Final Project Part 2

Nathan Tansey

Table of Contents

Introduction

In this project I seek to estimate the associated effect on Total Value Added from eight different features: total employment, total payroll, total value of shipments, total cost of materials, total capital expenditure, total real capital stock, end of year inventories, and cost of electricity & fuels. The benefit of this project is to better inform businesses and their decision making process by allowing them to have an estimated associated effect from the eight labels above on the value added. The data used for this project contains 459 industries with annual observations for 54 years, from 1958 to 2011.

Data Wrangling and Cleaning

The data used here comes from the National Bureau of Economic Research and their collaboration with the US Census Bureau's Center for Economic Studies. Link: https://www.nber.org/research/data/nber-ces-manufacturing-industry-database

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(rc = {'axes.titlesize': 24,
              'axes.labelsize': 20,
              'xtick.labelsize': 12,
              'ytick.labelsize': 12,
              'figure.figsize': (12, 6)})
from scipy.stats import binned_statistic
```

In [2]:
```
sic = pd.read_excel('sic5811.xls')
```

In [3]:
```
sic = sic.set_index(['sic', 'year'])
# moving identifying columns of industry number (sic) and year to the index
```

In [4]:
```
sic = sic[['vadd', 'emp', 'invest', 'pay', 'matcost', 'vship', 'cap', 'invent', 'ene
# selecting the continuous label vadd, and the eight features
```

In [5]:
```
sic.info()
# Checking if all variables are correct data type for continuous variables, looks go
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 24786 entries, (2011, 1958) to (3999, 2011)
Data columns (total 9 columns):
vadd       24676 non-null float64
emp        24676 non-null float64
invest     24676 non-null float64
pay        24676 non-null float64
matcost    24676 non-null float64
vship      24676 non-null float64
cap        24676 non-null float64
invent     24676 non-null float64
energy     24676 non-null float64
dtypes: float64(9)
memory usage: 1.8 MB
```

In [6]:
```
print(
    sic.isnull().any(),
    sic.shape
) # checking for null values
```

```
vadd       True
emp        True
invest     True
pay        True
matcost    True
vship      True
cap        True
invent     True
energy     True
dtype: bool (24786, 9)
```

In [7]:
```
sic = sic.dropna() #dropping na's
```

In [8]:
```
print(
    sic.isnull().any(),
    sic.shape
) # all null values gone, got rid of 110 rows with na's
```

```
vadd       False
emp        False
invest     False
pay        False
matcost    False
vship      False
cap        False
invent     False
energy     False
dtype: bool (24676, 9)
```

In [9]:
```
sic.head()
```

Out[9]:

		vadd	emp	invest	pay	matcost	vship	cap	invent	energy
sic	**year**									
2011	**1958**	1748.6	200.9	65.9	1067.8	10230.1	11950.7	3575.5	408.1	47.9
	1959	1833.2	197.2	67.4	1101.0	9939.1	11788.4	3717.8	370.1	49.4
	1960	1910.7	194.2	77.2	1138.6	9890.8	11806.2	3883.3	381.6	50.9

sic	year	vadd	emp	invest	pay	matcost	vship	cap	invent	energy
	1961	1889.2	189.3	75.4	1143.2	10047.3	11916.8	4023.8	395.3	52.4
	1962	1986.1	185.6	90.8	1161.1	10508.8	12468.3	4211.6	411.1	53.9

In [10]:
```
sic.shape
```

Out[10]:
```
(24676, 9)
```

In [11]:
```
sic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 24676 entries, (2011, 1958) to (3999, 2011)
Data columns (total 9 columns):
vadd       24676 non-null float64
emp        24676 non-null float64
invest     24676 non-null float64
pay        24676 non-null float64
matcost    24676 non-null float64
vship      24676 non-null float64
cap        24676 non-null float64
invent     24676 non-null float64
energy     24676 non-null float64
dtypes: float64(9)
memory usage: 1.8 MB
```

In [12]:
```
sic.describe()
```

Out[12]:

	vadd	emp	invest	pay	matcost	vship	
count	24676.000000	24676.000000	24676.000000	24676.000000	24676.000000	24676.000000	246
mean	2382.981350	36.343804	164.578988	784.571012	2852.133575	5220.267661	24
std	5506.653754	51.156499	518.200755	1479.515880	12126.707451	16307.052642	58
min	10.200000	0.100000	0.100000	5.000000	5.700000	19.100000	
25%	315.150000	10.300000	14.900000	135.300000	300.675000	644.375000	4
50%	852.000000	20.200000	45.900000	332.900000	873.900000	1777.250000	9
75%	2218.275000	40.800000	135.400000	800.900000	2351.075000	4644.150000	22
max	111665.700000	565.400000	17608.100000	22245.300000	688029.100000	793716.900000	1054

In [13]:
```
#everything looks good, converting to pkl
sic.to_pickle('sic_fp.pkl')
```

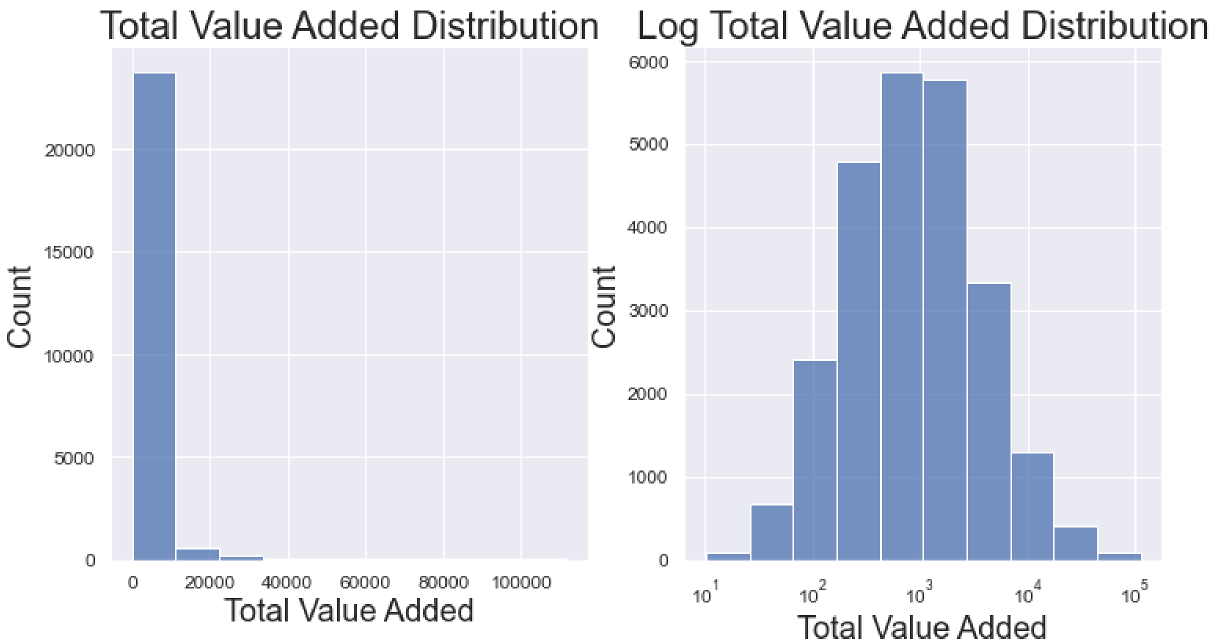
Label Figure

TOP

In [14]:
```
# label is continuous with structure of panel data, creating histogram of label and
```

```
plt.figure()
plt.subplot(1,2,1)
plt.title('Total Value Added Distribution')
plt.xlabel('Total Value Added')
sns.histplot(x = 'vadd', data = sic, bins = 10)

plt.subplot(1,2,2)
plt.title('Log Total Value Added Distribution')
plt.xlabel('Total Value Added')
plt.semilogx()
sns.histplot(x = 'vadd', data = sic, bins = 10)
```

Out[14]: <AxesSubplot:title={'center':'Log Total Value Added Distribution'}, xlabel='Total Value Added', ylabel='Count'>



Response A log transformation does indeed seem appropriate for Total Value Added, the right graph with the transformation is more symmetric than the non-transformed histogram

In [15]:

```
sic['l_vadd'] = np.log(sic['vadd']) # creating new column with log transformation of
sic = sic.drop('vadd', 1) # dropping old label
```

In [16]:

```
sic = sic[['l_vadd', 'emp', 'invest', 'pay', 'matcost', 'vship', 'cap', 'invent', 'e
# put label as first column because of personal preference
```

Feature Transformations

TOP

All eight features could need log transformations

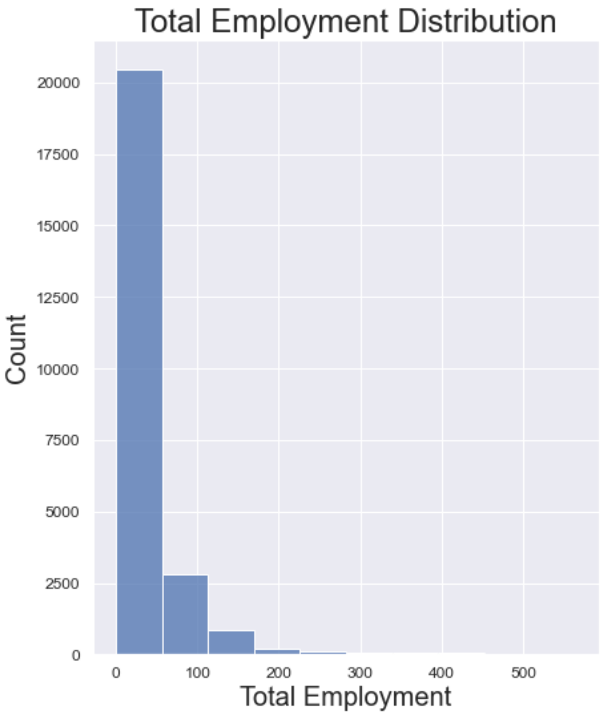
In [17]:

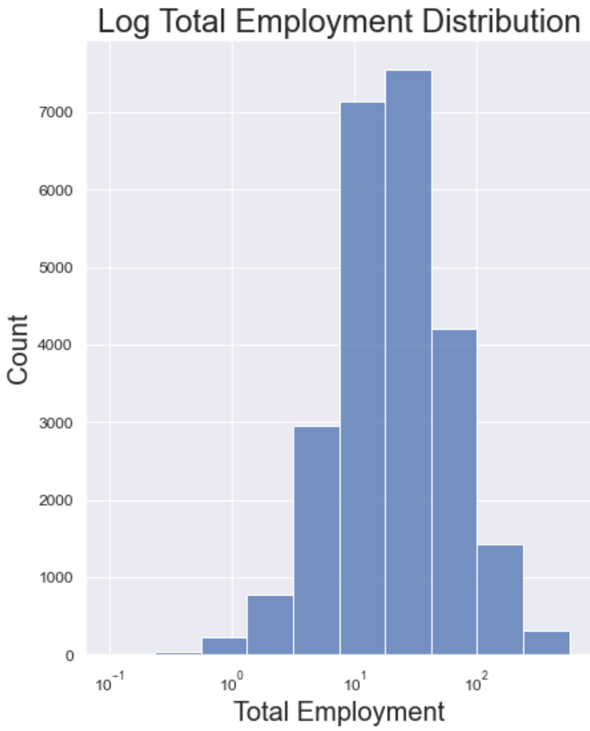
```
# Histograms of emp
plt.figure(figsize = (15,80))
plt.subplot(8,2,1)
plt.title('Total Employment Distribution')
plt.xlabel('Total Employment')
sns.histplot(x = 'emp', data = sic, bins = 10)
```

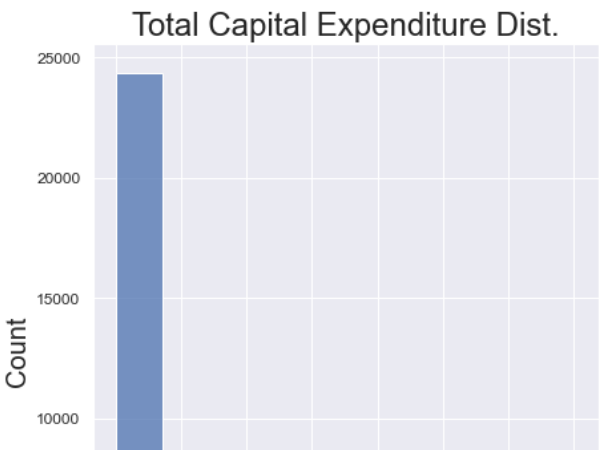
```
plt.subplot(8,2,2)
plt.title('Log Total Employment Distribution')
plt.xlabel('Total Employment')
plt.semilogx()
sns.histplot(x = 'emp', data = sic, bins = 10)

#Histograms of invest
plt.subplot(8,2,3)
plt.title('Total Capital Expenditure Dist.')
plt.xlabel('Total Capital Expenditure')
sns.histplot(x = 'invest', data = sic, bins = 10)

plt.subplot(8,2,4)
plt.title('Log Total Capital Expenditure Dist.')
plt.xlabel('Total Capital Expenditure')
plt.semilogx()
sns.histplot(x = 'invest', data = sic, bins = 10)


#Histograms of pay
plt.subplot(8,2,5)
plt.title('Total Payroll Distribution')
plt.xlabel('Total Payroll')
sns.histplot(x = 'pay', data = sic, bins = 10)

plt.subplot(8,2,6)
plt.title('Log Total Payroll')
plt.xlabel('Total Payroll')
plt.semilogx()
sns.histplot(x = 'pay', data = sic, bins = 10)

#Histograms of matcost
plt.subplot(8,2,7)
plt.title('Total Material Cost Dist.')
plt.xlabel('Total Material Costs')
sns.histplot(x = 'matcost', data = sic, bins = 10)

plt.subplot(8,2,8)
plt.title('Log Total Material Cost Dist.')
plt.xlabel('Total Material Costs')
plt.semilogx()
sns.histplot(x = 'matcost', data = sic, bins = 10)

#Histograms of vship
plt.subplot(8,2,9)
plt.title('Total Value of Shipments Dist.')
plt.xlabel('Total Value of Shipments')
sns.histplot(x = 'vship', data = sic, bins = 10)

plt.subplot(8,2,10)
plt.title('Total Value of Shipments Dist.')
plt.xlabel('Total Value of Shipments')
plt.semilogx()
sns.histplot(x = 'vship', data = sic, bins = 10)

#Histograms of cap
plt.subplot(8,2,11)
plt.title('Total Real Capital Stock Dist.')
plt.xlabel('Total Real Capital Stock')
sns.histplot(x = 'cap', data = sic, bins = 10)

plt.subplot(8,2,12)
plt.title('Log Total Real Capital Stock Dist.')
plt.xlabel('Total Real Capital Stock')
plt.semilogx()
```

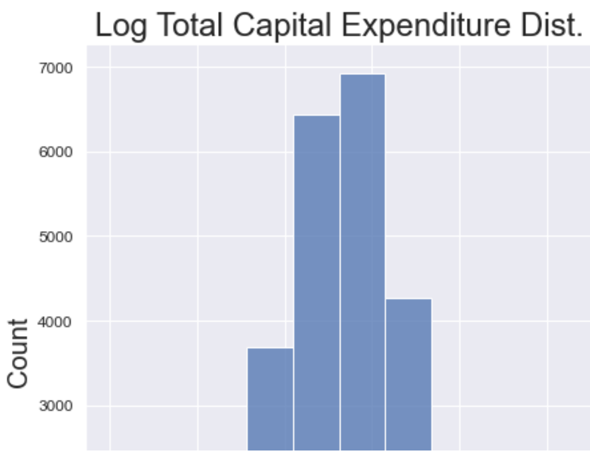
```
sns.histplot(x = 'cap', data = sic, bins = 10)

#Histograms of invent
plt.subplot(8,2,13)
plt.title('End of Year Inventory Dist.')
plt.xlabel('End of Year Inventories')
sns.histplot(x = 'invent', data = sic, bins = 10)

plt.subplot(8,2,14)
plt.title('Log End of Year Inventory Dist.')
plt.xlabel('End of Year Inventory')
plt.semilogx()
sns.histplot(x = 'invent', data = sic, bins = 10)

#Histograms of energy
plt.subplot(8,2,15)
plt.title('Electricity & Fuel Cost Dist.')
plt.xlabel('Cost of Electricity & Fuel')
sns.histplot(x = 'energy', data = sic, bins = 10)

plt.subplot(8,2,16)
plt.title('Log Electricity & Fuel Cost Dist. ')
plt.xlabel('Cost of Electricity & Fuel')
plt.semilogx()
sns.histplot(x = 'energy', data = sic, bins = 10)
```

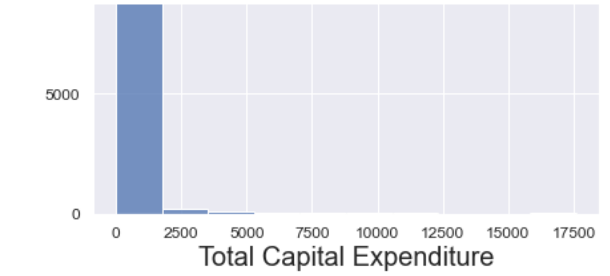
```
Out[17]: <AxesSubplot:title={'center':'Log Electricity & Fuel Cost Dist. '}, xlabel='Cost of Electricity & Fuel', ylabel='Count'>
```

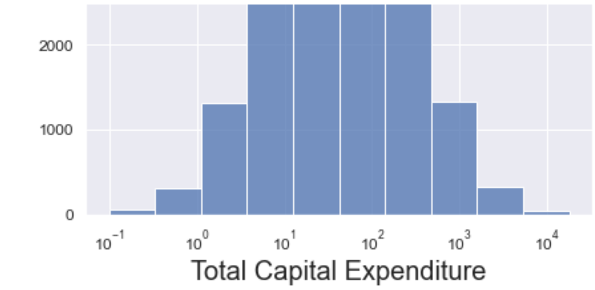




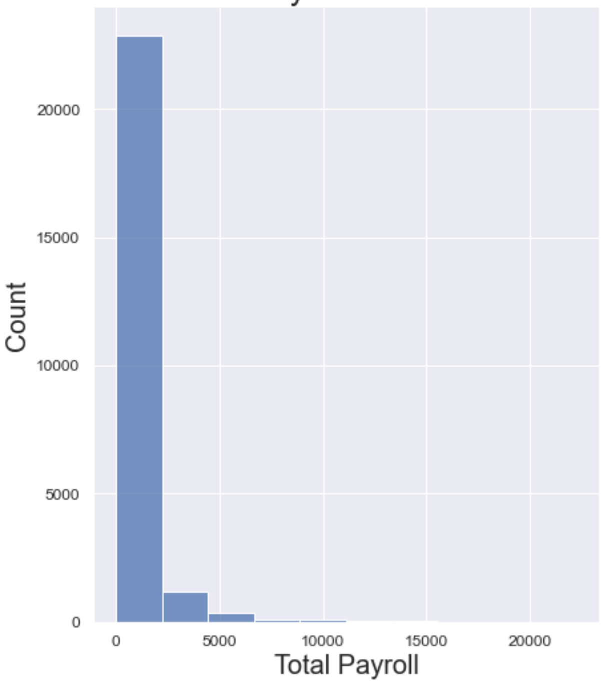




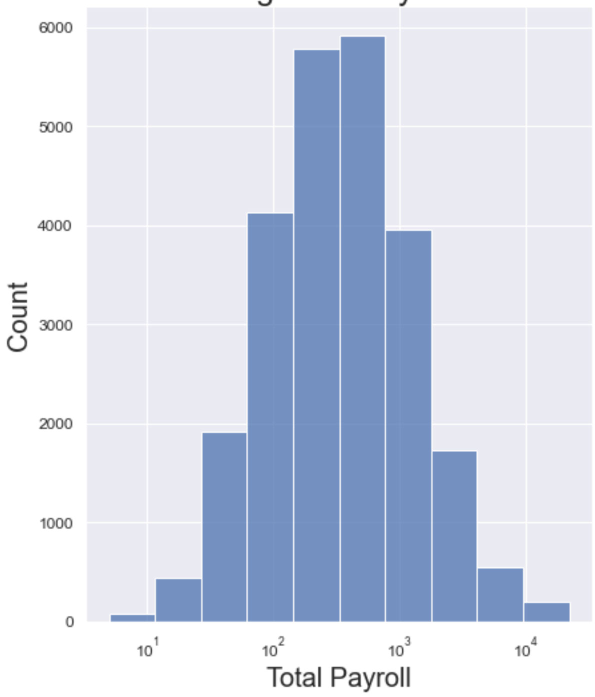




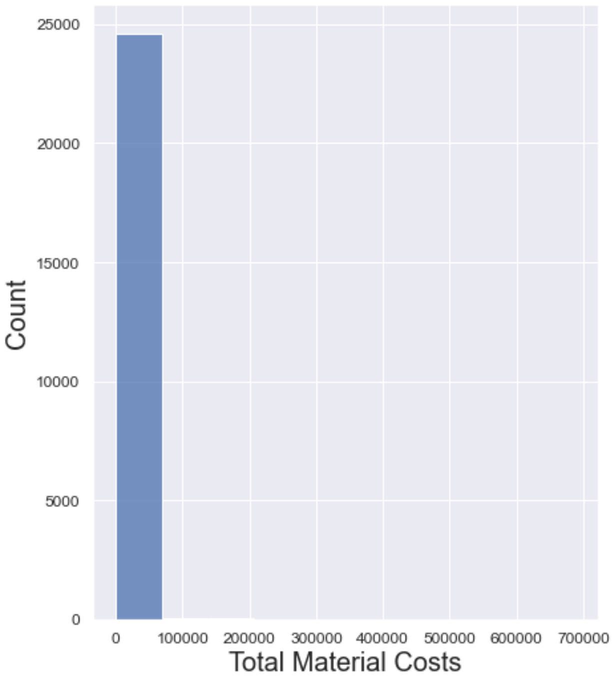
Total Payroll Distribution



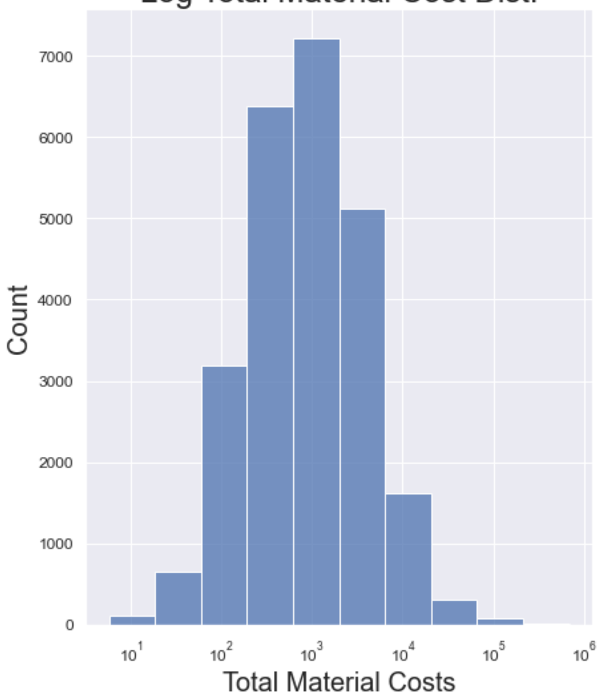
Log Total Payroll



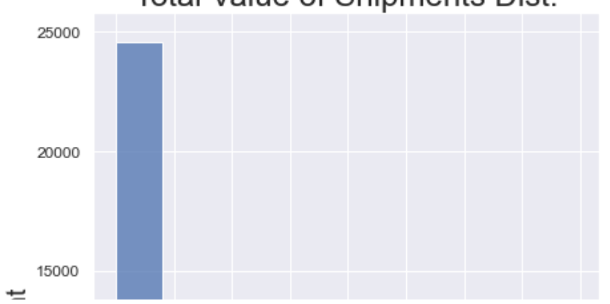
Total Material Cost Dist.



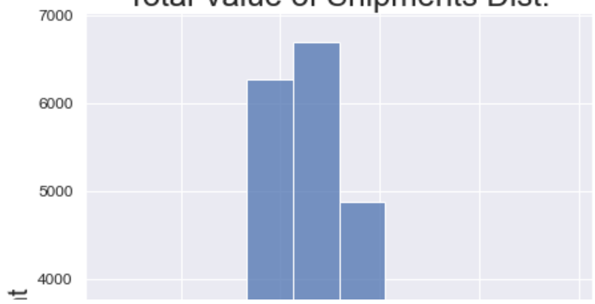
Log Total Material Cost Dist.

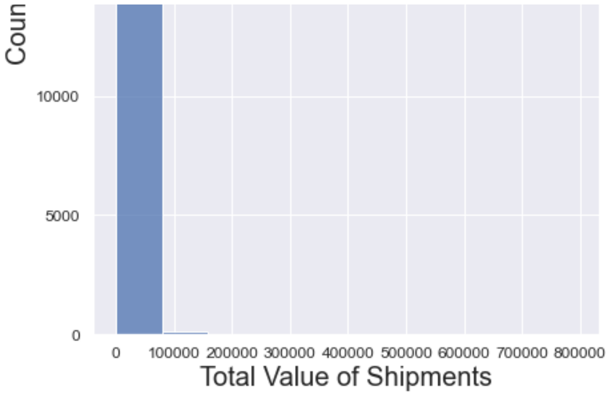


Total Value of Shipments Dist.

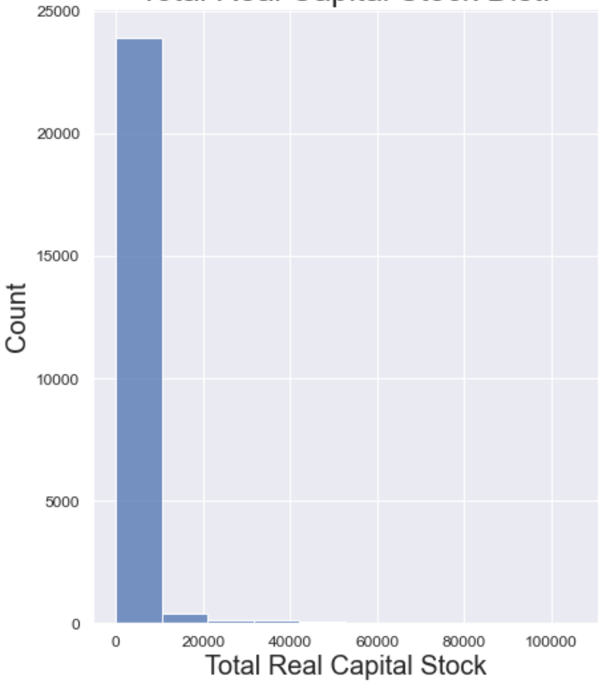


Total Value of Shipments Dist.

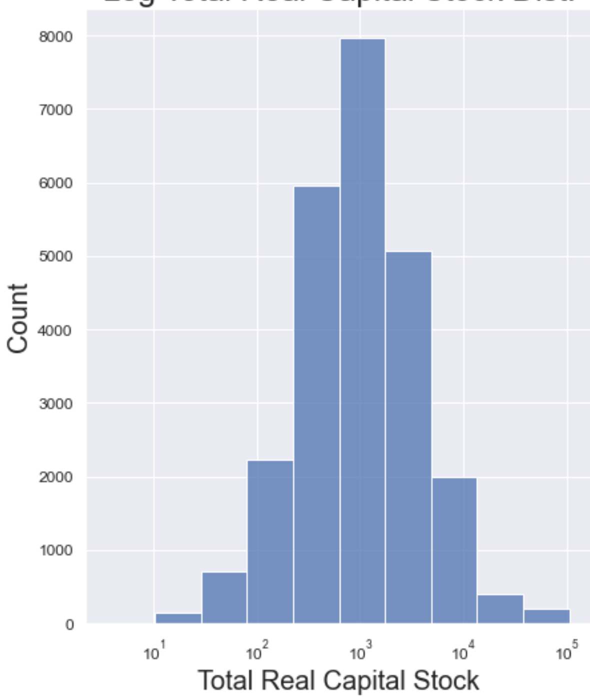




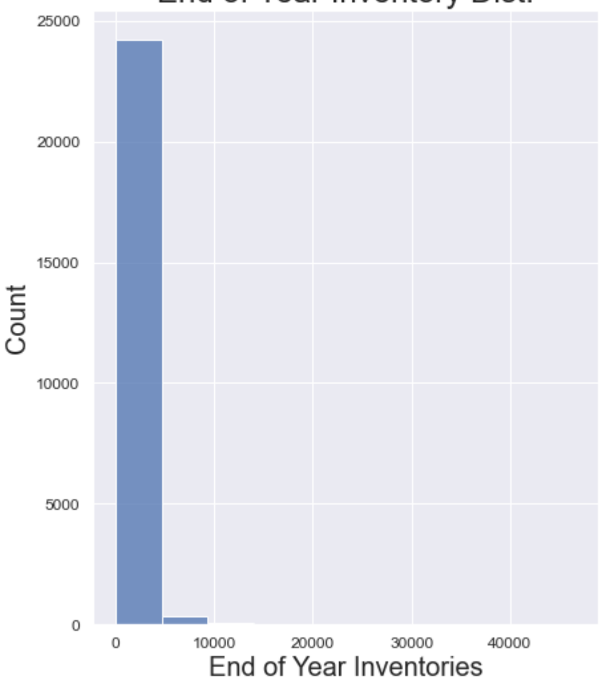
Total Real Capital Stock Dist.



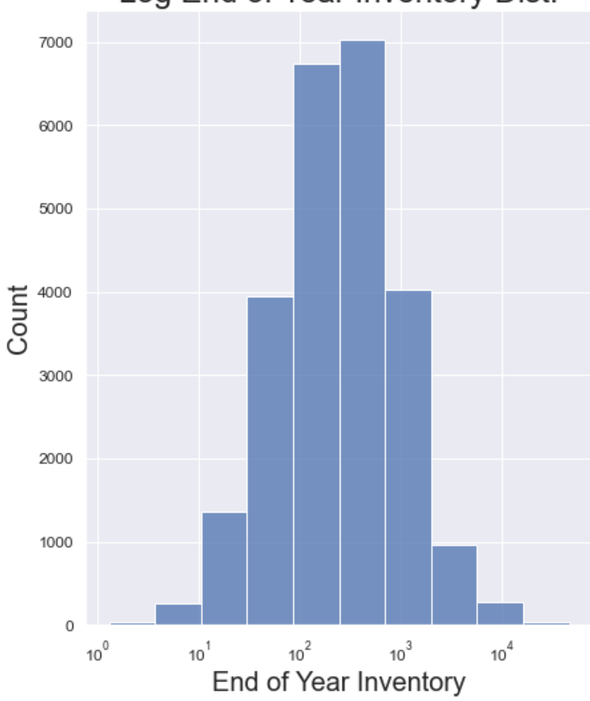
Log Total Real Capital Stock Dist.



End of Year Inventory Dist.

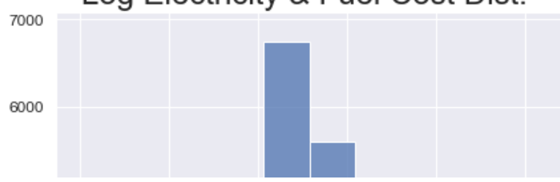


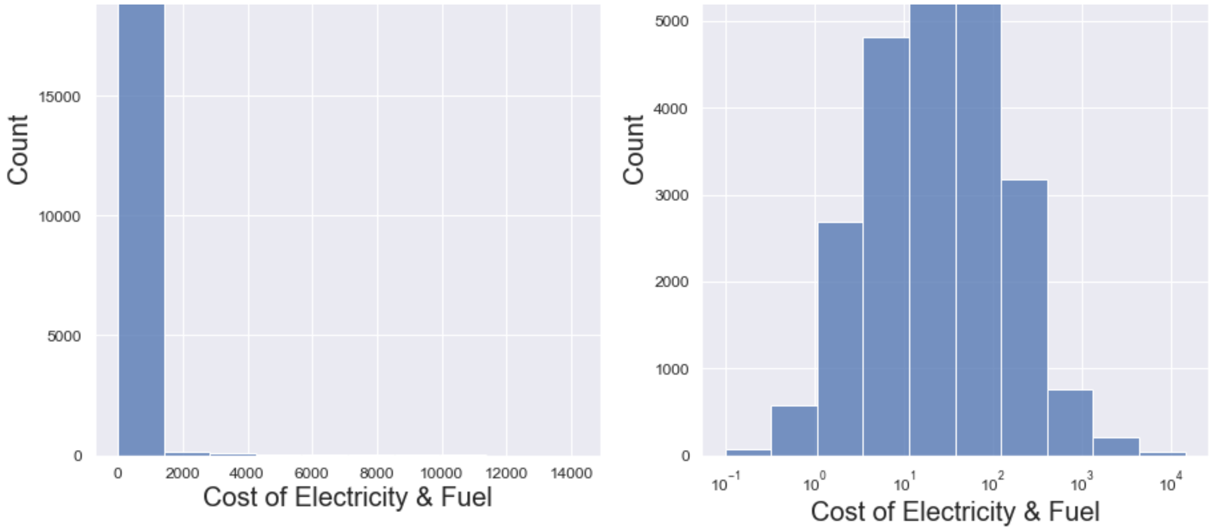
Log End of Year Inventory Dist.



Electricity & Fuel Cost Dist.

Log Electricity & Fuel Cost Dist.





All eight features need log transformations based on the histogram plots.

In [18]:

```
sic['l_emp'] = np.log(sic['emp']) # creating new column with log transformation
sic = sic.drop('emp', 1) # dropping old column

sic['l_invest'] = np.log(sic['invest']) # creating new column with log transformatio
sic = sic.drop('invest', 1) # dropping old column

sic['l_pay'] = np.log(sic['pay']) # creating new column with log transformation
sic = sic.drop('pay', 1) # dropping old column

sic['l_matcost'] = np.log(sic['matcost']) # creating new column with log transformat
sic = sic.drop('matcost', 1) # dropping old column

sic['l_vship'] = np.log(sic['vship']) # creating new column with log transformation
sic = sic.drop('vship', 1) # dropping old column

sic['l_cap'] = np.log(sic['cap']) # creating new column with log transformation
sic = sic.drop('cap', 1) # dropping old column

sic['l_invent'] = np.log(sic['invent']) # creating new column with log transformatio
sic = sic.drop('invent', 1) # dropping old column

sic['l_energy'] = np.log(sic['energy']) # creating new column with log transformatio
sic = sic.drop('energy', 1) # dropping old column
```

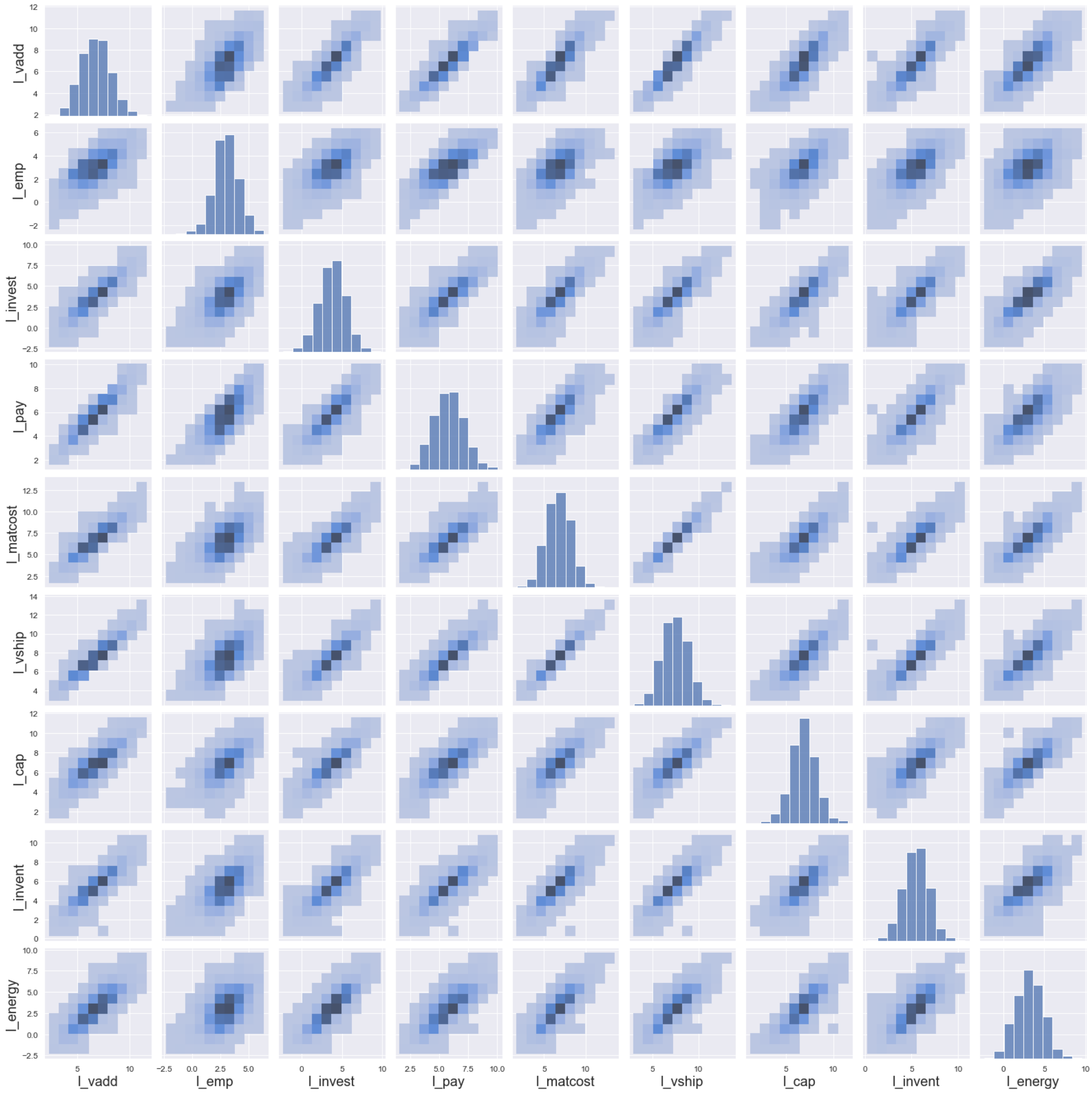
Feature vs Label Figures

TOP

Figure 1

In [19]:

```
sns.pairplot(data = sic, kind = 'hist', plot_kws = {'bins':10}, diag_kws = {'bins':
plt.show()
```



The relationships between the different features appear to be linear for all features shown. No evidence of any higher degree relationships that can be noticed from the pairplot graph.

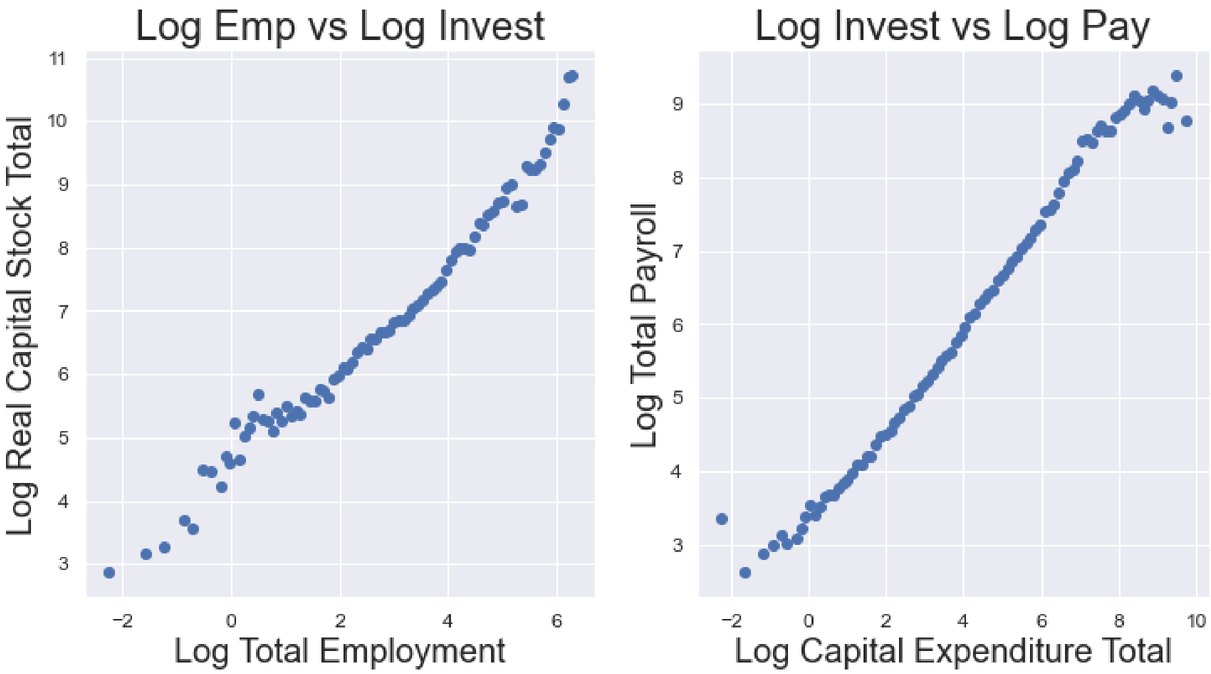
Figure 2

In [20]:

```
# Examining l_emp vs l_cap, as well as  l_invest vs l_pay


#creating l_emp vs l_cap
plt.figure()
plt.subplot(1,2,1)
n = 100
bin_mean, bin_edge, _ = binned_statistic(sic.l_emp, sic.l_cap, bins = n)
x = np.average([bin_edge[:-1], bin_edge[1:]], axis = 0)
plt.xlabel('Log Total Employment')
plt.ylabel('Log Real Capital Stock Total')
plt.title('Log Emp vs Log Invest')
plt.scatter(x, bin_mean)


# creating l_invest vs l_pay
plt.subplot(1,2,2)
bin_mean2, bin_edge2, _ = binned_statistic(sic.l_invest, sic.l_pay, bins = n)
y = np.average([bin_edge2[:-1], bin_edge2[1:]], axis = 0)
plt.xlabel('Log Capital Expenditure Total')
```

```
plt.ylabel('Log Total Payroll')
plt.title('Log Invest vs Log Pay')
plt.scatter(y, bin_mean2)
```

Out[20]: <matplotlib.collections.PathCollection at 0x275c13fcd30>



Both graphs appear to be fairly linear, confirming my initial ideas from the pairplot. Log Total Employment vs Log Real Capital Stock Total, the left graph, shows indication a possibly a very very slight quadratic curve, but a linear relationship still appears better. Log Capital Expenditure vs Log Total Payroll is extremely linear for the majority, but starts to bend at Log Capital Expenditure Total = 8.