In [1]:

```
import pandas as pd
pd.set_option("display.float_format", lambda x: "%.3f" % x)
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(color_codes=True)

from sklearn.cluster import KMeans
color = sns.color_palette()

from IPython.core.display import display, HTML
display(HTML("{ width:100% !important; }"))
%matplotlib inline
```

{ width:100% !important; }

In [4]:

```
df = pd.read_csv("Housing.csv")
#df
```

In [6]:

```
df.describe()
```

Out[6]:

	RegionID	SizeRank	RegionName	2000-01-31	2000-02-29	2000-03-31	2000-04-30	
count	27339.000	27339.000	27339.000	13489.000	13579.000	13595.000	13622.000	
mean	80477.882	14392.359	48723.529	154473.301	154766.438	155239.430	156342.172	
std	26006.407	8742.151	27433.984	115853.572	116318.670	117092.769	118726.921	
min	58196.000	0.000	1001.000	9417.000	9762.000	10033.000	10663.000	
25%	68985.000	6900.500	26282.500	86786.000	86800.500	86951.000	87252.250	
50%	79012.000	13888.000	48091.000	127646.000	127934.000	128139.000	128776.500	
75%	89258.500	21520.000	71751.500	185683.000	185890.000	186415.500	187485.250	
max	753844.000	34430.000	99901.000	2598211.000	2627048.000	2665734.000	2745102.000	2

8 rows × 271 columns

In [7]:

```
df.rename(columns={"RegionName":"ZipCode"}, inplace=True)
df["ZipCode"]=df["ZipCode"].map(lambda x: "{:.0f}".format(x))
df["RegionID"]=df["RegionID"].map(lambda x: "{:.0f}".format(x))
df.head()
```

Out[7]:

	RegionID	SizeRank	ZipCode	RegionType	StateName	State	City	Metro	CountyName
0	61639	0	10025	Zip	NY	NY	New York	New York-Newark-Jersey City	New York County
1	84654	1	60657	Zip	IL	IL	Chicago	Chicago-Naperville-Elgin	Cook County
2	61637	2	10023	Zip	NY	NY	New York	New York-Newark-Jersey City	New York County
3	91982	3	77494	Zip	TX	TX	Katy	Houston-The Woodlands-Sugar Land	Harris County
4	84616	4	60614	Zip	IL	IL	Chicago	Chicago-Naperville-Elgin	Cook County

5 rows × 277 columns

In [8]:

```
median_prices = df.median()
```

```
C:\Users\HP\AppData\Local\Temp/ipykernel_5456/376284158.py:1: FutureWarning: Dr
opping of nuisance columns in DataFrame reductions (with 'numeric_only=None') i
s deprecated; in a future version this will raise TypeError.  Select only valid
columns before calling the reduction.
  median_prices = df.median()
```

In [10]:

```
median_prices.tail()
```

Out[10]:

```
2021-12-31   220384.000
2022-01-31   223411.500
2022-02-28   226808.000
2022-03-31   228971.000
2022-04-30   231152.000
dtype: float64
```

In [17]:

```
#sf_prices = df["RegionName"] == "San Francisco".median()
sc_f = df[df["CountyName"] == "Santa Clara"].median()
sf_df = df[df["City"] == "San Francisco"].median()
los_ang= df[df["City"] == "Palo Alto"].median()
df_comparison = pd.concat([marin_df, sf_df, palo_alto, median_prices], axis=1)
df_comparison.columns = ["Marin County", "San Francisco", "Palo Alto", "Median US
import cufflinks as cf
cf.go_offline()
df_comparison.iplot(title="Bay Area Median Single Family Home Prices 2000-2022",
                    xTitle="Year",
                    yTitle="Sales Price",
                   #bestfit=True, bestfit_colors=["pink"],
                   #subplots=True,
                   shape=(4,1),
                    #subplot_titles=True,
                    fill=True,)
```

```
C:\Users\HP\AppData\Local\Temp/ipykernel_5456/3629507379.py:3: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.

C:\Users\HP\AppData\Local\Temp/ipykernel_5456/3629507379.py:4: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.
```

In [18]:

```
marin_df = df[df["CountyName"] == "Marin"].median()
sf_df = df[df["City"] == "San Francisco"].median()
palo_alto = df[df["City"] == "Palo Alto"].median()
df_comparison = pd.concat([marin_df, sf_df, palo_alto, median_prices], axis=1)
df_comparison.columns = ["Marin County","San Francisco", "Palo Alto", "Median USA
```

```
C:\Users\HP\AppData\Local\Temp/ipykernel_5456/1872928650.py:2: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.

C:\Users\HP\AppData\Local\Temp/ipykernel_5456/1872928650.py:3: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.
```

In [25]:

```
ca_df = df[df["StateName"] == "CA"].median()
sf_df = df[df["CountyName"] == "San Francisco"].median()
los_ang = df[df["CountyName"] == "Los Angeles"].median()
df_comparison = pd.concat([ca_df, sf_df, los_ang, median_prices], axis=1)
df_comparison.columns = ["CA state","San Francisco", "Los Angeles", "Median USA"]
```

```
C:\Users\HP\AppData\Local\Temp/ipykernel_5456/3653745613.py:1: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.
```

In [28]:

```
import cufflinks as cf
cf.go_offline()
df_comparison.iplot(title="CA MedianSingle Family Home Prices 2000-2022",
    xTitle="Year", yTitle="Sales Price",#bestfit=True, bestfit_colors=["pink"],
    #subplots=True,
    shape=(4,1),
    #subplot_titles=True, fill=True,)
    fill=True)
```

In [ ]:

```
```

In [20]:

```
cf.go_offline()
df_comparison.iplot(title="Bay Area MedianSingle Family Home Prices 2000-2022",
    xTitle="Year",yTitle="Sales Price",#bestfit=True, bestfit_colors=["pink"],
    #subplots=True,
    shape=(4,1),
    #subplot_titles=True, fill=True,)
    fill=True)
```

In [22]:

```
NY_df = df[df["StateName"] == "NY"].median()
nyc_df = df[df["City"] == "NY"].median()
#palo_alto = df[df["City"] == "Palo Alto"].median()
df_comparison = pd.concat([NY_df, nyc_df, median_prices], axis=1)
df_comparison.columns = ["NY State","New York", "Median USA"]
```

```
C:\Users\HP\AppData\Local\Temp/ipykernel_5456/66171252.py:1: FutureWarning:

Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only vali
d columns before calling the reduction.
```

In [23]:

```
cf.go_offline()
df_comparison.iplot(title="New York MedianSingle Family Home Prices 2000-2022",
    xTitle="Year",yTitle="Sales Price",#bestfit=True, bestfit_colors=["pink"],
    #subplots=True,
    shape=(4,1),
    #subplot_titles=True, fill=True,)
    fill=True)
```

In [ ]:

In [40]:

```python
#Cluster on Size Rank and Price
from sklearn.preprocessing import MinMaxScaler
columns_to_drop = ['RegionID', 'ZipCode', 'City', 'State', 'Metro', 'CountyName',
df_numerical = df.dropna()
df_numerical = df_numerical.drop(columns_to_drop, axis=1)
df_numerical.describe()
```

Out[40]:

	SizeRank	2000-01-31	2000-02-29	2000-03-31	2000-04-30	2000-05-31	2000-06-30
count	9520.000	9520.000	9520.000	9520.000	9520.000	9520.000	9520.000
mean	9363.613	169123.017	169739.046	170370.277	171694.384	173007.987	174345.834
std	7090.340	120153.672	120904.209	121779.754	123617.930	125528.648	127506.702
min	3.000	9417.000	9762.000	10033.000	10663.000	11208.000	11693.000
25%	3592.500	99590.500	99842.000	100079.250	100570.000	100912.250	101364.500
50%	7788.000	142323.500	142779.000	143231.000	144092.500	144964.000	145701.500
75%	13824.000	201702.750	202373.000	202848.250	204332.250	205877.000	207005.250
max	34430.000	2598211.000	2627048.000	2665734.000	2745102.000	2837169.000	2934390.000

8 rows × 269 columns

In [41]:

```python
df_numerical
```

Out[41]:

	SizeRank	2000-01-31	2000-02-29	2000-03-31	2000-04-30	2000-05-31	2000-06-30	2000-07
3	3	225097.000	225416.000	226142.000	227372.000	226933.000	226596.000	225474.
5	5	106812.000	106825.000	106628.000	106633.000	106664.000	106876.000	107030.
6	6	409351.000	403981.000	404189.000	402311.000	407324.000	410228.000	414704.
7	7	106328.000	106273.000	106016.000	105964.000	105930.000	106096.000	106136.
8	8	90610.000	90610.000	90643.000	90609.000	90628.000	90588.000	90573.
...	...	...	...	...	...	...	...	
27310	33487	283986.000	283502.000	286100.000	289885.000	293778.000	297077.000	302631.
27313	33581	205652.000	206110.000	206667.000	209776.000	213670.000	216877.000	216560.
27331	34322	135840.000	136537.000	137563.000	137586.000	137970.000	139003.000	141185.
27335	34430	486003.000	486258.000	489144.000	491714.000	493964.000	496664.000	502387.
27337	34430	126208.000	125243.000	123175.000	119849.000	117842.000	117244.000	116315.

9520 rows × 269 columns

In [42]:

```
scaler = MinMaxScaler()
scaled_df = scaler.fit_transform(df_numerical)
kmeans = KMeans(n_clusters=3, random_state=0).fit(scaled_df)
print(len(kmeans.labels_))
```

```
9520
```

In [44]:

```
cluster_df = df.copy(deep=True)
cluster_df.dropna(inplace=True)
cluster_df.describe()
cluster_df['cluster'] = kmeans.labels_
cluster_df['appreciation_ratio'] = round(cluster_df["2017-09"]/cluster_df["1996-0
cluster_df['CityZipCodeAppRatio'] = cluster_df['City'].map(str) + "-" + cluster_c
cluster_df.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key,
method, tolerance)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:

~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.index.Inde
xEngine.get_loc()

~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.index.Inde
xEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashT
able.get_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashT
able.get_item()

KeyError: '2017-09'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_5456/2927337548.py in <module>
      3 cluster_df.describe()
      4 cluster_df['cluster'] = kmeans.labels_
----> 5 cluster_df['appreciation_ratio'] = round(cluster_df["2017-09"]/cluster_
df["1996-04"],2)
      6 cluster_df['CityZipCodeAppRatio'] = cluster_df['City'].map(str) + "-" +
cluster_df['ZipCode'] + "-" + cluster_df["appreciation_ratio"].map(str)
      7 cluster_df.head()

~\anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   3456             if self.columns.nlevels > 1:
   3457                 return self._getitem_multilevel(key)
-> 3458             indexer = self.columns.get_loc(key)
   3459             if is_integer(indexer):
   3460                 indexer = [indexer]

~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key,
method, tolerance)
   3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:
-> 3363                 raise KeyError(key) from err
   3364 
   3365         if is_scalar(key) and isna(key) and not self.hasnans:
```

```
KeyError: '2017-09'
```

```
In [ ]:
```