2. Load Data

2.1 Log the price ¶

In [2]:
```
df = pd.read_csv('all_stocks_5yr.csv')
df.describe()
```

Out[2]:

	open	high	low	close	volume
count	619029.000000	619032.000000	619032.000000	619040.000000	6.190400e+05
mean	83.023334	83.778311	82.256096	83.043763	4.321823e+06
std	97.378769	98.207519	96.507421	97.389748	8.693610e+06
min	1.620000	1.690000	1.500000	1.590000	0.000000e+00
25%	40.220000	40.620000	39.830000	40.245000	1.070320e+06
50%	62.590000	63.150000	62.020000	62.620000	2.082094e+06
75%	94.370000	95.180000	93.540000	94.410000	4.284509e+06
max	2044.000000	2067.990000	2035.110000	2049.000000	6.182376e+08

In [4]:
```
df1 = df.ffill().copy()
```

In [4]:
```
df1['open'] = np.log(df1['open'])
df1['high'] = 10* (np.log(df1['high']) - np.log(df1['open']))
df1['low'] = 10* (np.log(df1['low']) - np.log(df1['open']))
df1['close'] = np.log(df1['close'])
df1['volume'] = np.log(df1['volume'])
df1.loc[:, 'date'] = pd.to_datetime(df1.loc[:,'date'], format="%Y/%m/%d")
df1['year'] = pd.DatetimeIndex(df1['date']).year
df1.head()
```

Out[4]:

	date	open	high	low	close	volume	Name	year
0	2013-02-08	2.712706	17.180717	16.851275	2.691243	15.944635	AAL	2013
1	2013-02-11	2.700690	17.152094	16.639512	2.671386	15.999537	AAL	2013
2	2013-02-12	2.670694	16.924995	16.638363	2.658159	15.910579	AAL	2013
3	2013-02-13	2.660260	17.256185	16.783332	2.685123	16.143715	AAL	2013
4	2013-02-14	2.704042	17.106322	15.824342	2.638343	17.277486	AAL	2013

In [5]:
```
df2 = df1[df1['year'] == 2018]
df2.to_csv (r'test.csv', index = None, header=True)
df3 = df1[df1['year'] != 2018]
df3.to_csv (r'train.csv', index = None, header=True)
```

In [6]:
```
name = list(np.unique(df3[df3['year'] == 2017].Name))
```

Generate the feature matrix

In [7]:
```
def create_feature(df3):

    grouped = df3.groupby('date')
    i = 0
    price = nd.zeros((2525, len(grouped)))

    for date, group in grouped:
        rec = grouped.get_group(date).reset_index(drop = True)
        price_date = np.zeros((505,5))
        for a in range(len(name)):
            if len(rec[rec['Name'] == name[a]]) > 0:
                price_date[a] = rec[rec['Name'] == name[a]].iloc[:,1:6].values
            '''
            else:
                print('here == 0: ', name[a])
                unnamed.append(name[a])
            '''
        price[:,i] = price_date.flatten()
        i+=1
    return price

test_feature = create_feature(df2) #test
print('test feature matrix: \n',test_feature)

train_feature = create_feature(df3) #train
print('train feature matrix: \n',train_feature)
```

```
test feature matrix:

[[ 4.210942   4.213904   4.2419024 ...  4.260706   4.2040954  4.222298 ]
 [27.802025  28.027933  28.009087  ... 28.199827  27.955805  27.949581 ]
 [27.72068   27.752186  27.859013  ... 27.727163  27.555634  27.777302 ]
 ...
 [28.142883  28.150318  28.246767  ... 28.252508  28.229082  28.307703 ]
 [ 4.2734666  4.278054   4.2840004 ...  4.301765   4.2941513  4.302171 ]
 [14.574242  14.660737  14.745314  ... 14.901385  15.409698  15.327316 ]]
<NDArray 2525x26 @cpu(0)>
train feature matrix:

[[ 3.8082168  3.8104331  3.8024313 ...  4.207822   4.2112384  4.2121277]
 [24.772491  24.729116  24.6991    ... 27.745897  27.740746  27.753443 ]
 [24.695015  24.566221  24.598486  ... 27.699837  27.672344  27.656794 ]
 ...
 [22.293644  22.405806  22.458382  ... 28.25052   28.238503  28.2227   ]
 [ 3.4980216  3.5043554  3.5186841 ...  4.2828965  4.2820683  4.2772217]
 [14.770726  14.211676  14.308546  ... 13.963733  13.473722  14.34856  ]]
<NDArray 2525x1233 @cpu(0)>
```

Fill the missing security with the very first day's price

In [9]:
```
train_feature2 = train_feature
for i in range(2525):
    if train_feature2[i].min() == 0:
        max_index = np.nonzero(train_feature2[i])[0].max()
        min_index = np.nonzero(train_feature2[i])[0].min()
        if max_index == 1232:
            #print(train_feature2[i])
            train_feature2[i,:min_index+1] = train_feature2[i,min_index+1]
            #print(train_feature2[i])
        elif min_index == 0:
            train_feature2[i,:max_index+1:] = train_feature2[i,max_index]
```

In [10]:
```
train_feature2 = train_feature2.T
feature = train_feature2.asnumpy()
np.savetxt("feature.csv", feature, delimiter=",")
```

Generate the label and do the same preprocessing process

In [22]:
```
def create_label(df3):

    grouped = df3.groupby('date')
    i = 0
    price = nd.zeros((505, len(grouped)))

    for date, group in grouped:
        rec = grouped.get_group(date).reset_index(drop = True)
        price_date = nd.zeros((505,))
        for a in range(len(name)):
            if len(rec[rec['Name'] == name[a]]) > 0:
                price_date[a] = rec[rec['Name'] == name[a]].iloc[:,1].values
        price[:,i] = price_date
        i+=1
    return price
```

In [23]:
```
train_label = create_label(df3)
print('train label matrix: \n',train_label)
```

```
train label matrix:

[[3.8082168 3.8104331 3.8024313 ... 4.207822  4.2112384 4.2121277]
 [2.712706  2.7006898 2.6706944 ... 3.9665112 3.9598603 3.9592881]
 [4.361058  4.3650074 4.3616962 ... 4.622224  4.6037693 4.603669 ]
 ...
 [4.3177547 4.3274384 4.3261175 ... 4.787492  4.7889905 4.8019695]
 [3.179303  3.1838703 3.189653  ... 3.9275026 3.930256  3.937301 ]
 [3.4753768 3.4983242 3.508556  ... 4.284827  4.2834487 4.284276 ]]
<NDArray 505x1233 @cpu(0)>
```

In [20]:
```
test_label = create_label(df2)
print('test label matrix: \n',test_label)
```

```
test label matrix:

[[4.210942  4.213904  4.2419024 ... 4.260706  4.2040954 4.222298 ]
 [3.9575698 3.9676468 3.960432  ... 3.9510515 3.8983297 3.9300594]
 [4.61413   4.6673937 4.679814  ... 4.7278304 4.6847205 4.719302 ]
 ...
 [4.7957907 4.825991  4.8306313 ... 4.8266325 4.7957907 4.801148 ]
 [3.9545076 3.9257286 3.9322176 ... 3.980429  3.9201896 3.955657 ]
 [4.284138  4.2734666 4.287029  ... 4.339119  4.2868915 4.286341 ]]
<NDArray 505x26 @cpu(0)>
```

In [25]:
```
train_label2 = train_label
for i in range(505):
    if train_label2[i].min() == 0:
        max_index = np.nonzero(train_label2[i])[0].max()
        min_index = np.nonzero(train_label2[i])[0].min()
        if max_index == 1232:
            #print(train_feature2[i])
            train_label2[i,:min_index+1] = train_label2[i,min_index+1]
            #print(train_feature2[i])
        elif min_index == 0:
            train_label2[i,:max_index+1:] = train_label2[i,max_index]
```

In [26]:
```
test_label = test_label.T

train_label2 = train_label2.T
label = train_label2.asnumpy()
np.savetxt("label.csv", label, delimiter=",")
```

To save time, we could load file from CSV

In [38]:
```
feature_df = pd.read_csv("feature.csv", header = None)
featureMatrix = nd.array(feature_df.values)
```

In [39]:
```
label_df = pd.read_csv("label.csv", header = None)
labelMatrix = nd.array(label_df.values)

ctx = d2l.try_gpu()
featureMatrix = featureMatrix[226:1233,:].as_in_context(ctx)
labelMatrix = labelMatrix[226:,:].as_in_context(ctx)
print(featureMatrix.shape, labelMatrix.shape)
```

```
(1007, 2525) (1007, 505)
```

In [59]:
```
train_iter = gdata.DataLoader(gdata.ArrayDataset(featureMatrix, labelMatrix), batch_size=
```

In [ ]:
```

```