In [1]:

```python
import numpy as np
import pandas as pd
```

In [2]:

```python
dp=pd.read_table('cash_flow_statement_filtered_data.txt', sep=',')
dp
```

Out[2]:

	Symbol	Period Ending	Next Period Start Opening Price (Period Ending + 1 Day)	Next Period End Closing Price (Period Ending + 1 year)	Price Percentage Change	Net Income	Depreciation	Net Income Adjustments	Accounts Receivable	Changes in Inventories	...	Capital Expenditures	Inv
0	AROW	12/31/2019	36.6893	29.9100	-18.477594	37475.0	5503.0	2775.0	10030.0	0.0	...	-7785.0	
1	AROW	12/31/2018	29.7672	36.6990	23.286705	36279.0	4751.0	2557.0	-676.0	0.0	...	-5103.0	
2	AROW	12/31/2017	31.2978	30.1819	-3.565426	29326.0	5398.0	1575.0	982.0	0.0	...	-2602.0	
3	AROW	12/31/2016	36.3835	31.1148	-14.481015	26534.0	5940.0	2107.0	1077.0	0.0	...	-1441.0	
4	KMB	12/31/2019	137.4000	134.8300	-1.870451	2157000.0	917000.0	-90000.0	0.0	0.0	...	-1209000.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
10494	BEST	12/31/2016	11.4800	9.2300	-19.599303	-196198.0	35443.0	5131.0	145600.0	-23809.0	...	-106002.0	
10495	PRVB	12/31/2019	15.0500	16.9400	12.558140	-43285.0	-16.0	2826.0	1969.0	58.0	...	0.0	
10496	PRVB	12/31/2018	1.7700	14.9000	741.807910	-26478.0	0.0	5639.0	593.0	0.0	...	0.0	
10497	PRVB	12/31/2017	8.0000	1.7700	-77.875000	-9133.0	0.0	3732.0	988.0	0.0	...	0.0	
10498	PRVB	12/31/2016	8.0000	4.8100	-39.875000	-165.0	0.0	0.0	290.0	0.0	...	0.0	

10499 rows × 23 columns

In [3]:

```python
#Normalization
```

In [4]:

```python
array=np.array(dp.values)
x=np.delete(array,4,1)
y=x[:,3]
x=np.delete(x,3,1)
```

In [5]:

```python
print(x,y)
x.shape
y.shape
```

```
[['AROW' '12/31/2019' 36.6893 ... 134269.0 0.0 -14018.0]
 ['AROW' '12/31/2018' 29.7672 ... 197195.0 0.0 11401.0]
 ['AROW' '12/31/2017' 31.2978 ... 126222.0 0.0 15483.0]
 ...
 ['PRVB' '12/31/2018' 1.77 ... 59347.0 0.0 36705.0]
 ['PRVB' '12/31/2017' 8.0 ... 26716.0 0.0 21834.0]
 ['PRVB' '12/31/2016' 8.0 ... 0.0 0.0 0.0]] [29.91 36.699 30.1819 ... 14.9 1.77 4.81]
```

Out[5]:

```
(10499,)
```

In [6]:

```python
#Imputing
```

In [7]:

```python
from sklearn.impute import SimpleImputer
imputer= SimpleImputer(missing_values=np.nan,strategy= 'mean')
imputer.fit(x[:,2:])
x[:,2:] = imputer.transform(x[:,2:])
```

In [8]:

```python
#Encoding
```

In [9]:

```python
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
x[:,0]=le.fit_transform(x[:,0])
```

In [10]:

```python
print(x)
print(x)
```

```
[[110 '12/31/2019' 36.6893 ... 134269.0 0.0 -14018.0]
 [110 '12/31/2018' 29.7672 ... 197195.0 0.0 11401.0]
 [110 '12/31/2017' 31.2978 ... 126222.0 0.0 15483.0]
 ...
 [1989 '12/31/2018' 1.77 ... 59347.0 0.0 36705.0]
 [1989 '12/31/2017' 8.0 ... 26716.0 0.0 21834.0]
 [1989 '12/31/2016' 8.0 ... 0.0 0.0 0.0]]
```

In [11]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

In [12]:

```
#Standarizing
```

In [13]:

```
from sklearn.preprocessing import StandardScaler

Scaler=StandardScaler()
x_train[:,2:]=Scaler.fit_transform(x_train[:,2:])
x_test[:,2:]= Scaler.transform(x_test[:,2:])
```

In [14]:

```
print(x_train)
```

```
[[1435 '6/30/2017' -0.010927179360328066 ... 0.016999605122782133
  0.031836217094894724 -0.03228197367653003]
 [61 '12/31/2018' -0.010924156142402981 ... -0.006486991887826915
  0.0329752581476474 0.019499260491123778]
 [841 '12/31/2018' -0.01091533088049592 ... 0.07323475919120097
  0.031836217094894724 0.02765406362214818]
 ...
 [1448 '12/31/2017' -0.010911348198199401 ... -0.13621217237809113
  0.3273421505920509 0.049813603905770935]
 [1898 '9/30/2019' -0.010924875740681557 ... -0.036446100929511116
  0.031836217094894724 -0.018501010223878324]
 [2077 '12/31/2017' -0.010926170112427927 ... 0.013537672369570694
  0.031836217094894724 -0.020843641145194772]]
```

In [15]:

```
pd.DataFrame(x_train).to_csv("x_train.csv")
```

In [16]:

```
pd.DataFrame(x_train).to_csv("x_test.csv")
```

In [17]:

```
pd.DataFrame(x_train).to_csv("y_train.csv")
```

In [18]:

```
pd.DataFrame(x_train).to_csv("y_test.csv")
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js