Data Explore

Polish companies bankruptcy data Data Set

https://archive.ics.uci.edu/ml/datasets/Polish+companies+bankruptcy+data (https://archive.ics.uci.edu/ml/datasets/Polish+companies+bankruptcy+data)

Abstract:

The dataset is about bankruptcy prediction of Polish companies.The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013.

Source:

Creator: Sebastian Tomczak -- Department of Operations Research, Wroclaw University of Science and Technology, Poland

Donor: Sebastian Tomczak, Maciej Zieba, Jakub M. Tomczak

Data Set Information:

The dataset is about bankruptcy prediction of Polish companies. The data was collected from Emerging Markets Information Service (EMIS, [Web Link]), which is a database containing information on emerging markets around the world. The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013. Basing on the collected data five classification cases were distinguished, that depends on the forecasting period:

- 1stYear: the data contains financial rates from 1st year of the forecasting period and corresponding class label that indicates bankruptcy status after 5 years. The data contains 7027 instances (financial statements), 271 represents bankrupted companies, 6756 firms that did not bankrupt in the forecasting period.
- 2ndYear: the data contains financial rates from 2nd year of the forecasting period and corresponding class label that indicates bankruptcy status after 4 years. The data contains 10173 instances (financial statements), 400 represents bankrupted companies, 9773 firms that did not bankrupt in the forecasting period.
- 3rdYear: the data contains financial rates from 3rd year of the forecasting period and corresponding class label that indicates bankruptcy status after 3 years. The data contains 10503 instances (financial statements), 495 represents bankrupted companies, 10008 firms that did not bankrupt in the forecasting period.
- 4thYear: the data contains financial rates from 4th year of the forecasting period and corresponding class label that indicates bankruptcy status after 2 years. The data contains 9792 instances (financial statements), 515 represents bankrupted companies, 9277 firms that did not bankrupt in the forecasting period.
- 5thYear: the data contains financial rates from 5th year of the forecasting period and corresponding class label that indicates bankruptcy status after 1 year. The data contains 5910

instances (financial statements), 410 represents bankrupted companies, 5500 firms that did not bankrupt in the forecasting period.

Attribute Information:

- X1 net profit / total assets
- X2 total liabilities / total assets
- X3 working capital / total assets
- X4 current assets / short-term liabilities
- X5 [(cash + short-term securities + receivables - short-term liabilities) / (operating expenses - depreciation)] * 365
- X6 retained earnings / total assets
- X7 EBIT / total assets
- X8 book value of equity / total liabilities
- X9 sales / total assets
- X10 equity / total assets
- X11 (gross profit + extraordinary items + financial expenses) / total assets
- X12 gross profit / short-term liabilities
- X13 (gross profit + depreciation) / sales
- X14 (gross profit + interest) / total assets
- X15 (total liabilities * 365) / (gross profit + depreciation)
- X16 (gross profit + depreciation) / total liabilities
- X17 total assets / total liabilities
- X18 gross profit / total assets
- X19 gross profit / sales
- X20 (inventory * 365) / sales
- X21 sales (n) / sales (n-1)
- X22 profit on operating activities / total assets
- X23 net profit / sales
- X24 gross profit (in 3 years) / total assets
- X25 (equity - share capital) / total assets
- X26 (net profit + depreciation) / total liabilities
- X27 profit on operating activities / financial expenses
- X28 working capital / fixed assets
- X29 logarithm of total assets
- X30 (total liabilities - cash) / sales
- X31 (gross profit + interest) / sales
- X32 (current liabilities * 365) / cost of products sold
- X33 operating expenses / short-term liabilities
- X34 operating expenses / total liabilities
- X35 profit on sales / total assets
- X36 total sales / total assets
- X37 (current assets - inventories) / long-term liabilities
- X38 constant capital / total assets
- X39 profit on sales / sales
- X40 (current assets - inventory - receivables) / short-term liabilities
- X41 total liabilities / ((profit on operating activities + depreciation) * (12/365))
- X42 profit on operating activities / sales
- X43 rotation receivables + inventory turnover in days

- X44 (receivables * 365) / sales
- X45 net profit / inventory
- X46 (current assets - inventory) / short-term liabilities
- X47 (inventory * 365) / cost of products sold
- X48 EBITDA (profit on operating activities - depreciation) / total assets
- X49 EBITDA (profit on operating activities - depreciation) / sales
- X50 current assets / total liabilities
- X51 short-term liabilities / total assets
- X52 (short-term liabilities * 365) / cost of products sold)
- X53 equity / fixed assets
- X54 constant capital / fixed assets
- X55 working capital
- X56 (sales - cost of products sold) / sales
- X57 (current assets - inventory - short-term liabilities) / (sales - gross profit - depreciation)
- X58 total costs /total sales
- X59 long-term liabilities / equity
- X60 sales / inventory
- X61 sales / receivables
- X62 (short-term liabilities *365) / sales
- X63 sales / short-term liabilities
- X64 sales / fixed assets

Load Libraries

In [40]:
```
# Import base libraries
import pandas as pd
from scipy.io import arff
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Load Data

In [41]:
```
# Load all five data files

data1 = arff.loadarff('data/1year.arff')
df1 = pd.DataFrame(data1[0])

data2 = arff.loadarff('data/2year.arff')
df2 = pd.DataFrame(data2[0])

data3 = arff.loadarff('data/3year.arff')
df3 = pd.DataFrame(data3[0])

data4 = arff.loadarff('data/4year.arff')
df4 = pd.DataFrame(data4[0])

data5 = arff.loadarff('data/5year.arff')
df5 = pd.DataFrame(data5[0])
```

In [42]:
```
# Size of datasets

print('Size of datasets')
print("Data 1 (Year1):", len(df1))
print("Data 2 (Year2):", len(df2))
print("Data 3 (Year3):", len(df3))
print("Data 4 (Year4):", len(df4))
print("Data 5 (Year5):", len(df5))
```

```
Size of datasets
Data 1 (Year1): 7027
Data 2 (Year2): 10173
Data 3 (Year3): 10503
Data 4 (Year4): 9792
Data 5 (Year5): 5910
```

In [43]:
```
# Data 1, first five rows
df3.head()
```

Out[43]:

	Attr1	Attr2	Attr3	Attr4	Attr5	Attr6	Attr7	Attr8	Attr9	Attr10	...	
0	0.174190	0.41299	0.14371	1.3480	-28.9820	0.60383	0.219460	1.1225	1.1961	0.46359	...	0.1
1	0.146240	0.46038	0.28230	1.6294	2.5952	0.00000	0.171850	1.1721	1.6018	0.53962	...	0.0
2	0.000595	0.22612	0.48839	3.1599	84.8740	0.19114	0.004572	2.9881	1.0077	0.67566	...	0.0
3	0.024526	0.43236	0.27546	1.7833	-10.1050	0.56944	0.024526	1.3057	1.0509	0.56453	...	0.0
4	0.188290	0.41504	0.34231	1.9279	-58.2740	0.00000	0.233580	1.4094	1.3393	0.58496	...	0.1

5 rows × 65 columns

In [44]:
```
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10503 entries, 0 to 10502
Data columns (total 65 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Attr1   10503 non-null  float64
 1   Attr2   10503 non-null  float64
 2   Attr3   10503 non-null  float64
 3   Attr4   10485 non-null  float64
 4   Attr5   10478 non-null  float64
 5   Attr6   10503 non-null  float64
 6   Attr7   10503 non-null  float64
 7   Attr8   10489 non-null  float64
 8   Attr9   10500 non-null  float64
 9   Attr10  10503 non-null  float64
 10  Attr11  10503 non-null  float64
 11  Attr12  10485 non-null  float64
 12  Attr13  10460 non-null  float64
 13  Attr14  10503 non-null  float64
 14  Attr15  10495 non-null  float64
 15  Attr16  10489 non-null  float64
 16  Attr17  10489 non-null  float64
 17  Attr18  10503 non-null  float64
 18  Attr19  10460 non-null  float64
 19  Attr20  10460 non-null  float64
 20  Attr21  9696 non-null   float64
 21  Attr22  10503 non-null  float64
 22  Attr23  10460 non-null  float64
 23  Attr24  10276 non-null  float64
 24  Attr25  10503 non-null  float64
 25  Attr26  10489 non-null  float64
 26  Attr27  9788 non-null   float64
 27  Attr28  10275 non-null  float64
 28  Attr29  10503 non-null  float64
 29  Attr30  10460 non-null  float64
 30  Attr31  10460 non-null  float64
 31  Attr32  10402 non-null  float64
 32  Attr33  10485 non-null  float64
 33  Attr34  10489 non-null  float64
 34  Attr35  10503 non-null  float64
 35  Attr36  10503 non-null  float64
 36  Attr37  5767 non-null   float64
 37  Attr38  10503 non-null  float64
 38  Attr39  10460 non-null  float64
 39  Attr40  10485 non-null  float64
 40  Attr41  10301 non-null  float64
 41  Attr42  10460 non-null  float64
 42  Attr43  10460 non-null  float64
 43  Attr44  10460 non-null  float64
 44  Attr45  9912 non-null   float64
 45  Attr46  10485 non-null  float64
 46  Attr47  10417 non-null  float64
 47  Attr48  10503 non-null  float64
 48  Attr49  10460 non-null  float64
 49  Attr50  10489 non-null  float64
```

```
 50  Attr51  10503 non-null  float64
 51  Attr52  10417 non-null  float64
 52  Attr53  10275 non-null  float64
 53  Attr54  10275 non-null  float64
 54  Attr55  10503 non-null  float64
 55  Attr56  10460 non-null  float64
 56  Attr57  10503 non-null  float64
 57  Attr58  10474 non-null  float64
 58  Attr59  10503 non-null  float64
 59  Attr60  9911 non-null   float64
 60  Attr61  10486 non-null  float64
 61  Attr62  10460 non-null  float64
 62  Attr63  10485 non-null  float64
 63  Attr64  10275 non-null  float64
 64  class   10503 non-null  object
dtypes: float64(64), object(1)
memory usage: 5.2+ MB
```

In [45]:
```
df3.describe()
```

Out[45]:

	Attr1	Attr2	Attr3	Attr4	Attr5	Attr6	
count	10503.000000	10503.000000	10503.000000	10485.000000	1.047800e+04	10503.000000	10503
mean	0.052844	0.619911	0.095490	9.980499	-1.347662e+03	-0.121159	0
std	0.647797	6.427041	6.420056	523.691951	1.185806e+05	6.970625	0
min	-17.692000	0.000000	-479.730000	0.002080	-1.190300e+07	-508.120000	-17
25%	0.000686	0.253955	0.017461	1.040100	-5.207075e+01	0.000000	0
50%	0.043034	0.464140	0.198560	1.605600	1.579300e+00	0.000000	0
75%	0.123805	0.689330	0.419545	2.959500	5.608400e+01	0.072584	0
max	52.652000	480.730000	17.708000	53433.000000	6.854400e+05	45.533000	52

8 rows × 64 columns

Change the label/class dytpe

In [46]:
```
# Class
df3['class'].unique()
```

Out[46]:
```
array([b'0', b'1'], dtype=object)
```

In [47]:
```
# Convert class/label type to binary

df1['class'] = df1['class'].astype('int64')
df2['class'] = df2['class'].astype('int64')
df3['class'] = df3['class'].astype('int64')
df4['class'] = df4['class'].astype('int64')
df5['class'] = df5['class'].astype('int64')
```

In [48]:
```
df3['class'].unique()
```

Out[48]:
```
array([0, 1])
```

Check class distribution

In [49]:
```
# Class Distribution

df3['class'].value_counts(normalize=True)

df_list = [df1, df2, df3, df4, df5]

for i, df in enumerate(df_list, start=1):
    val_counts = df['class'].value_counts()
    print(f'Data {i}:')
    print('Total:', len(df))
    print(val_counts, '\n-------\n')
```

```
Data 1:
Total: 7027
0    6756
1     271
Name: class, dtype: int64
-------

Data 2:
Total: 10173
0    9773
1     400
Name: class, dtype: int64
-------

Data 3:
Total: 10503
0    10008
1      495
Name: class, dtype: int64
-------

Data 4:
Total: 9792
0    9277
1     515
Name: class, dtype: int64
-------

Data 5:
Total: 5910
0    5500
1     410
Name: class, dtype: int64
-------
```

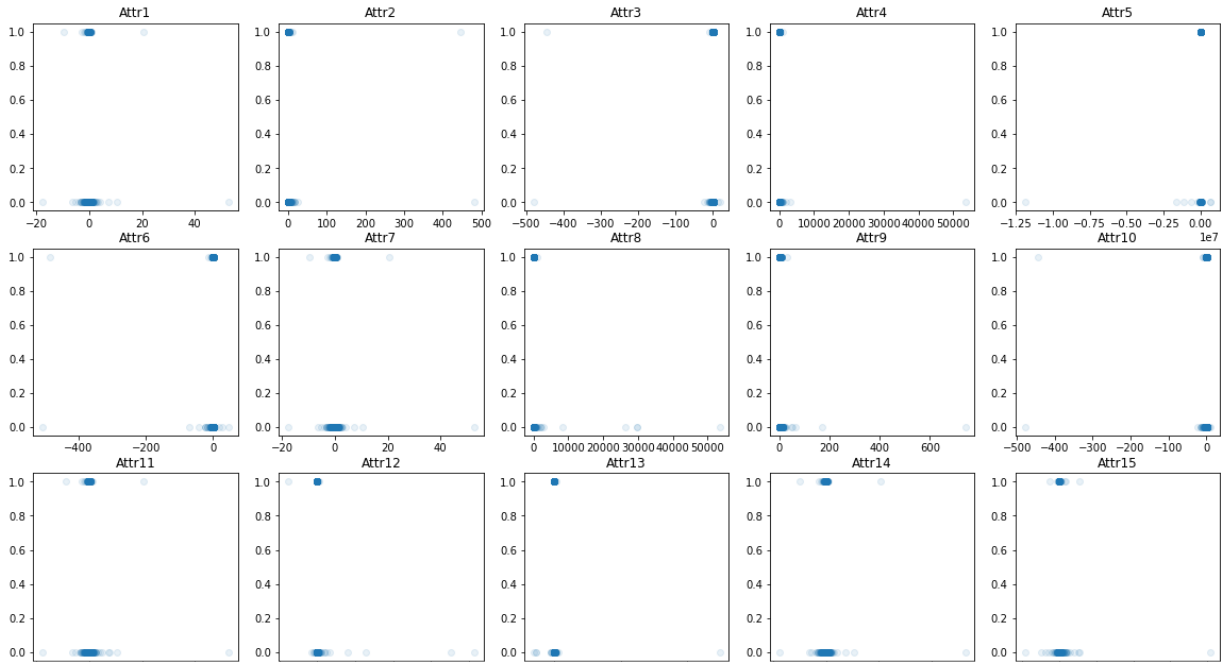
Visualizations

In [50]:
```
# Scatter Graphs

plt.figure(figsize=(20, 50))

for i, col in enumerate(df3.columns, start=1):
    plt.subplot(13, 5, i)
    plt.scatter(df3[col], df3['class'], alpha=0.1)
    plt.title(col)

plt.savefig('figures/scatter_d3.png')
```

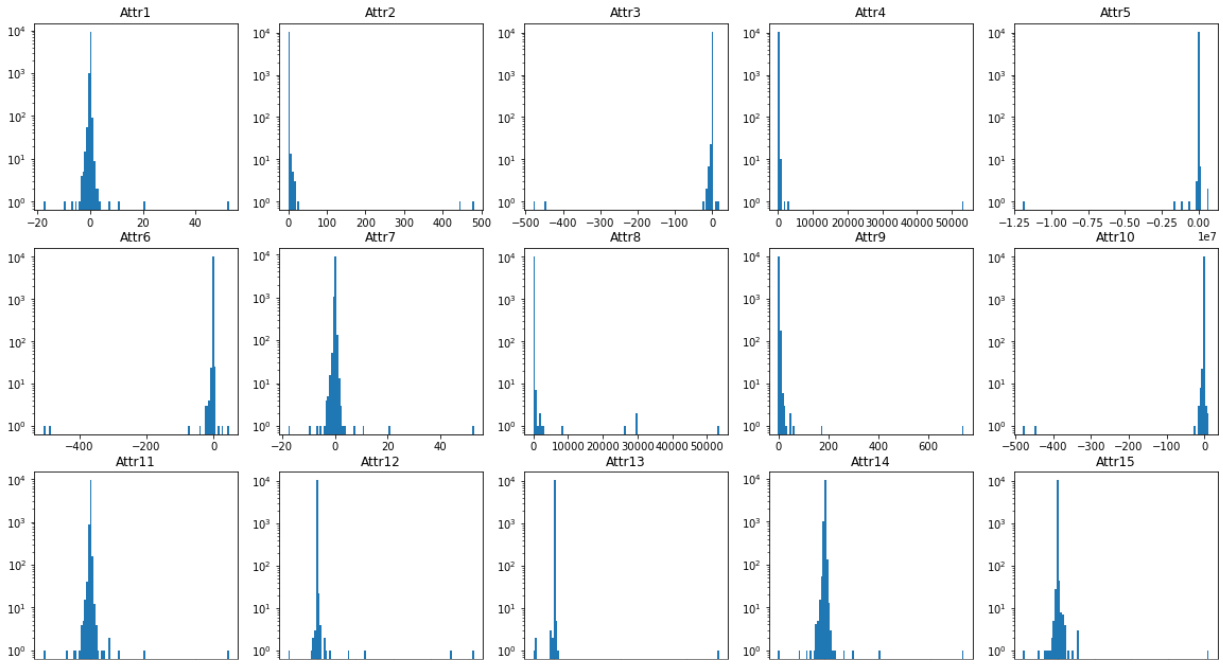


In [51]:
```
# Histograms

plt.figure(figsize=(20, 50))

for i, col in enumerate(df3.columns, start=1):
    plt.subplot(13, 5, i)
    plt.hist(df3[col], bins=100, log=True)
    plt.title(col)

plt.savefig('figures/hist_d3_log.png')
```



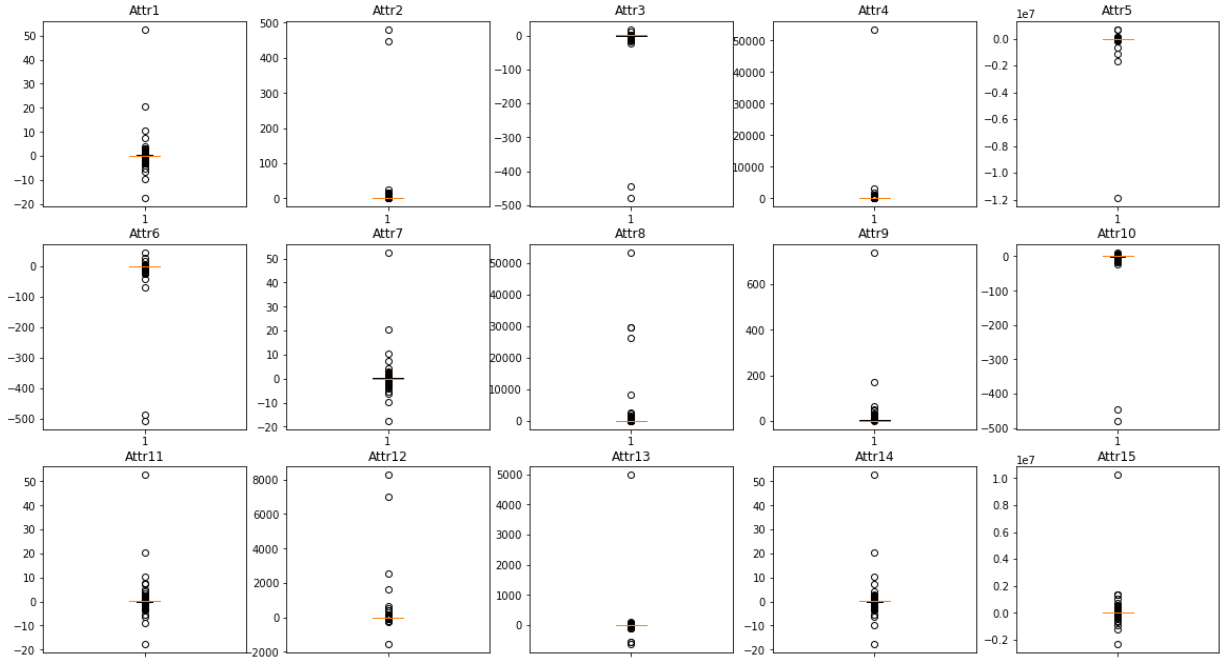
In [55]:
```
# Box plots

plt.figure(figsize=(20, 50))

for i, col in enumerate(df3.columns, start=1):
    plt.subplot(13, 5, i)
    plt.boxplot(df3[col].dropna())
    plt.title(col)

plt.savefig('figures/boxPlot_d3.png')

## Boxplot only draws the attribute only when missing entries removed.
```



In [ ]: