Houston - House Price Index vs Energy Industry

Analysis

- This study shall focus on the correlation between fluctuations in oil and gas prices and the strength of the correlation with house prices in Houston.
- Houston is the oil and gas capital of the USA - and it would be interesting to investigate the impact of oil prices on the local economy with respect to the housing market.
- In addition to focussing on the city of Houston, other major cities in Texas, such as Dallas and Austin shall be used to confirm the strength of the impact emanating from the energy industry.
- A comparison of the labor market involved in the oil/gas sector by each of the cities is a good starting point on which to base the argument.

Note

- The FRED API shall be utilized for gathering the data.

In [1]:

```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import requests
import time
import requests
import json

# Import API key
import config
from config import api_key

# Output File (CSV)
output_data_file = "output_data/quarter_data.csv"
output_data_file_2 = "output_data/annual_data.csv"
```

In [2]:

```
# New Dependency! Use this to pretty print the JSON
# https://docs.python.org/3/library/pprint.html
from pprint import pprint
```

In [3]:

```
# The URL for the FRED API
base = 'https://api.stlouisfed.org/fred/series/observations?series_id='

# List of FRED series IDs and their description
q_dict = {'ATNHPIUS26420Q': 'House Price Index for Houston',
          'HOUS448UR':'Unemployment Rate in Houston',
          #Thousands of Persons (monthly),
           'SMU48264201000000001SA': 'Mining and Logging',
           'HOUS448MFG': 'Manufacturing',
           'HOUS448TRAD': 'Trade, Transportation, and Utilities',
           'HOUS448FIRE': 'Financial Activities',
           'HOUS448PBSV': 'Professional and Business Services',
           'HOUS448EDUH': 'Education and Health Services',
           'HOUS448LEIH': 'Leisure and Hospitality',
           'HOUS448GOVT': 'Government',
           'SMU48264201021100001SA':'Oil and Gas Extraction',
           'SMU48264202000000001SA':'Construction in Houston',
           #Dollars (monthly)
           'WTISPLC': 'Global price of WTI Crude',
           #Percent (monthly)
           'MORTGAGE30US': '30Yr Fixed Mortgage Rate',
           #CPI (monthly)
           'CUURA318SA0':'CPI for Houston'
          }

# Include start date, API key from config.py file and file type json
start_date = '1990-01-01'
s_dates = '&observation_start={}'.format(start_date)
end_date = '2017-01-01'
e_dates = '&observation_end={}'.format(end_date)

api_key = '&api_key={}'.format(config.api_key)

ftype = '&file_type=json'
frequency = 'q'
freq = '&frequency={}'.format(frequency)
#aggregation = 'eop'
#agg = '&aggregation_method={}'.format(aggregation)
```

In [4]:

```
df = pd.DataFrame()
for code, name in q_dict.items():
    url = '{}{}{}{}{}{}{}{}'.format(base, code, s_dates,e_dates, api_key, ftype,freq)
    r = requests.get(url).json()['observations']
    df[name] = [i['value'] for i in r]
df.index = pd.to_datetime([i['date'] for i in r])
df.head()
df.to_csv('output_data/general.csv')
len(df)
```

Out[4]:

```
109
```

In [5]:

```
# List of FRED series IDs and their description
a_dict = {#'HOUS448PCPI':'Per Capita Personal Income - Houston',#2001 (annual)
          #
          # Population and Unemployment is annual only
           'TXHARR1POP':'Population in Harris County',
           'TXFORT5POP':'Population in Fort Bend County',
           'TXMONT0POP':'Population in Montgomery County',
           'TXGALV7POP':'Population in Galveston County',
           'TXBRAZ0POP':'Population in Brazoria County',
           'TXLIBE1POP':'Population in Liberty County',
           'TXWALL3POP':'Population in Waller County',
           'TXCHAM1POP':'Population in Chambers County',
           'TXAUST5POP':'Population in Austin County',
           'HOUS448LFN':'Labor Force'
           }

# Include start date, API key from config.py file and file type json
start_date = '1990-01-01'
s_dates = '&observation_start={}'.format(start_date)
end_date = '2017-01-01'
e_dates = '&observation_end={}'.format(end_date)
api_key = '&api_key={}'.format(config.api_key)
ftype = '&file_type=json'
frequency = 'a'
freq = '&frequency={}'.format(frequency)
#aggregation = 'eop'
#agg = '&aggregation_method={}'.format(aggregation)
```

In [6]:

```
dff = pd.DataFrame()
for code, name in a_dict.items():
    url = '{}{}{}{}{}{}{}{}'.format(base, code, s_dates,e_dates, api_key, ftype,freq)
    r = requests.get(url).json()['observations']
    dff[name] = [i['value'] for i in r]
dff.index = pd.to_datetime([i['date'] for i in r])
dff.to_csv('output_data/population.csv')
dff.head()
len(dff)
#dff.index.values
```

Out[6]:

```
28
```

Plotting the Data

- Use proper labeling of the plots using plot titles (including date of analysis) and axes labels.
- Save the plotted figures as .pngs.

House Price Index vs. Oil Price

In [7]:

```
x1_axis = df.index.values
x2_axis =dff.index.values
```

In [8]:

```
df.dtypes
cols = df.columns[df.dtypes.eq(object)]
cols
for c in cols:
    df[c] = pd.to_numeric(df[c], errors='coerce')
df.dtypes
```

Out[8]:

```
House Price Index for Houston           float64
Unemployment Rate in Houston            float64
Mining and Logging                      float64
Manufacturing                           float64
Trade, Transportation, and Utilities    float64
Financial Activities                    float64
Professional and Business Services      float64
Education and Health Services           float64
Leisure and Hospitality                 float64
Government                              float64
Oil and Gas Extraction                  float64
Construction in Houston                 float64
Global price of WTI Crude               float64
30Yr Fixed Mortgage Rate                float64
CPI for Houston                         float64
dtype: object
```

In [9]:

```
dff.dtypes
cols = dff.columns[dff.dtypes.eq(object)]
cols
for d in cols:
    dff[d] = pd.to_numeric(dff[d], errors='coerce')
dff.dtypes
```

Out[9]:

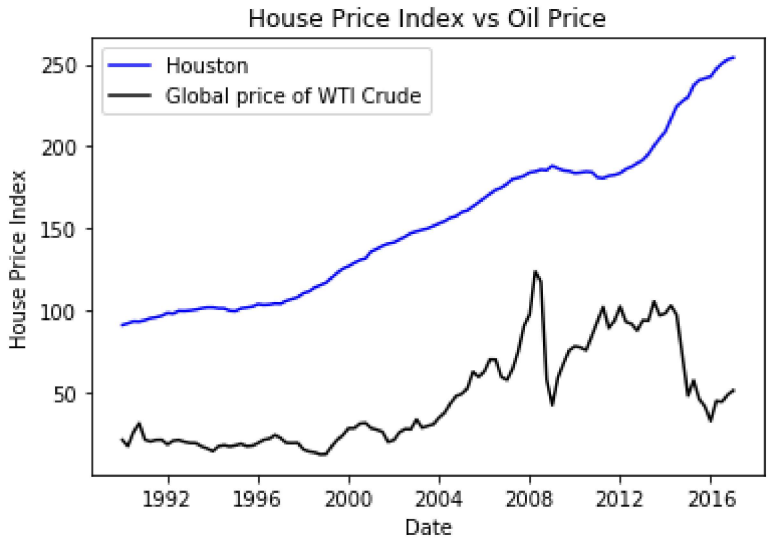
```
Population in Harris County        float64
Population in Fort Bend County     float64
Population in Montgomery County    float64
Population in Galveston County     float64
Population in Brazoria County      float64
Population in Liberty County       float64
Population in Waller County        float64
Population in Chambers County      float64
Population in Austin County        float64
Labor Force                          int64
dtype: object
```

In [10]:

```
df.to_csv(output_data_file)
dff.to_csv(output_data_file_2)
```

In [11]:

```
x1_axis = df.index.values

#plt.plot(x_axis, y_axis)
plt.plot(x1_axis, df["House Price Index for Houston"],color="blue", label="Houston" )
plt.plot(x1_axis, df["Global price of WTI Crude"],color='black')

plt.legend(loc="best")

plt.title(f"House Price Index vs Oil Price")
plt.ylabel("House Price Index")
plt.xlabel("Date")

#plt.savefig("Latitude_v_MaxTemp.png")
plt.savefig("output_data/House price Index vs WTI Oil Price.png")
plt.show()
```



In [12]:

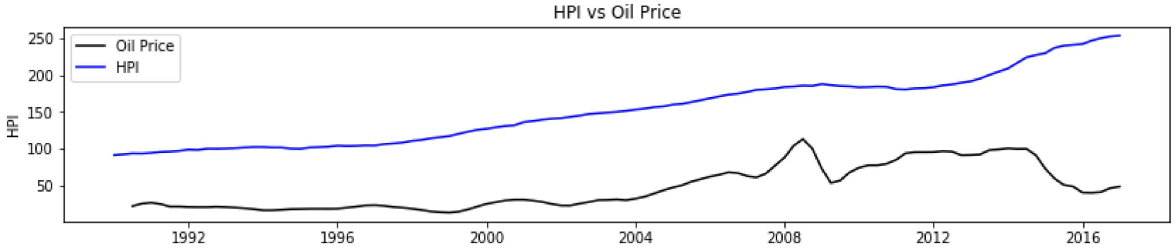
```
df['MA'] = df.rolling(window=3)["Global price of WTI Crude"].mean()
df.tail()
```

Out[12]:

	House Price Index for Houston	Unemployment Rate in Houston	Mining and Logging	Manufacturing	Trade, Transportation, and Utilities	Financial Activities	Professiona an Busines Service
2016-01-01	242.53	4.9	87.9	229.8	610.6	154.5	474.(
2016-04-01	247.21	5.2	82.5	224.0	609.6	154.9	471.5
2016-07-01	250.47	5.5	78.2	219.9	610.6	156.3	472.4
2016-10-01	252.91	5.5	76.2	217.7	612.9	157.1	471.5
2017-01-01	253.99	5.5	76.6	218.0	614.4	158.1	475.(

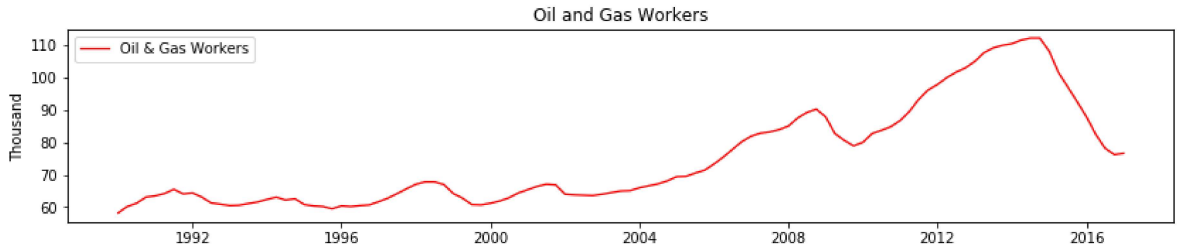
In [13]:

```
fig, ax = plt.subplots(figsize=(14,2.5), linewidth=5)
plt.plot(x1_axis, df["MA"],color='black',label = "Oil Price")
plt.plot(x1_axis, df["House Price Index for Houston"],color="blue", label="HPI" )


plt.legend(loc="best")

plt.title(f"HPI vs Oil Price")
plt.ylabel("HPI")
#plt.xlabel("Date")

plt.savefig("output_data/House price Index vs MA WTA Oil Price.png")
plt.show()
```



In [14]:

```
width = 0.000001        # the width of the bars
chart1 = df[["House Price Index for Houston","Global price of WTI Crude","Mining and Loggir
years = chart1.index.values
oil_price = chart1["Global price of WTI Crude"]
HPI = chart1["House Price Index for Houston"]
Mining_and_Logging = chart1["Mining and Logging"]
#fourth_year = (years % 4 == 0)
#years_4 = years[fourth_year]


#fig1, ax = plt.subplots(figsize=(14,4), linewidth=5, edgecolor='.5')
fig1, ax = plt.subplots(figsize=(14,2.5), linewidth=5)
#ax.bar(years, Mining_and_Logging,275 ,facecolor='0', alpha=.3, label='Oil Price')
ax.plot(years, Mining_and_Logging, linestyle='-', linewidth=1.3, color='red', label='Oil &
#ax.plot(years, oil_price, linestyle='-', linewidth=3, color='.2', label='Oil Price')
#ax.plot(years, HPI, linestyle='--', linewidth=3, color='.2', label='HPI')


ax.set_title('Oil and Gas Workers')
ax.set_ylabel('Thousand')
ax.legend()

#for x, y, v in zip(years, oil_price, oil_price):
#    ax.text(x, y + .5, str(v), ha='center')
plt.savefig("output_data/Bar_Chart_House price Index vs Oil Price.png")
plt.show()
```

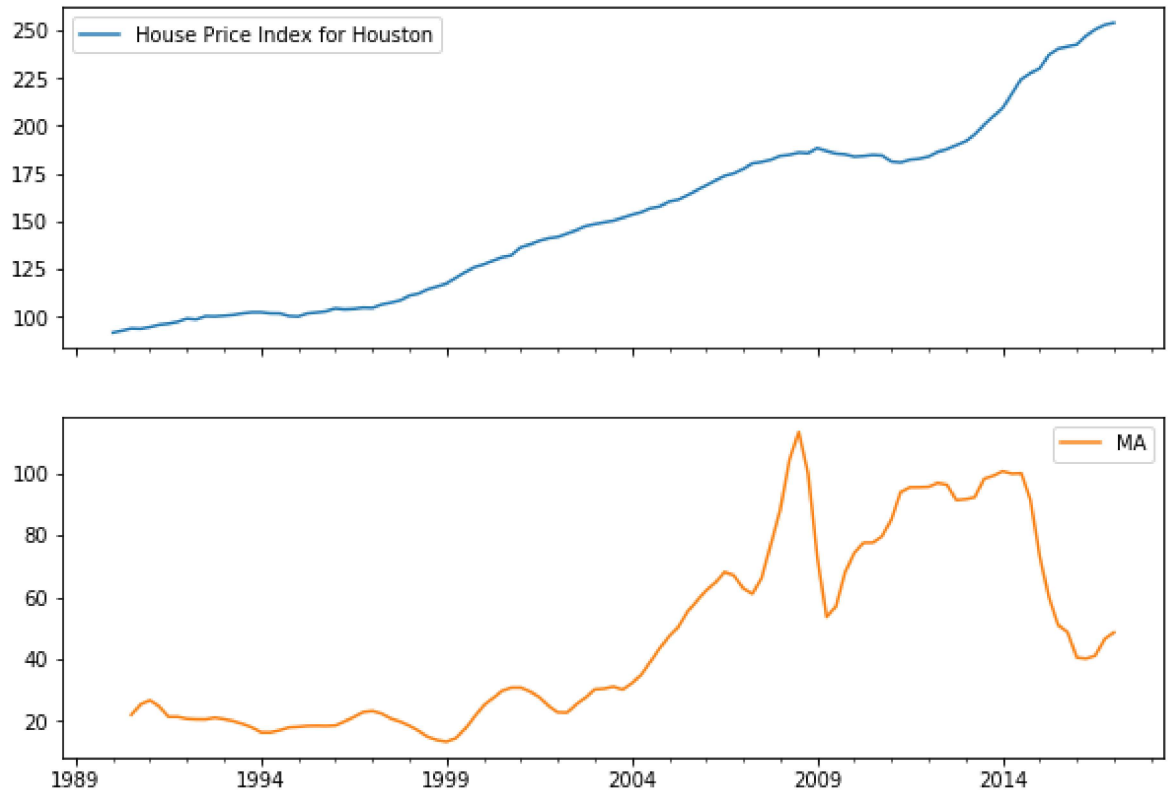


In [15]:

```
plot_cols = df[["House Price Index for Houston","MA"]]
plot_cols
fig, axes = plt.subplots(2,1, figsize=(10,7), sharex=True)
plot_cols.plot(subplots=True, ax=axes)
#for ax in axes:
#    ax.axvspan(x1_axis, df["House Price Index for Houston"], color=sns.xkcd_rgb['grey'], a
#    ax.axvspan(x1_axis, df["MA"],  color=sns.xkcd_rgb['grey'], alpha=0.5)
```

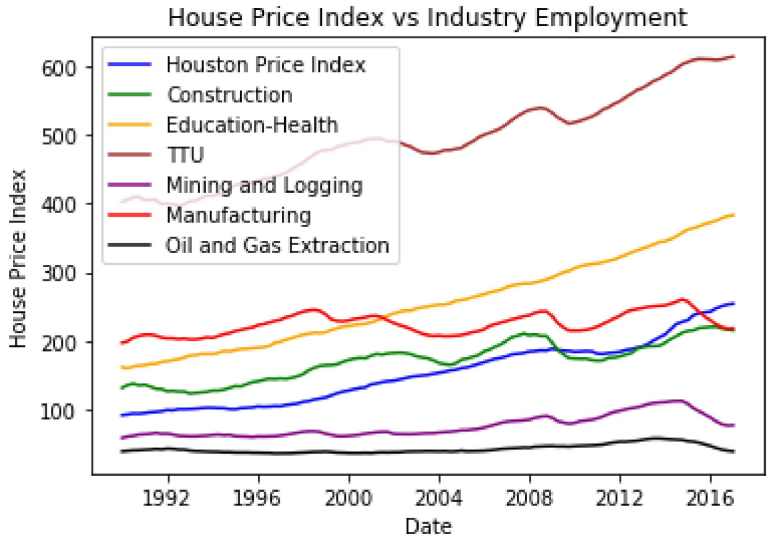
Out[15]:

```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x000001FEE72CE4E0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x000001FEE730A908
>],
      dtype=object)
```



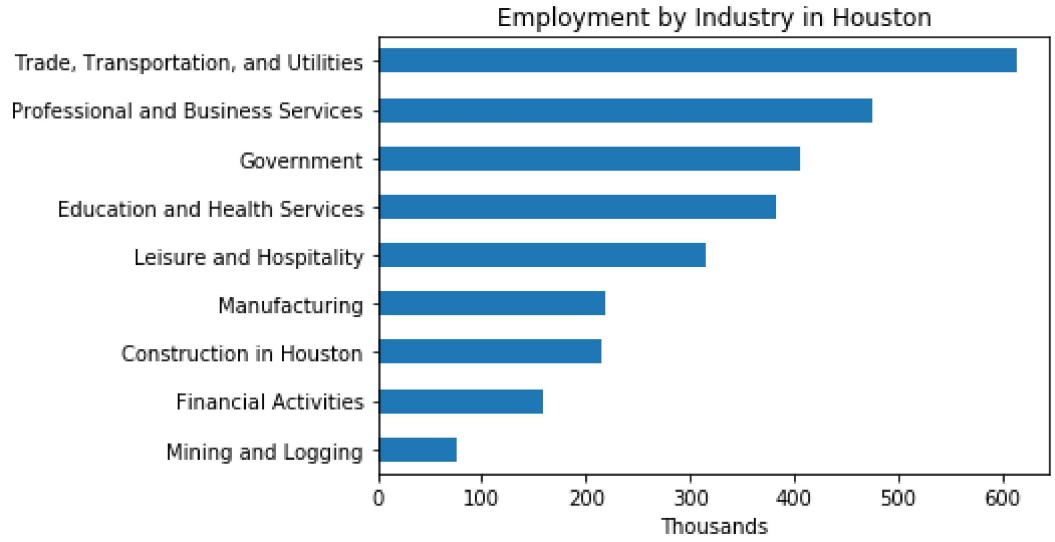
In [16]:

```
plt.plot(x1_axis, df["House Price Index for Houston"],color="blue", label="Houston Price Ir
plt.plot(x1_axis, df["Construction in Houston"],color="green", label="Construction" )
plt.plot(x1_axis, df["Education and Health Services"],color="orange", label="Education-Heal
plt.plot(x1_axis, df["Trade, Transportation, and Utilities"],color="brown", label="TTU" )
plt.plot(x1_axis, df["Mining and Logging"],color="purple", label="Mining and Logging" )
plt.plot(x1_axis, df["Manufacturing"],color="red", label="Manufacturing" )
plt.plot(x1_axis, df["Oil and Gas Extraction"],color="black", label="Oil and Gas Extractior

plt.legend(loc="best")

plt.title(f"House Price Index vs Industry Employment")
plt.ylabel("House Price Index")
plt.xlabel("Date")

plt.savefig("output_data/House Price Index vs Industry Employment")
plt.show()
```



In [33]:

```
df2 = df.tail(1)
df2

cols = [0,1,10,12,13,14,15,16]
df2.drop(df2.columns[cols],axis=1,inplace=True)

df2 = df2.reset_index(drop=True).T
df2=df2.sort_values(by=[0], ascending=True)

#fig2, df2 = plt.subplots(figsize=(14,4), linewidth=5)

df2.plot(kind='barh',  title ='Employment by Industry in Houston',legend=None)
plt.xlabel("Thousands")
plt.show()
df2
```

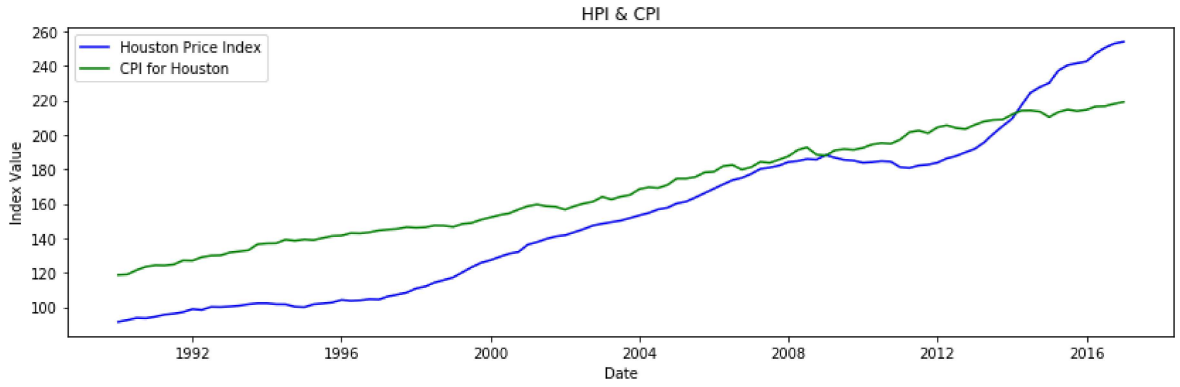


Out[33]:

	0
Mining and Logging	76.6
Financial Activities	158.1
Construction in Houston	215.3
Manufacturing	218.0
Leisure and Hospitality	315.4
Education and Health Services	383.3
Government	405.0
Professional and Business Services	475.6
Trade, Transportation, and Utilities	614.4

In [18]:

```
fig3, ax = plt.subplots(figsize=(14,4), linewidth=5)
ax.plot(x1_axis, df["House Price Index for Houston"],color="blue", label="Houston Price Inc
#ax.plot(x1_axis, df["30Yr Fixed Mortgage Rate"],color="green", label="Mortgage Rate" )
ax.plot(x1_axis, df["CPI for Houston"],color="green", label="CPI for Houston" )

plt.legend(loc="best")

plt.title(f"HPI & CPI")
plt.ylabel("Index Value")
plt.xlabel("Date")

plt.savefig("output_data/HPI vs Mortagage vs CPI")
plt.show()
```



In [19]:

```
dff.head()
```

Out[19]:

	Population in Harris County	Population in Fort Bend County	Population in Montgomery County	Population in Galveston County	Population in Brazoria County	Population in Liberty County	Population in Waller County	Pop Ch
1990-01-01	2835.927	228.191	184.066	218.363	192.644	52.887	23.507	
1991-01-01	2912.041	240.604	192.732	222.854	198.707	53.848	23.474	
1992-01-01	2982.258	253.352	202.374	227.406	203.275	55.051	24.655	
1993-01-01	3033.757	264.940	211.878	232.199	208.005	56.645	25.718	
1994-01-01	3080.698	275.507	221.428	234.557	212.209	58.925	26.716	

In [20]:

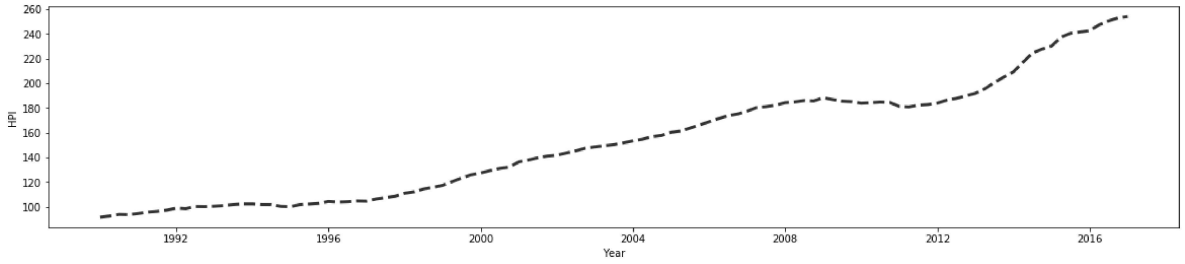
```
import datetime as dt
dff.columns
dff['Houston_Population']=dff['Population in Harris County']\
                        +dff['Population in Fort Bend County']\
                        +dff['Population in Montgomery County']\
+dff['Population in Galveston County']\
+dff['Population in Brazoria County']\
+dff['Population in Liberty County']\
+dff['Population in Waller County']\
+dff['Population in Chambers County']\
+dff['Population in Austin County']


dff['Difference'] = dff['Houston_Population'].diff()


dff.head()
```

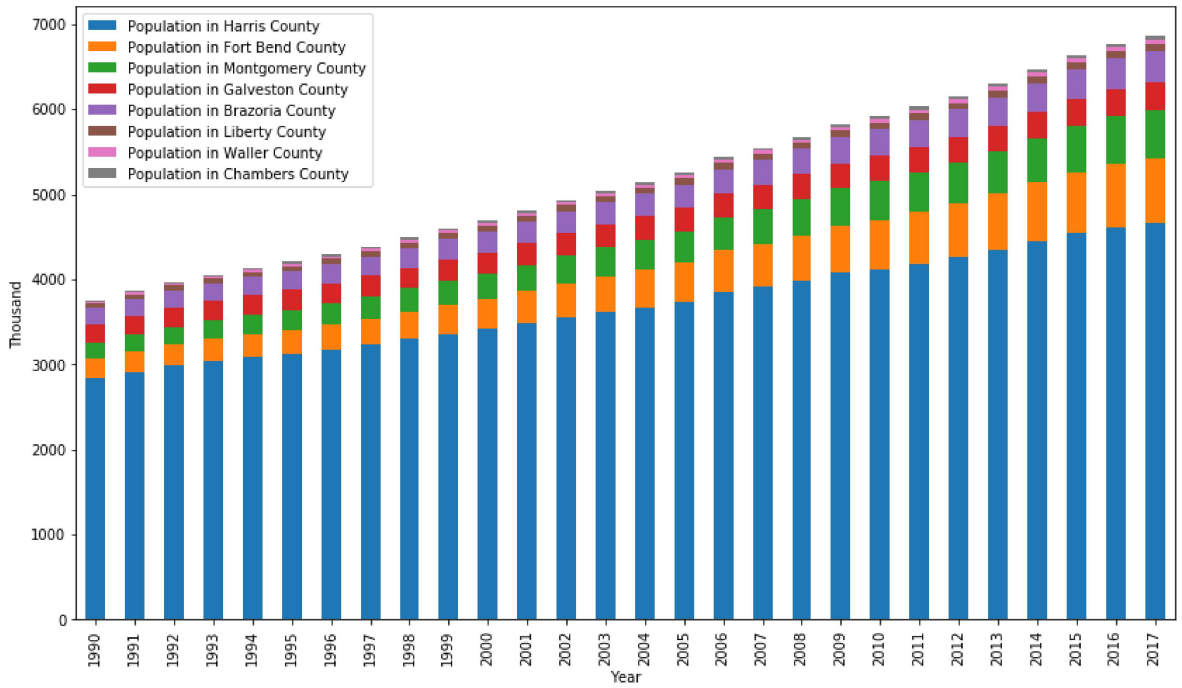
Out[20]:

	Population in Harris County	Population in Fort Bend County	Population in Montgomery County	Population in Galveston County	Population in Brazoria County	Population in Liberty County	Population in Waller County	Pop Cha
1990-01-01	2835.927	228.191	184.066	218.363	192.644	52.887	23.507	
1991-01-01	2912.041	240.604	192.732	222.854	198.707	53.848	23.474	
1992-01-01	2982.258	253.352	202.374	227.406	203.275	55.051	24.655	
1993-01-01	3033.757	264.940	211.878	232.199	208.005	56.645	25.718	
1994-01-01	3080.698	275.507	221.428	234.557	212.209	58.925	26.716	

In [21]:

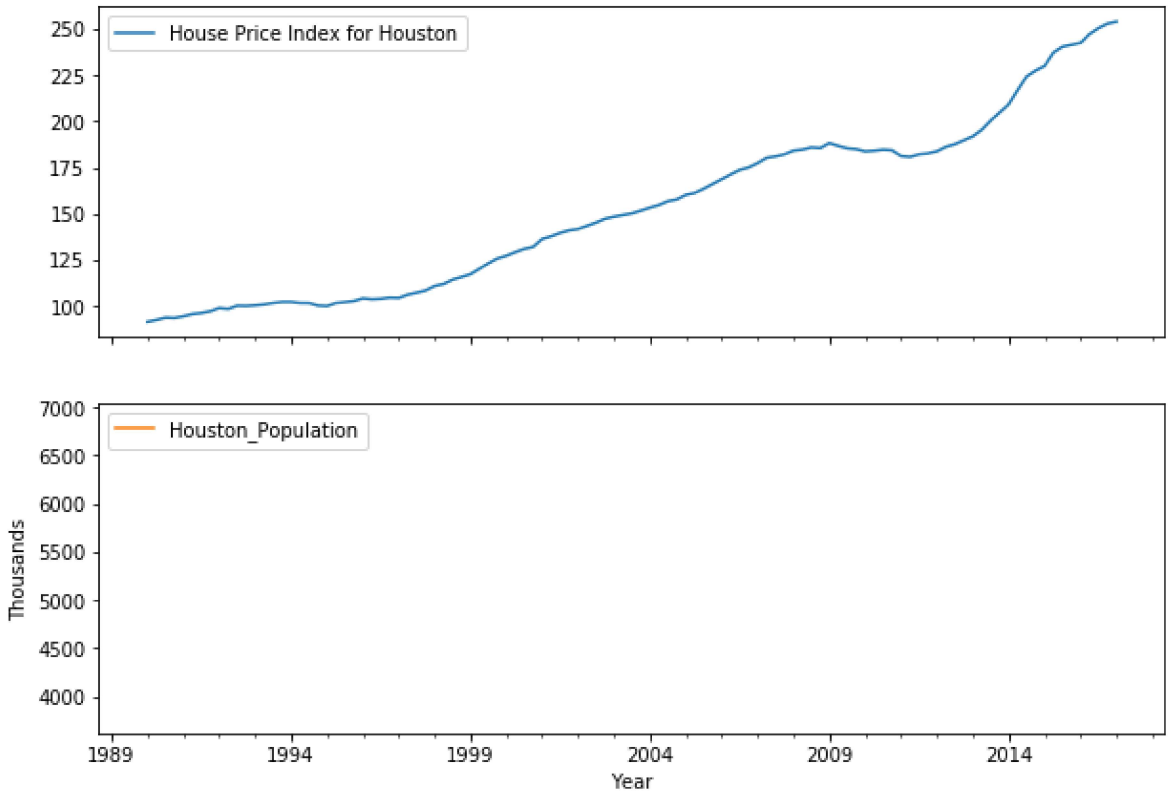
```
import datetime as dt
dates = pd.Series(range(1990,2018))
dates
dates[1]
pop = dff.iloc[:,0:8]
pop.head()
#cols = [9,10]
#pop = pop.drop(pop.columns[cols],axis=1,inplace=True)
#pop
fig2, ax = plt.subplots(figsize=(20,4), linewidth=5)

ax.plot(years, HPI, linestyle='--', linewidth=3, color='.2', label='HPI')
plt.xlabel("Year")
plt.ylabel("HPI")



pop.plot.bar(stacked=True, figsize=(14,8))
plt.xticks(np.arange(28), dates[0:28])


#plt.xticks([dates])

plt.ylabel("Thousand")
plt.xlabel("Year")
plt.show()
```





In [22]:

```
#plot_cols = [df["House Price Index for Houston"],dff["Houston_Population"]]
#plot_cols
#df["House Price Index for Houston"].describe
#dff["Houston_Population"].describe
#plot_cols = [df["House Price Index for Houston"],dff["Houston_Population"]]
df_all_cols = pd.concat([df["House Price Index for Houston"],dff["Houston_Population"]], ax
#df_all_cols
fig, axes = plt.subplots(2,1, figsize=(10,7), sharex=True)
df_all_cols.plot(subplots=True, ax=axes)

#plt.subplot(2, 1, 1)
#plt.plot(x1_axis, df_all_cols["House Price Index for Houston"], 'o-')
#plt.title('A tale of 2 subplots')
plt.ylabel('House Price Index')

#plt.subplot(2, 1, 2)
#plt.plot(x1_axis, df_all_cols["Houston_Population"], '.-')
plt.xlabel('Year')
plt.ylabel('Thousands')

plt.show()
```



In [23]:

```
new_df = df["House Price Index for Houston"]
new_df['Houston_Population'] = None
#new_df
```

In [24]:

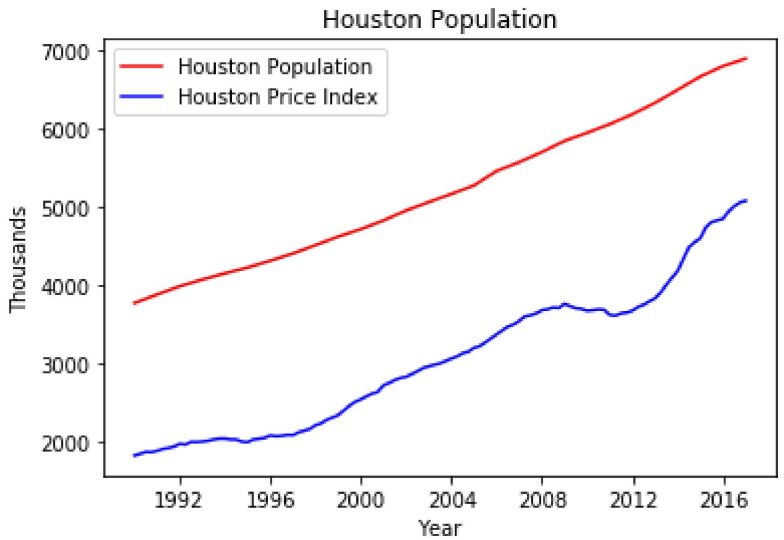
```
df["Houston_Population"] = dff['Difference']
df.to_csv('output_data/Houston_df.csv')
df.head()
```

Out[24]:

	House Price Index for Houston	Unemployment Rate in Houston	Mining and Logging	Manufacturing	Trade, Transportation, and Utilities	Financial Activities	Professiona an Busines Service
1990-01-01	91.49	5.0	58.2	197.0	402.7	113.3	202.
1990-04-01	92.52	5.0	60.2	199.2	405.8	115.0	211.
1990-07-01	93.75	5.1	61.2	204.5	409.4	114.8	215.
1990-10-01	93.56	5.4	63.1	207.0	409.6	114.6	220.
1991-01-01	94.46	5.4	63.5	208.8	405.1	113.8	223.

In [25]:

```
x2_axis =dff.index.values
plt.plot(x2_axis, dff["Houston_Population"],color="red", label="Houston Population" )
plt.plot(x1_axis, df["House Price Index for Houston"].values*20,color="blue", label="Houstc
plt.legend(loc="best")

plt.title(f"Houston Population")
plt.ylabel("Thousands")
plt.xlabel("Year")

plt.savefig("Houston Population")
plt.show()
```



In [26]:

```
plt.plot(x1_axis, df["Unemployment Rate in Houston"],color="blue", label="Unemployment Rate

plt.legend(loc="best")

plt.title(f"Unemployment Rate in Houston")
plt.ylabel("Percent")
plt.xlabel("Year")

plt.savefig("output_data/Unemployment Rate in Houston")
plt.show()
```

In [27]:

```
BP = pd.read_csv('building_permits - annual - 1990.csv')
BP.head()
```

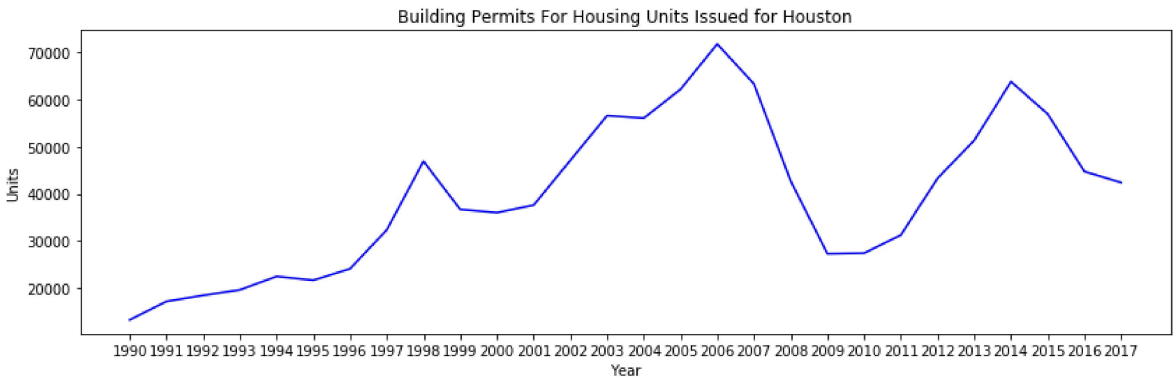
Out[27]:

	DATE	BPPRIV048039	BPPRIV048291	BPPRIV048071	BPPRIV048201	BPPRIV048473	BPPRIV
0	01 01 1990	940	174	21	10151	4	
1	01 01 1991	952	155	78	12647	9	
2	01 01 1992	1463	175	125	12558	18	
3	01 01 1993	1388	213	182	13222	92	
4	01 01 1994	1198	262	176	15855	110	

In [28]:

```
#plt.plot(BP['DATE'], BP["Total_Units"],color="blue", label="Building Permits" )

#plt.legend(loc="best")

fig4, ax = plt.subplots(figsize=(14,4), linewidth=5)
ax.plot(BP['DATE'], BP["Total_Units"],color="blue", label="Building Permits" )

plt.title(f"Building Permits For Housing Units Issued for Houston")
plt.ylabel("Units")
plt.xlabel("Year")
plt.locator_params(axis='x', nbins=100)
plt.savefig("output_data/Unemployment Rate in Houston")
dates = pd.Series(range(1990,2018))
plt.xticks(np.arange(28), dates[0:28])

plt.show()
```

```
C:\Users\User\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\matpl
otlib\ticker.py:1437: UserWarning: 'set_params()' not defined for locator of
type <class 'matplotlib.category.StrCategoryLocator'>
  str(type(self)))
```



In [29]:

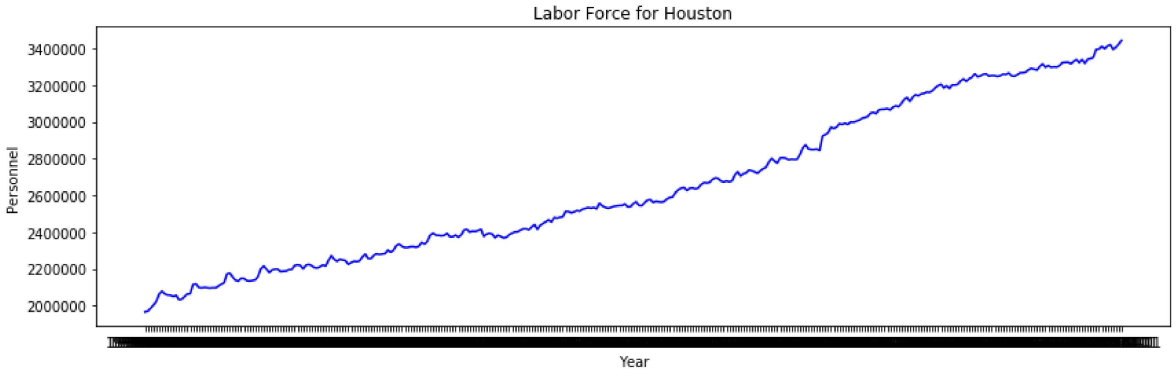
```
LF = pd.read_csv('Civilian labor Force - monthly - 1990.csv')
LF.head()
```

Out[29]:

	DATE	HOUS448LFN
0	1990-01-01	1966455
1	1990-02-01	1971027
2	1990-03-01	1987424
3	1990-04-01	2003628
4	1990-05-01	2024329

In [30]:

```
LF_axis = LF['DATE'].values

fig4, ax = plt.subplots(figsize=(14,4), linewidth=5)
ax.plot(LF_axis, LF["HOUS448LFN"],color="blue", label="Labor Force" )

plt.title(f"Labor Force for Houston")
plt.ylabel("Personnel")
plt.xlabel("Year")
#plt.locator_params(axis='x', nbins=100)
#dates = pd.Series(range(1990,2018))
#plt.xticks(np.arange(28), dates[0:28])

plt.show()
```



In [31]:

```
EC = pd.read_csv('Economic conditions Index - monthly.csv')
EC.head()
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-31-fbfb244bc3d3> in <module>
----> 1 EC = pd.read_csv('Economic conditions Index - monthly.csv')
      2 EC.head()

~\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\pandas\io\parser
s.py in parser_f(filepath_or_buffer, sep, delimiter, header, names, index_co
l, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, tr
ue_values, false_values, skipinitialspace, skiprows, nrows, na_values, keep_
default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetim
e_format, keep_date_col, date_parser, dayfirst, iterator, chunksize, compres
sion, thousands, decimal, lineterminator, quotechar, quoting, escapechar, co
mment, encoding, dialect, tupleize_cols, error_bad_lines, warn_bad_lines, sk
ipfooter, doublequote, delim_whitespace, low_memory, memory_map, float_preci
sion)
    676                     skip_blank_lines=skip_blank_lines)
    677
--> 678         return _read(filepath_or_buffer, kwds)
    679
    680     parser_f.__name__ = name

~\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\pandas\io\parser
s.py in _read(filepath_or_buffer, kwds)
    438
    439     # Create the parser.
--> 440     parser = TextFileReader(filepath_or_buffer, **kwds)
    441
    442     if chunksize or iterator:

~\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\pandas\io\parser
s.py in __init__(self, f, engine, **kwds)
    785             self.options['has_index_names'] = kwds['has_index_names'
]
    786
--> 787         self._make_engine(self.engine)
    788
    789     def close(self):

~\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\pandas\io\parser
s.py in _make_engine(self, engine)
   1012     def _make_engine(self, engine='c'):
   1013         if engine == 'c':
-> 1014             self._engine = CParserWrapper(self.f, **self.options)
   1015         else:
   1016             if engine == 'python':

~\Miniconda3\envs\Tensorflow_11_Oct_2018\lib\site-packages\pandas\io\parser
s.py in __init__(self, src, **kwds)
   1706         kwds['usecols'] = self.usecols
   1707
-> 1708         self._reader = parsers.TextReader(src, **kwds)
   1709
   1710         passed_names = self.names is None
```

```
pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader.__cinit__()

pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader._setup_parser_so
urce()

FileNotFoundError: File b'Economic conditions Index - monthly.csv' does not
 exist
```

In [ ]: