Problem Statement

How can banks identify and predict future fraudulent transactions? Out of the provided 28 features, which features will identify frauds with high confidence levels?

Data Cleaning

- Some revenue data was arranged by Months, instead of year. The ‘agg’ function was useful in arranging data by year

```
YtRevAd = YtRev.groupby(YtRev["Date"].dt.year)['Ads Revenue (Mn)'].agg(sum)
```

- Slicing was useful when only the Year needed to be stored

```
disRev['Year'] = disRev['Year'].str[-4:]
```

- The data from different companies came in different csv files and the merge operation was used a lot to combine multiple data frames to create a master data frame

```
df1 = disRevdf.merge(NetRev, on="Year", how="outer", suffixes=('_disney','_netflix'))
```

- Most missing data was fixed by looking up various websites for revenue and subscriber numbers.
- Any more null and not available data were replaced with 0 which just meant the company was non-existent then.

```
df.replace(np.nan, 0, inplace=True)
```

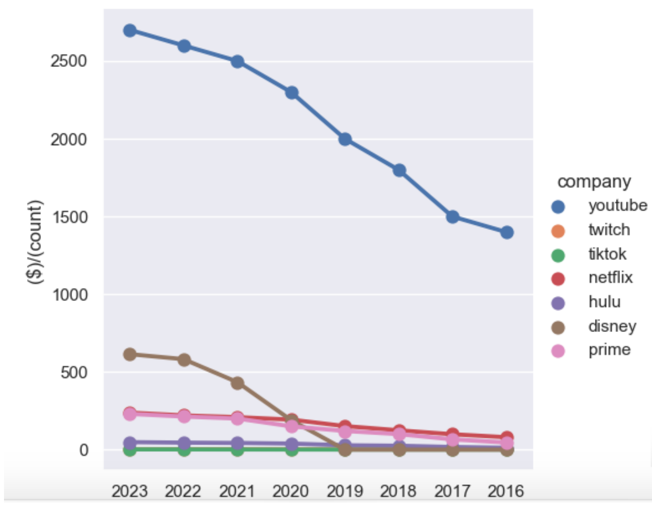
Exploratory Data Analysis

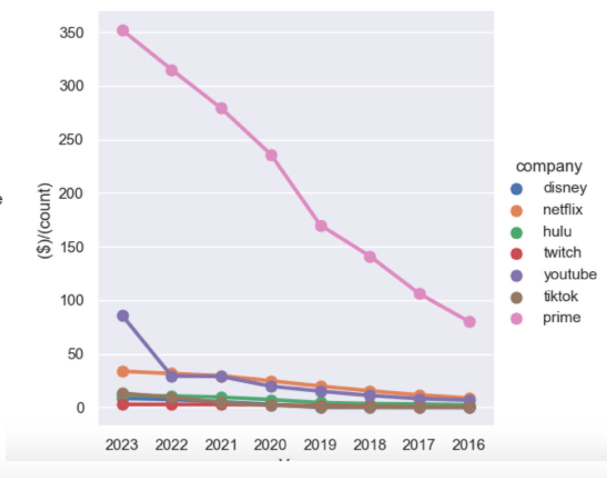
Explore the data using describe(), info(), dtypes and other methods and attributes available on the data frame

Melt and realign the data in a plottable format

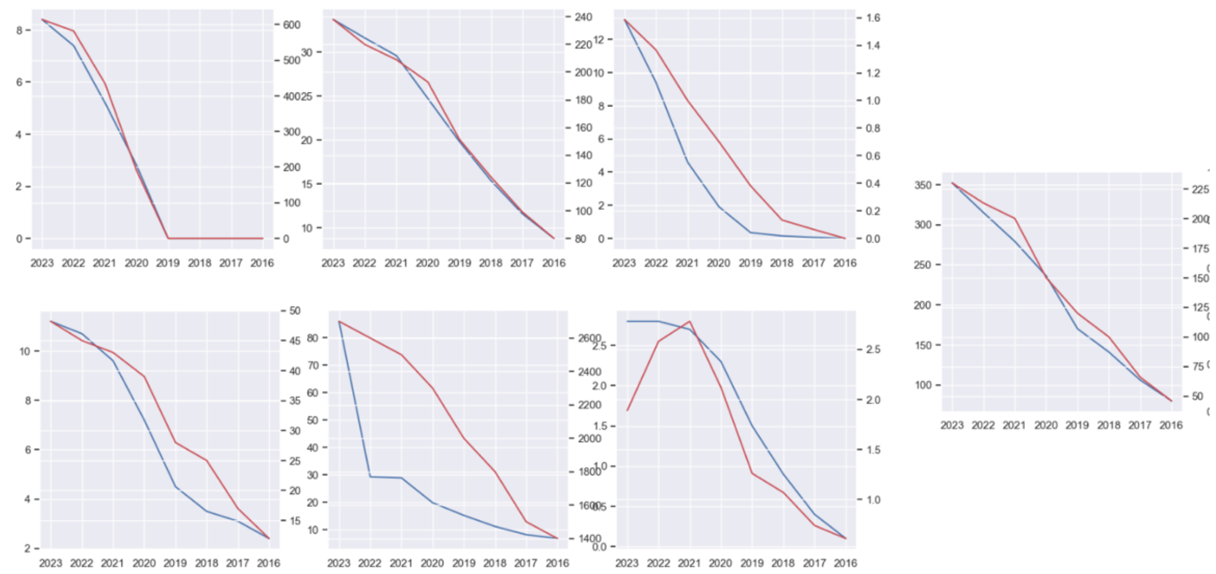
```
df_T_clean = pd.melt(df_T,id_vars= ['company data'])

df_T_clean = df_T_clean.rename(columns={0:'Year', 'value':
'Revenue($)/Subscribers(count)'})

df_T_clean['type'] = df_T_clean['company data'].str.split('_').str[0]

df_T_clean['company'] = df_T_clean['company data'].str.split('_').str[1]
```

This chart shows the revenue and subscribers growth by the year and for each company. We can look more closely at each company as shown in the next plot





The next chart shows how each of the company grew over the past 7 years



The above charts and data helps us formulate the following hypothesis

Hypothesis

Based on the above plots, it seems like # of subscribers and revenue generated are related positively.

Null Hypothesis - Number of subscribers does not influence the revenue generated

We can calculate the p-value to see if this holds

Pearson's co-efficient shows a negative correlation between the two variables as shown below. The [0,1] array gives the results we want

corr_mat = np.corrcoef(x,y)

Permutation replicates on Pearson's co-eff also gives a very high p-value.

Next the linear regression to find the slope and intercept to establish a relationship is used. This is no clear relationship

bs_slope_reps, bs_intercept_reps = draw_bs_pairs_linreg(df_T_sub_series, df_T_rev_series, 1000)

```
# Compute and print 95% CI for slope

print(np.percentile(bs_slope_reps, [2.5, 97.5]))
```

Hypothesis Conclusion:

The p-value is very high and so we cannot reject the null.

Also, the slope and intercept graph does not show strong correlation between the two variables.

Assumption for our model:

The revenue listed is from various sources, especially for Amazon Prime, whose parent company is a conglomerate, unlike the other companies listed but much like our company XYZ.

This means XYZ should be safe to implement a new subscription model given its vast customer base and availability of original content.

Pre-processing for modeling

- The first step in this was to create dummies for categorical data

```
dfo=df.select_dtypes(include=['object']) # select object type columns

df = pd.concat([df.drop(dfo, axis=1), pd.get_dummies(dfo)], axis=1)
```

Based on the data set obtained, divide the data into test and training data sets

- Pick the dependent and independent variable. Revenue, here is the dependent variable

```
X_train, X_test, y_train, y_test=train_test_split(X, y,
                                  test_size = 0.25,
                                    random_state = 246)
```

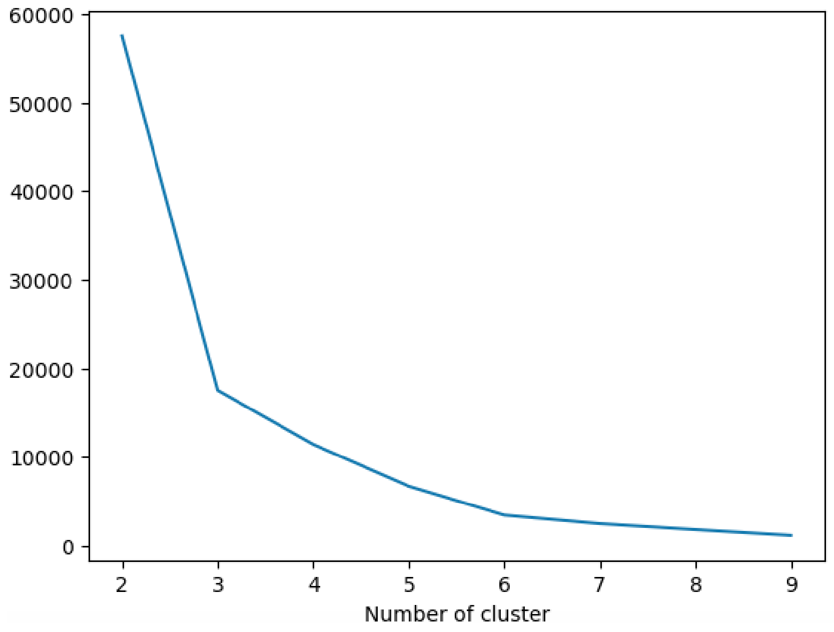
Modeling

K-means clustering to predict revenue and subscribers

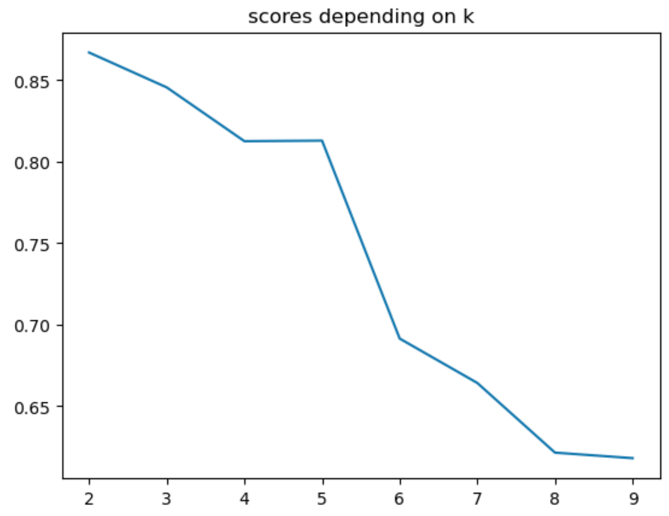
The knee-elbow curve which predicts the best number of clusters looks like this when either revenue or number of subscribers is the dependent variable.

The code to generate the centroids, the silhouette scores is as below:

```
for k in range(2, 10):
  kmeans = KMeans(n_clusters=k, max_iter=1000).fit(x_cols)
  #x_cols["clusters"] = kmeans.labels_
  sse[k] = kmeans.inertia_
  score[k] = silhouette_score(x_cols, kmeans.labels_)
```



Silhouette scores:



Centroids:

centroids = model.cluster_centers_

```
array([[  8.79391489],
       [295.5      ],
       [116.6      ]])
```

Conclusion based on K-means

Prediction for XYZ:

Revenue of 116M with 2.56 M subscribers

Silouette Score was 86% which is high, saying that this is a good model which shows clustering around the means

Linear Regression

Linear regression model is created and the RMSE score

```
rModel = linear_model.LinearRegression()
```

RMSE is around 50% for this model

```
np.sqrt(((predictions - targets) ** 2).mean())
```

Conclusion 2:

RMSE is about 50 i.e if we use subscribers to predict revenue we may be right 50% of the times, which is as good as taking a random guess or a coin toss.

Number of subcribers will not be a good prdictor for Ad Revenue