik='0001755672', company_name='Corteva, Inc.', end_date='2023-06-30', filing_date='2
nancials(balance_sheet={'noncurrent_assets': DataPoint(formula=None, label='Noncurrent Assets', order
6982000000.0, xpath=None), 'liabilities': DataPoint(formula=None, label='Liabilities', order=600, uni
0.0, xpath=None), 'equity': DataPoint(formula=None, label='Equity', order=1400, unit='USD', value=264
'assets': DataPoint(formula=None, label='Assets', order=100, unit='USD', value=44189000000.0, xpath=N
uity': DataPoint(formula=None, label='Liabilities And Equity', order=1900, unit='USD', value=44189000
ent_liabilities': DataPoint(formula=None, label='Current Liabilities', order=700, unit='USD', value=1
e), 'noncurrent_liabilities': DataPoint(formula=None, label='Noncurrent Liabilities', order=800, unit
0, xpath=None), 'equity_attributable_to_parent': DataPoint(formula=None, label='Equity Attributable T
it='USD', value=26220000000.0, xpath=None), 'fixed_assets': DataPoint(formula=None, label='Fixed Asse
D', value=4306000000.0, xpath=None), 'other_than_fixed_noncurrent_assets': DataPoint(formula=None, la
current Assets', order=500, unit='USD', value=22676000000.0, xpath=None), 'current_assets': DataPoint
rent Assets', order=200, unit='USD', value=17207000000.0, xpath=None), 'equity_attributable_to_noncon
oint(formula=None, label='Equity Attributable To Noncontrolling Interest', order=1500, unit='USD', va
ne)}, cash_flow_statement=CashFlowStatement(exchange_gains_losses=ExchangeGainsLosses(formula=None, l
es', order=1000, unit='USD', value=11000000.0, xpath=None), net_cash_flow=NetCashFlow(formula=None, l
er=1100, unit='USD', value=895000000.0, xpath=None), net_cash_flow_from_financing_activities=NetCashF
s(formula=None, label='Net Cash Flow From Financing Activities', order=700, unit='USD', value=1050000
ehensive_income=ComprehensiveIncome(comprehensive_income_loss=ComprehensiveIncomeLoss(formula=None, l
e/Loss', order=100, unit='USD', value=779000000.0, xpath=None), comprehensive_income_loss_attributabl
IncomeLossAttributableToParent(formula=None, label='Comprehensive Income/Loss Attributable To Parent'
```


2

```python
for i in range(len(statements)):
    try:
        dict_fs = {}
        dict_fs["cik"] = statements[i].cik
        dict_fs["company_name"] = statements[i].company_name
        dict_fs["fiscal_period"] = statements[i].fiscal_period
        dict_fs["fiscal_year"] = statements[i].fiscal_year
        dict_fs["filing_date"] = statements[i].filing_date


        # Calling attributes of the Income Statement:
        for attr in attributes_is:
            try:
                financials = statements[i].financials.income_statement
                attr_obj = getattr(financials, attr)
                dict_fs["is_" + attr + "_unit"] = attr_obj.unit
                dict_fs["is_" + attr + "_value"] = attr_obj.value
            except Exception as e:
                pass

        # Calling attributes of the Comprehensive Income Statement:
        for attr in attributes_ci:
            try:
                financials = statements[i].financials.comprehensive_income
                attr_obj = getattr(financials, attr)
                dict_fs["ci_" + attr + "_unit"] = attr_obj.unit
                dict_fs["ci_" + attr + "_value"] = attr_obj.value
            except Exception as e:
                pass

        # Calling attributes of the Cash Flow Statement:
        for attr in attributes_cfs:
            try:
                financials = statements[i].financials.cash_flow_statement
                attr_obj = getattr(financials, attr)
                dict_fs["cfs_" + attr + "_unit"] = attr_obj.unit
                dict_fs["cfs_" + attr + "_value"] = attr_obj.value
            except Exception as e:
                pass
```

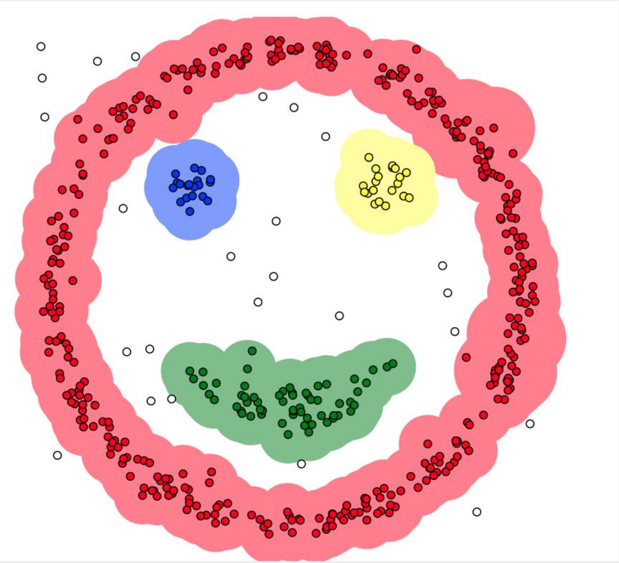
3

```
df_poly
```

	cik	company_name	filing_date	is_basic_earnings_per_share_unit	is_basic_earnings_per_share_value
5154	1705843	Calyxt, Inc.	2023-05-01	USD / shares	-1.09
3367	96021	SYSCO CORP	2023-05-02	USD / shares	0.85
6261	1747748	Qualtrics International Inc.	2023-05-02	USD / shares	-0.43
6737	1853021	Metals Acquisition Corp	2023-05-02	NaN	NaN

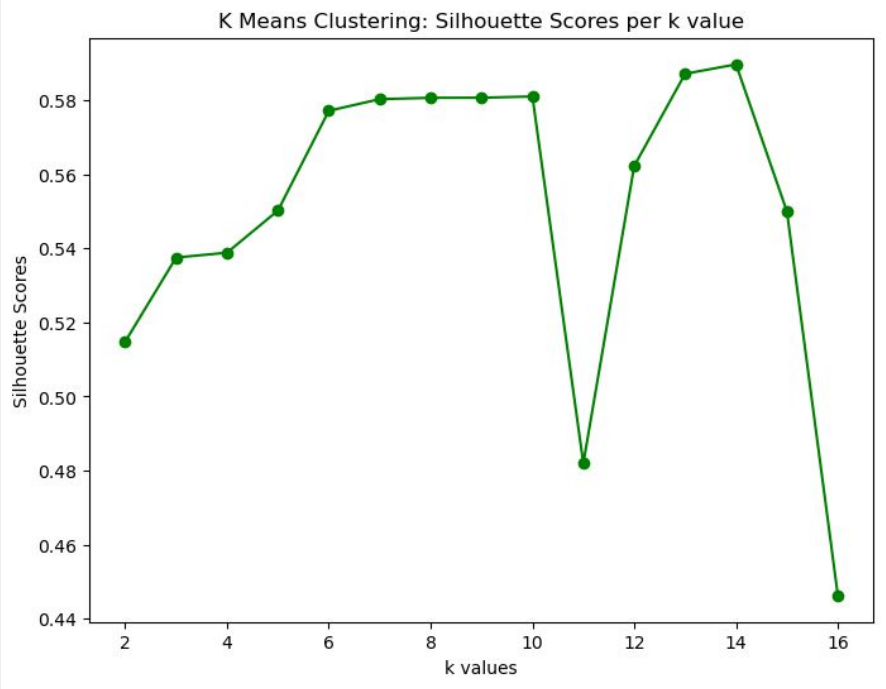
Model Selection

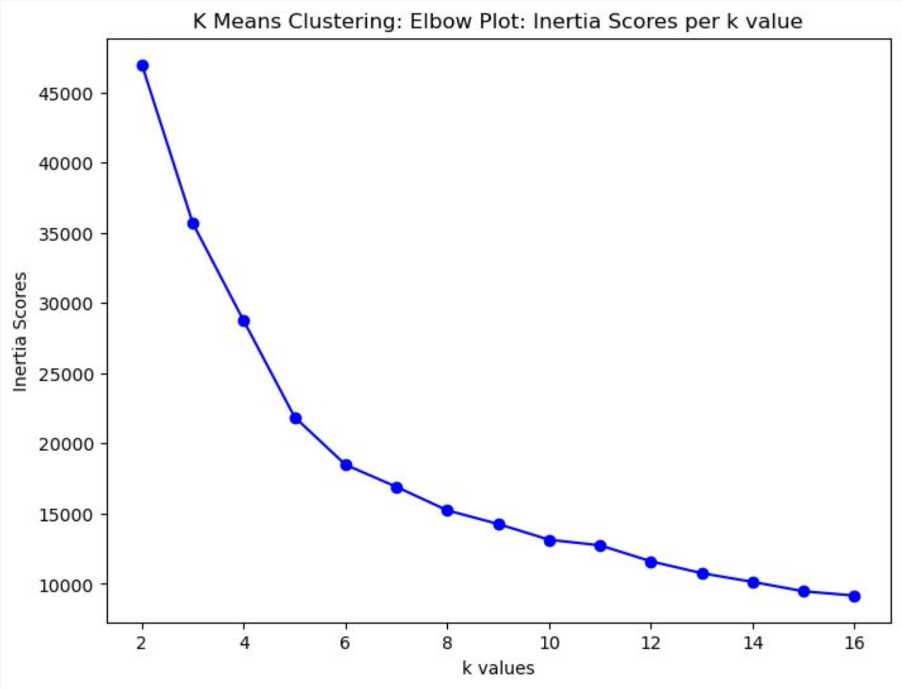
- DBSCAN is robust to outliers, important in right skewed financial datasets
- DBSCAN solves for cluster parameter on its own
- Can identify clusters of arbitrary shapes and sizes



Model Performance

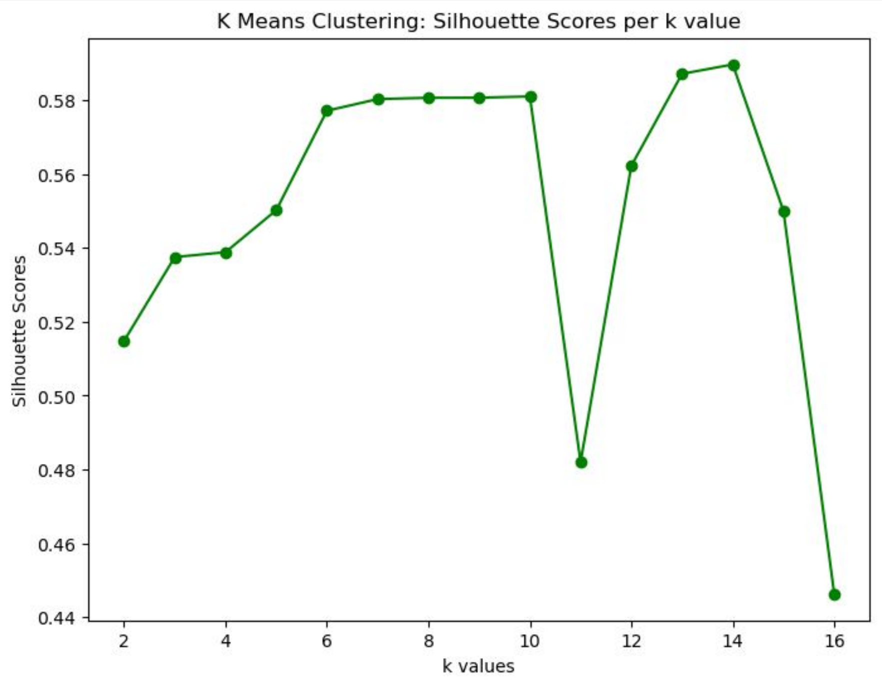
- Silhouette Score for k-means clusters optimizes at 6

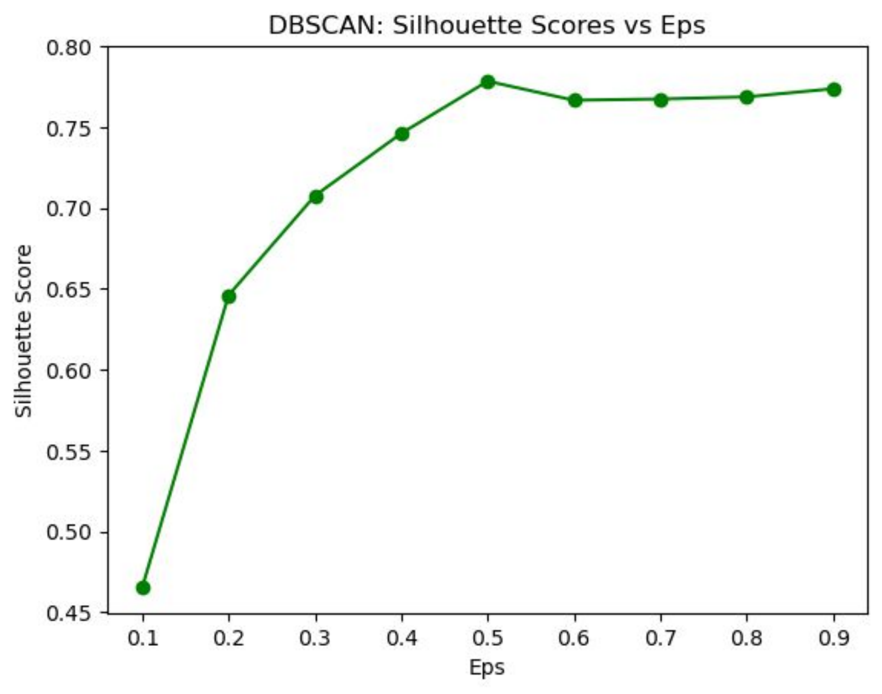




Model Performance

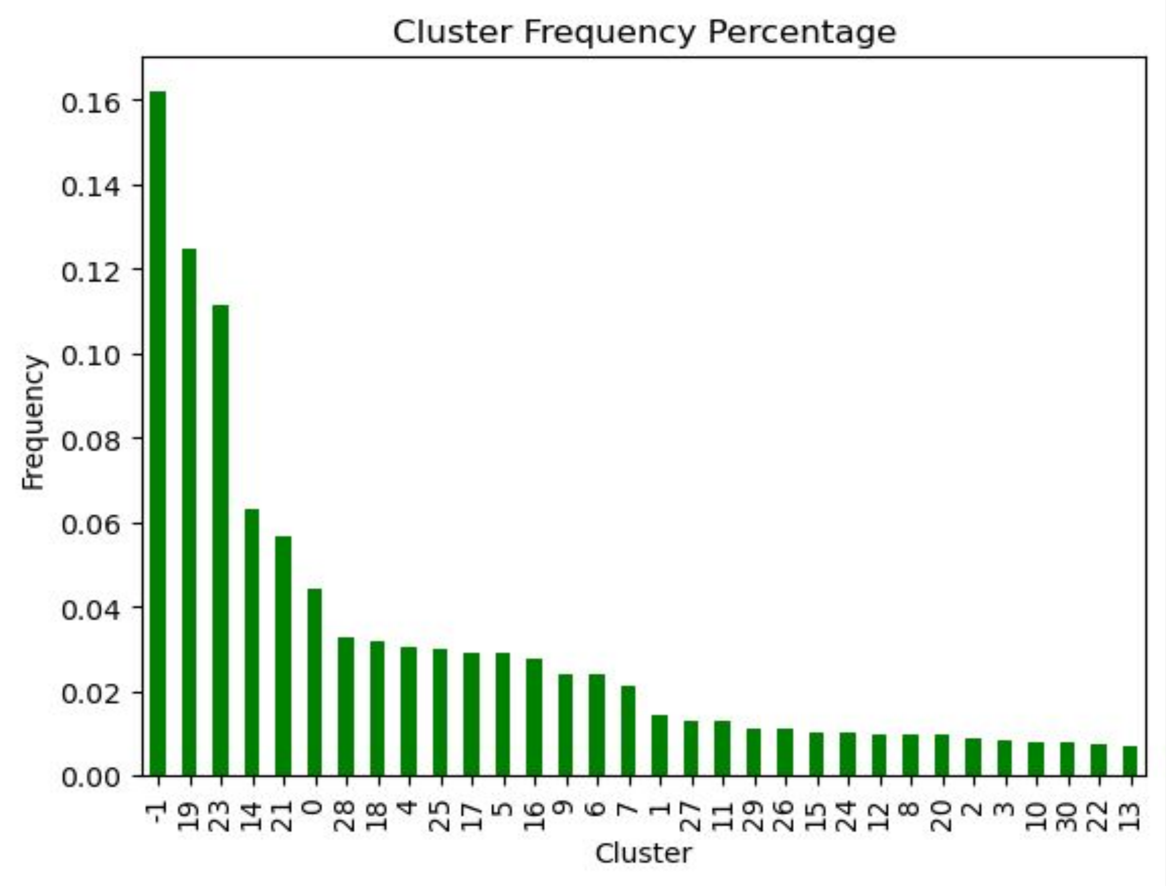
- K-means max Silhouette Score was 58
- DBScan max Silhouette Score was ~.75





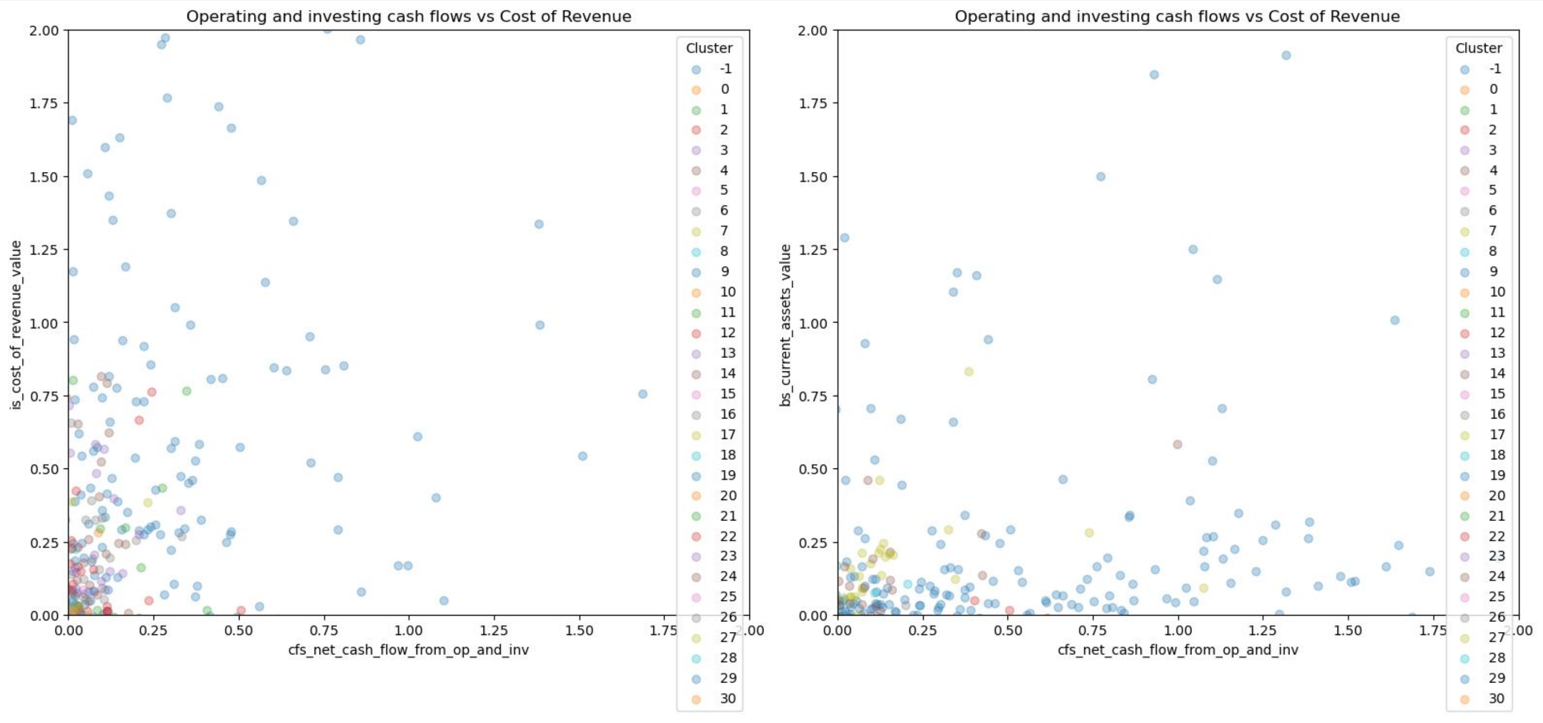
EDA: Cluster Frequencies

- Noise cluster is 16% of the data
- Steep drop-off in frequency after the first three clusters
  - Possibly driven by two most important variables in finance, growth and required rate of return



EDA: Scatterplots

- Relatively robust at identifying outliers and placing them into the “-1” cluster
- Relatively poor at preventing seemingly clustered data from being labeled as noise



Conclusion

- There is no loss function by which to measure unsupervised learning models because there is no target variable
- DBSCAN outperformed K-means clustering at creating distinctive clusters, with a higher Silhouette score of 0.75.
  - DBSCAN did not assume the clusters to be spherical (unlike K-Means)
- Our DBSCAN model was moderately successful in that distinct clusters were formed but we could not reveal insightful financial relationships from the cluster formations.
- Our project showcases the difficulty in fitting clustering models and interpreting financial insights.

Next Steps

- Include other clustering models which adapt better to concavity, geometry flatness, even/uneven cluster size. Examples of other models to explore include Agglomerative Hierarchical Clustering and Gaussian Mixture Models.
- Leverage clustered features for a supervised learning regression model.
- Use GICS subindustries to create more accurate imputations for missing values.
- Gather more data to impute fewer data points