Assignment 3: Feature Stores

Credits: I used the Amazon Documentation Developer Guide to help with the feature store. Link provided HERE

Creating Feature Store Session

In [ ]:

```
# Import the necessary libraries
# Libraries for creating feature store session
import boto3
import sagemaker
from sagemaker.session import Session
from sagemaker import get_execution_role

# Libraries for interacting with the dataset
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import io

# Feature Group
import time
from time import gmtime, strftime, sleep
from sagemaker.feature_store.feature_group import FeatureGroup
```

```
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /root/.config/sagemaker/config.yaml
```

In [ ]:

```
# Helpful Functions + Variables stored here
def encode_col(df, col):
    names = df[col].unique()
    values = len(names)
    dict_pairs = dict([(key,value) for _, (key,value) in enumerate(zip(names,np.arange(values)))])
    df[col] = df[col].map(dict_pairs)
    df[col] = df[col].astype('float64')
    return df, dict_pairs

# def encode_col(df, col):
#     df[col], _ = df[col].factorize()
```

```python
#     df[col] = df[col].astype('float64')
#     return df

def convert_to_strings(df):
    for col in df.columns:
        if df.dtypes[col] == 'object':
            df[col] = df[col].astype(str)

def wait_for_feature_group_creation_complete(feature_group):
    status = feature_group.describe().get("FeatureGroupStatus")
    while status == "Creating":
        print("Waiting for Feature Group Creation")
        time.sleep(5)
        status = feature_group.describe().get("FeatureGroupStatus")
    if status != "Created":
        raise RuntimeError(f"Failed to create feature group {feature_group.name}")
    print(f"FeatureGroup {feature_group.name} successfully created.")


time_now = int(round(time.time()))
```

In [ ]:

```python
# Create the session by identifying the variables
region = boto3.Session().region_name

boto_session = boto3.Session(region_name=region)

sagemaker_client = boto_session.client(service_name="sagemaker", region_name=region)
featurestore_runtime = boto_session.client(
    service_name="sagemaker-featurestore-runtime", region_name=region
)

feature_store_session = Session(
    boto_session=boto_session,
    sagemaker_client=sagemaker_client,
    sagemaker_featurestore_runtime_client=featurestore_runtime,
)
```

In [ ]:

```python
# Creating default bucket
default_s3_bucket_name = feature_store_session.default_bucket()
prefix = "sagemaker-featurestore-demo"
```

```
print(default_s3_bucket_name)
```

```
sagemaker-us-east-1-004608622582
```

In [ ]:

```
# Grab Role
role = get_execution_role()
print(role)
```

```
arn:aws:iam::004608622582:role/LabRole
```

In [ ]:

```
# Start the client + feature store runtime
sagemaker_client = boto_session.client(service_name='sagemaker', region_name=region)
featurestore_runtime = boto_session.client(service_name='sagemaker-featurestore-runtime', region_name=region)
```

In [ ]:

```
# Create feature store session
feature_store_session = Session(boto_session=boto_session, sagemaker_client=sagemaker_client, sagemaker_featurestore_
```

Loading Data and Partitioning it into DataGroups

In [ ]:

```
# Reading in the data
housing_df = pd.read_csv('housing.csv')
housing_gmaps_df =  pd.read_csv('housing_gmaps_data_raw.csv')
```

In [ ]:

```
housing_df.head()
```

Out[ ]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_hou
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	

In [ ]:

```
housing_gmaps_df.head()
```

Out[ ]:

	street_number	route	locality-political	administrative_area_level_2-political	administrative_area_level_1-political	country-political	postal_code	address
0	3130	Grizzly Peak Boulevard	Berkeley	Alameda County	California	United States	94705.0	3130 Grizzly Peak Blvd, Berkeley, CA 94705, USA
1	2005	Tunnel Road	Oakland	Alameda County	California	United States	94611.0	2005 Tunnel Rd, Oakland, CA 94611, USA
2	6886	Chabot Road	Oakland	Alameda County	California	United States	94618.0	6886 Chabot Rd, Oakland, CA 94618, USA
3	6365	Florio Street	Oakland	Alameda County	California	United States	94618.0	6365 Florio St, Oakland, CA 94618, USA
4	5407	Bryant Avenue	Oakland	Alameda County	California	United States	94618.0	5407 Bryant Ave, Oakland,

street_number	route	locality-political	administrative_area_level_2-political	administrative_area_level_1-political	country-political	postal_code	address
							CA 94618, USA

5 rows × 30 columns

```
In [ ]: housing_df.describe()
```

Out[ ]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_inc
count	20640.000000	20640.000000	20640.000000	20640.000000	20433.000000	20640.000000	20640.000000	20640.000
mean	-119.569704	35.631861	28.639486	2635.763081	537.870553	1425.476744	499.539680	3.870
std	2.003532	2.135952	12.585558	2181.615252	421.385070	1132.462122	382.329753	1.899
min	-124.350000	32.540000	1.000000	2.000000	1.000000	3.000000	1.000000	0.499
25%	-121.800000	33.930000	18.000000	1447.750000	296.000000	787.000000	280.000000	2.563
50%	-118.490000	34.260000	29.000000	2127.000000	435.000000	1166.000000	409.000000	3.534
75%	-118.010000	37.710000	37.000000	3148.000000	647.000000	1725.000000	605.000000	4.743
max	-114.310000	41.950000	52.000000	39320.000000	6445.000000	35682.000000	6082.000000	15.000

```
In [ ]: housing_gmaps_df.describe()
```

Out[ ]:

	postal_code	longitude	latitude	postal_code_suffix
count	12410.000000	12590.000000	12590.000000	7999.000000
mean	93348.943836	-119.676724	35.895577	4177.914614
std	1765.572652	2.042677	2.219248	2474.063791
min	85344.000000	-124.350000	32.540000	110.000000
25%	92054.000000	-121.760000	33.970000	2230.500000
50%	93301.000000	-119.270000	35.340000	3556.000000
75%	95050.000000	-117.950000	37.810000	5529.000000
max	96161.000000	-114.310000	41.950000	9859.000000

In [ ]:

```
df = pd.merge(housing_gmaps_df, housing_df, left_on=['longitude', 'latitude'], right_on=['longitude', 'latitude'], ho
```

In [ ]:

```
df.head()
```

Out[ ]:

	street_number	route	locality-political	administrative_area_level_2-political	administrative_area_level_1-political	country-political	postal_code	address
0	3130	Grizzly Peak Boulevard	Berkeley	Alameda County	California	United States	94705.0	3130 Grizzly Peak Blvd, Berkeley, CA 94705, USA
1	2005	Tunnel Road	Oakland	Alameda County	California	United States	94611.0	2005 Tunnel Rd, Oakland, CA 94611, USA
2	6886	Chabot Road	Oakland	Alameda County	California	United States	94618.0	6886 Chabot Rd, Oakland, CA 94618, USA
3	6365	Florio Street	Oakland	Alameda County	California	United States	94618.0	6365 Florio St, Oakland, CA 94618, USA
4	6365	Florio Street	Oakland	Alameda County	California	United States	94618.0	6365 Florio St, Oakland, CA

street_number	route	locality-political	administrative_area_level_2-political	administrative_area_level_1-political	country-political	postal_code	address
							94618, USA

5 rows × 38 columns

In [ ]:
```
# Priority Key
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 38 columns):
 #   Column                                                                              Non-Null Count  Dtype  
---  ------                                                                              --------------  -----  
 0   street_number                                                                       19008 non-null  object 
 1   route                                                                               20091 non-null  object 
 2   locality-political                                                                  20452 non-null  object 
 3   administrative_area_level_2-political                                               20589 non-null  object 
 4   administrative_area_level_1-political                                               20637 non-null  object 
 5   country-political                                                                   20640 non-null  object 
 6   postal_code                                                                         20454 non-null  float64
 7   address                                                                             20640 non-null  object 
 8   longitude                                                                           20640 non-null  float64
 9   latitude                                                                            20640 non-null  float64
 10  neighborhood-political                                                              9000 non-null   object 
 11  postal_code_suffix                                                                  14095 non-null  float64
 12  establishment-point_of_interest-transit_station                                     255 non-null    object 
 13  establishment-park-point_of_interest                                                46 non-null     object 
 14  premise                                                                             36 non-null     object 
 15  establishment-point_of_interest-subway_station-transit_station                      3 non-null      object 
 16  airport-establishment-finance-moving_company-point_of_interest-storage              1 non-null      object 
 17  subpremise                                                                          25 non-null     object 
 18  bus_station-establishment-point_of_interest-transit_station                         22 non-null     object 
 19  establishment-park-point_of_interest-tourist_attraction                             34 non-null     object 
 20  establishment-natural_feature                                                       11 non-null     object 
 21  airport-establishment-point_of_interest                                             8 non-null      object 
 22  political-sublocality-sublocality_level_1                                           33 non-null     object 
 23  administrative_area_level_3-political                                               1 non-null      object 
 24  post_box                                                                            6 non-null      object 
 25  establishment-light_rail_station-point_of_interest-transit_station                  13 non-null     object 
 26  establishment-point_of_interest                                                     1 non-null      object 
 27  aquarium-establishment-park-point_of_interest-tourist_attraction-zoo                1 non-null      object 
 28  campground-establishment-lodging-park-point_of_interest-rv_park-tourist_attraction  1 non-null      object 
 29  cemetery-establishment-park-point_of_interest                                       1 non-null      object 
 30  housing_median_age                                                                  20640 non-null  float64
 31  total_rooms                                                                         20640 non-null  float64
 32  total_bedrooms                                                                      20433 non-null  float64
 33  population                                                                          20640 non-null  float64
 34  households                                                                          20640 non-null  float64
 35  median_income                                                                       20640 non-null  float64
 36  median_house_value                                                                  20640 non-null  float64
 37  ocean_proximity                                                  20640 non-null  object
dtypes: float64(11), object(27)
memory usage: 6.0+ MB
```

In [ ]:

```
# Grabbing the features for our group
feature_cols = ['neighborhood-political',
                'ocean_proximity',
                'median_house_value',
                'housing_median_age',
                'households',
                'total_bedrooms',
                'locality-political']

# Creating new df based on the destired features
feature_df = df[feature_cols]

# Dropping an null values based on the primary key
feature_df = feature_df.dropna(subset='neighborhood-political')

# Renaming some of the columss for simplicity sake
feature_df = feature_df.rename(columns={'neighborhood-political':'nbh_pol',
                                        'locality-political': 'loc_pol',
                                        'ocean_proximity':'ocn_prox',
                                        'median_house_value': 'med_hse_val',
                                        'housing_median_age': 'hse_med_age',
                                        'households': 'tot_house',
                                        'total_bedrooms': 'tot_bed'}
                               )
```

In [ ]:

```
feature_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9000 entries, 1 to 20636
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   nbh_pol      9000 non-null   object 
 1   ocn_prox     9000 non-null   object 
 2   med_hse_val  9000 non-null   float64
 3   hse_med_age  9000 non-null   float64
 4   tot_house    9000 non-null   float64
 5   tot_bed      8911 non-null   float64
 6   loc_pol      8955 non-null   object 
dtypes: float64(4), object(3)
memory usage: 562.5+ KB
```

In [ ]:

```
# Households becase on locality
house_df = feature_df[['loc_pol',
                      'tot_house',
                      'tot_bed']]

# Finding average for locality Code
house_df = house_df.groupby('loc_pol').mean()

# Renaming total to averages for average colculation
house_df = house_df.rename(columns={'tot_bed': 'avg_bed',
                                    'tot_house': 'avg_house'}
                          )

# Finding average bedrooms per household
house_df['avg_bed_per_house'] = house_df['avg_bed'].div(house_df['avg_house'], axis=0).round()

# Merge the new df back into the feature_df
feature_df = pd.merge(feature_df, house_df, left_on=['loc_pol'], right_on=['loc_pol'], how='left')
```

In [ ]:

```
feature_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9000 entries, 0 to 8999
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   nbh_pol            9000 non-null   object 
 1   ocn_prox           9000 non-null   object 
 2   med_hse_val        9000 non-null   float64
 3   hse_med_age        9000 non-null   float64
 4   tot_house          9000 non-null   float64
 5   tot_bed            8911 non-null   float64
 6   loc_pol            8955 non-null   object 
 7   avg_house          8955 non-null   float64
 8   avg_bed            8954 non-null   float64
 9   avg_bed_per_house  8954 non-null   float64
dtypes: float64(7), object(3)
memory usage: 703.2+ KB
```

In [ ]:

```
# Encoding datatypes from objects to floats

# One hot encode ocn_prox
encode_ocn_prox = pd.get_dummies(feature_df['ocn_prox'], dtype='float64')

# Encode Locality
encode_loc_pol, dict_pairs = encode_col(feature_df,'loc_pol')

# Combine the two encoded columns together
combined_cols = pd.concat([encode_loc_pol,encode_ocn_prox, ], axis=1)
```

In [ ]:

```
combined_cols.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9000 entries, 0 to 8999
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   nbh_pol             9000 non-null   object 
 1   ocn_prox            9000 non-null   object 
 2   med_hse_val         9000 non-null   float64
 3   hse_med_age         9000 non-null   float64
 4   tot_house           9000 non-null   float64
 5   tot_bed             8911 non-null   float64
 6   loc_pol             9000 non-null   float64
 7   avg_house           8955 non-null   float64
 8   avg_bed             8954 non-null   float64
 9   avg_bed_per_house   8954 non-null   float64
 10  <1H OCEAN           9000 non-null   float64
 11  INLAND              9000 non-null   float64
 12  NEAR BAY            9000 non-null   float64
 13  NEAR OCEAN          9000 non-null   float64
dtypes: float64(12), object(2)
memory usage: 984.5+ KB
```

In [ ]:

```
# Now that we got what we need form ocn_prox, we can drop that column
combined_cols = combined_cols.drop(columns=['ocn_prox'])
```

In [ ]:

```
combined_cols.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9000 entries, 0 to 8999
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   nbh_pol            9000 non-null   object 
 1   med_hse_val        9000 non-null   float64
 2   hse_med_age        9000 non-null   float64
 3   tot_house          9000 non-null   float64
 4   tot_bed            8911 non-null   float64
 5   loc_pol            9000 non-null   float64
 6   avg_house          8955 non-null   float64
 7   avg_bed            8954 non-null   float64
 8   avg_bed_per_house  8954 non-null   float64
 9   <1H OCEAN          9000 non-null   float64
 10  INLAND             9000 non-null   float64
 11  NEAR BAY           9000 non-null   float64
 12  NEAR OCEAN         9000 non-null   float64
dtypes: float64(12), object(1)
memory usage: 914.2+ KB
```

In [ ]:

```
# Rename and Group the neighboorhoods and create an index out of them
new_df = combined_cols.groupby('nbh_pol').mean()
```

In [ ]:

```
new_df.head()
```

Out[ ]:

	med_hse_val	hse_med_age	tot_house	tot_bed	loc_pol	avg_house	avg_bed	avg_bed_per_house	<1H OCEAN	II
nbh_pol										
28 Palms	222200.000000	25.0	923.000000	939.000000	5.0	863.238806	894.686567	1.0	1.0	
Acorn Industrial	81300.000000	52.0	147.000000	244.000000	0.0	370.966197	397.541076	1.0	0.0	
Adams Hill	250733.333333	39.5	493.666667	520.166667	36.0	579.542056	614.600000	1.0	1.0	
Agua Mansa Industrial Corridor	112300.000000	17.0	516.000000	569.000000	138.0	516.000000	569.000000	1.0	0.0	
Al Tahoe	109180.000000	23.8	248.800000	399.800000	20.0	248.800000	399.800000	2.0	0.0	

In [ ]:

```
# Rename
new_df = new_df.reset_index().rename(columns={'index': 'nbh_pol',
                                              '<1H OCEAN': 'ls_1_ocn',
                                              'INLAND': 'inland',
                                              'NEAR BAY': 'nr_bay',
                                              'NEAR OCEAN': 'nr_ocn'}
                                    )
```

In [ ]:

```
new_df.head()
```

Out[ ]:

	nbh_pol	med_hse_val	hse_med_age	tot_house	tot_bed	loc_pol	avg_house	avg_bed	avg_bed_per_house	ls_1_ocn
0	28 Palms	222200.000000	25.0	923.000000	939.000000	5.0	863.238806	894.686567	1.0	1.0
1	Acorn Industrial	81300.000000	52.0	147.000000	244.000000	0.0	370.966197	397.541076	1.0	0.0
2	Adams Hill	250733.333333	39.5	493.666667	520.166667	36.0	579.542056	614.600000	1.0	1.0
3	Agua Mansa Industrial Corridor	112300.000000	17.0	516.000000	569.000000	138.0	516.000000	569.000000	1.0	0.0
4	Al Tahoe	109180.000000	23.8	248.800000	399.800000	20.0	248.800000	399.800000	2.0	0.0

In [ ]:

```
new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1306 entries, 0 to 1305
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   nbh_pol            1306 non-null   object 
 1   med_hse_val        1306 non-null   float64
 2   hse_med_age        1306 non-null   float64
 3   tot_house          1306 non-null   float64
 4   tot_bed            1300 non-null   float64
 5   loc_pol            1306 non-null   float64
 6   avg_house          1293 non-null   float64
 7   avg_bed            1292 non-null   float64
 8   avg_bed_per_house  1292 non-null   float64
 9   ls_1_ocn           1306 non-null   float64
 10  inland             1306 non-null   float64
 11  nr_bay             1306 non-null   float64
 12  nr_ocn             1306 non-null   float64
dtypes: float64(12), object(1)
memory usage: 132.8+ KB
```

In [ ]:

```
# Encode the nbh_pol
# Creating a dataframe
encode_df = pd.DataFrame.from_dict(dict_pairs, orient='index')

# Reset the index and name the column
encode_df = encode_df.reset_index().rename(columns={'index': 'nbh_pol_new',
                                                    0: 'nbh_pol_encode'}
                                          )
# encode_df.info()

# Ensure DF for the encoded values are the same
encode_df['nbh_pol_encode'] = encode_df['nbh_pol_encode'].astype('float64')
```

In [ ]:

```
encode_df
```

Out[ ]:

	nbh_pol_new	nbh_pol_encode
0	Oakland	0.0
1	Berkeley	1.0
2	San Leandro	2.0
3	Alameda	3.0
4	Hayward	4.0
...	...	...
200	Porterville	200.0
201	Ventura	201.0
202	Oxnard	202.0
203	Thousand Oaks	203.0
204	Davis	204.0

205 rows × 2 columns

In [ ]:

```
new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1306 entries, 0 to 1305
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   nbh_pol            1306 non-null   object 
 1   med_hse_val        1306 non-null   float64
 2   hse_med_age        1306 non-null   float64
 3   tot_house          1306 non-null   float64
 4   tot_bed            1300 non-null   float64
 5   loc_pol            1306 non-null   float64
 6   avg_house          1293 non-null   float64
 7   avg_bed            1292 non-null   float64
 8   avg_bed_per_house  1292 non-null   float64
 9   ls_1_ocn           1306 non-null   float64
 10  inland             1306 non-null   float64
 11  nr_bay             1306 non-null   float64
 12  nr_ocn             1306 non-null   float64
dtypes: float64(12), object(1)
memory usage: 132.8+ KB
```

In [ ]:

```
# Calculating the bedrooms per houseold
new_df['bed_per_hse'] = new_df['tot_bed'].div(new_df['tot_house'], axis=0)
```

In [ ]:

```
new_df.isna().count()
```

Out[ ]:

```
nbh_pol              1306
med_hse_val          1306
hse_med_age          1306
tot_house            1306
tot_bed              1306
loc_pol              1306
avg_house            1306
avg_bed              1306
avg_bed_per_house    1306
ls_1_ocn             1306
inland               1306
nr_bay               1306
nr_ocn               1306
bed_per_hse          1306
dtype: int64
```

```
In [ ]: # Checking out the cities to add
        new_df[new_df['nbh_pol']=='Brooktree'], new_df[new_df['nbh_pol']== "Fisherman's Wharf"], new_df[new_df['nbh_pol']=='
```

```
Out[ ]: (       nbh_pol  med_hse_val  hse_med_age  tot_house  tot_bed  loc_pol  \
         130  Brooktree     257400.0          9.0     1438.0      NaN    182.0   

               avg_house     avg_bed  avg_bed_per_house  ls_1_ocn  inland  nr_bay  \
         130  532.506148  548.538144                1.0       1.0     0.0     0.0   

              nr_ocn  bed_per_hse  
         130     0.0          NaN  ,
                        nbh_pol  med_hse_val  hse_med_age  tot_house  tot_bed  loc_pol  \
         390  Fisherman's Wharf     500001.0         52.0      250.0    317.0    160.0   

              avg_house     avg_bed  avg_bed_per_house  ls_1_ocn  inland  nr_bay  \
         390      501.0  535.384899                1.0       0.0     0.0     1.0   

              nr_ocn  bed_per_hse  
         390     0.0        1.268  ,
               nbh_pol  med_hse_val  hse_med_age  tot_house  tot_bed  loc_pol  \
         604  Los Osos     221612.5       15.375     611.75    642.5    163.0   

              avg_house  avg_bed  avg_bed_per_house  ls_1_ocn  inland  nr_bay  nr_ocn  \
         604     611.75    642.5                1.0       0.0     0.0     0.0     1.0   

              bed_per_hse  
         604     1.050266  )
```

Ingest Data into Feature Store + Setup Feature Group

```
In [ ]: # Creating the names and the time-stamp
        neighborhood_feature_group_name = "neighborhood-feature-group-" + strftime("%d-%H-%M-%S", gmtime())
        encoded_feature_group_name = "encoded-feature-group-" + strftime("%d-%H-%M-%S", gmtime())
```

```
In [ ]: # Creating Feature Group
        neighborhood_feature_group = FeatureGroup(name=neighborhood_feature_group_name, sagemaker_session=feature_store_sess
        encoded_feature_group = FeatureGroup(name=encoded_feature_group_name, sagemaker_session=feature_store_session)
```

```
In [ ]: # Make all objects intro string
        convert_to_strings(new_df)
convert_to_strings(encode_df)
```

Setup Record Identifier and Event Time Features

In [ ]:

```
# Creating record identifier + time features
primary_key_identifier = 'nbh_pol'
secondary_key_identifier = 'nbh_pol_new' # pol_loc
event_time_identifier = 'event_time'

# Tack on the event time to the two df
new_df[event_time_identifier] = pd.Series([time_now]*len(new_df), dtype='float64')
encode_df[event_time_identifier] = pd.Series([time_now]*len(encode_df), dtype='float64')
```

In [ ]:

```
new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1306 entries, 0 to 1305
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   nbh_pol            1306 non-null   object 
 1   med_hse_val        1306 non-null   float64
 2   hse_med_age        1306 non-null   float64
 3   tot_house          1306 non-null   float64
 4   tot_bed            1300 non-null   float64
 5   loc_pol            1306 non-null   float64
 6   avg_house          1293 non-null   float64
 7   avg_bed            1292 non-null   float64
 8   avg_bed_per_house  1292 non-null   float64
 9   ls_1_ocn           1306 non-null   float64
 10  inland             1306 non-null   float64
 11  nr_bay             1306 non-null   float64
 12  nr_ocn             1306 non-null   float64
 13  bed_per_hse        1300 non-null   float64
 14  event_time         1306 non-null   float64
dtypes: float64(14), object(1)
memory usage: 153.2+ KB
```

In [ ]:

```
encode_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   nbh_pol_new     205 non-null    object 
 1   nbh_pol_encode  205 non-null    float64
 2   event_time      205 non-null    float64
dtypes: float64(2), object(1)
memory usage: 4.9+ KB
```

Load Feature Defintions

In [ ]:

```
neighborhood_feature_group.load_feature_definitions(data_frame=new_df)
```

Out[ ]:

```
[FeatureDefinition(feature_name='nbh_pol', feature_type=<FeatureTypeEnum.STRING: 'String'>, collection_type=None),
 FeatureDefinition(feature_name='med_hse_val', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_t
ype=None),
 FeatureDefinition(feature_name='hse_med_age', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_t
ype=None),
 FeatureDefinition(feature_name='tot_house', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_typ
e=None),
 FeatureDefinition(feature_name='tot_bed', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=
None),
 FeatureDefinition(feature_name='loc_pol', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=
None),
 FeatureDefinition(feature_name='avg_house', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_typ
e=None),
 FeatureDefinition(feature_name='avg_bed', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=
None),
 FeatureDefinition(feature_name='avg_bed_per_house', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collec
tion_type=None),
 FeatureDefinition(feature_name='ls_1_ocn', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type
=None),
 FeatureDefinition(feature_name='inland', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=N
one),
 FeatureDefinition(feature_name='nr_bay', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=N
one),
 FeatureDefinition(feature_name='nr_ocn', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_type=N
one),
 FeatureDefinition(feature_name='bed_per_hse', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_t
ype=None),
 FeatureDefinition(feature_name='event_time', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_ty
pe=None)]
```

In [ ]:

```
encoded_feature_group.load_feature_definitions(data_frame=encode_df)
```

Out[ ]:

```
[FeatureDefinition(feature_name='nbh_pol_new', feature_type=<FeatureTypeEnum.STRING: 'String'>, collection_type=Non
e),
 FeatureDefinition(feature_name='nbh_pol_encode', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collectio
n_type=None),
 FeatureDefinition(feature_name='event_time', feature_type=<FeatureTypeEnum.FRACTIONAL: 'Fractional'>, collection_ty
pe=None)]
```

Create a Feature Group

```
In [ ]: neighborhood_feature_group.create(
            s3_uri=f"s3://{default_s3_bucket_name}/{prefix}",
            record_identifier_name=primary_key_identifier,
            event_time_feature_name=event_time_identifier,
            role_arn=role,
            enable_online_store=True,
        )

        encoded_feature_group.create(
            s3_uri=f"s3://{default_s3_bucket_name}/{prefix}",
            record_identifier_name=secondary_key_identifier,
            event_time_feature_name=event_time_identifier,
            role_arn=role,
            enable_online_store=True,
        )

        wait_for_feature_group_creation_complete(feature_group=neighborhood_feature_group)
        wait_for_feature_group_creation_complete(feature_group=encoded_feature_group)
```

```
Waiting for Feature Group Creation
Waiting for Feature Group Creation
Waiting for Feature Group Creation
FeatureGroup neighborhood-feature-group-24-20-56-29 successfully created.
Waiting for Feature Group Creation
FeatureGroup encoded-feature-group-24-20-56-29 successfully created.
```

Checking Results + Putting Records In

```
In [ ]: neighborhood_feature_group.describe()
```

```
Out[ ]: {'FeatureGroupArn': 'arn:aws:sagemaker:us-east-1:004608622582:feature-group/neighborhood-feature-group-24-20-56-29',
         'FeatureGroupName': 'neighborhood-feature-group-24-20-56-29',
         'RecordIdentifierFeatureName': 'nbh_pol',
         'EventTimeFeatureName': 'event_time',
         'FeatureDefinitions': [{'FeatureName': 'nbh_pol', 'FeatureType': 'String'},
          {'FeatureName': 'med_hse_val', 'FeatureType': 'Fractional'},
          {'FeatureName': 'hse_med_age', 'FeatureType': 'Fractional'},
          {'FeatureName': 'tot_house', 'FeatureType': 'Fractional'},
          {'FeatureName': 'tot_bed', 'FeatureType': 'Fractional'},
          {'FeatureName': 'loc_pol', 'FeatureType': 'Fractional'},
          {'FeatureName': 'avg_house', 'FeatureType': 'Fractional'},
          {'FeatureName': 'avg_bed', 'FeatureType': 'Fractional'},
          {'FeatureName': 'avg_bed_per_house', 'FeatureType': 'Fractional'},
          {'FeatureName': 'ls_1_ocn', 'FeatureType': 'Fractional'},
          {'FeatureName': 'inland', 'FeatureType': 'Fractional'},
          {'FeatureName': 'nr_bay', 'FeatureType': 'Fractional'},
          {'FeatureName': 'nr_ocn', 'FeatureType': 'Fractional'},
          {'FeatureName': 'bed_per_hse', 'FeatureType': 'Fractional'},
          {'FeatureName': 'event_time', 'FeatureType': 'Fractional'}],
         'CreationTime': datetime.datetime(2024, 5, 24, 20, 56, 30, 717000, tzinfo=tzlocal()),
         'OnlineStoreConfig': {'EnableOnlineStore': True},
         'OfflineStoreConfig': {'S3StorageConfig': {'S3Uri': 's3://sagemaker-us-east-1-004608622582/sagemaker-featurestore-d
        emo',
           'ResolvedOutputS3Uri': 's3://sagemaker-us-east-1-004608622582/sagemaker-featurestore-demo/004608622582/sagemaker/
        us-east-1/offline-store/neighborhood-feature-group-24-20-56-29-1716584190/data'},
          'DisableGlueTableCreation': False,
          'DataCatalogConfig': {'TableName': 'neighborhood_feature_group_24_20_56_29_1716584190',
           'Catalog': 'AwsDataCatalog',
           'Database': 'sagemaker_featurestore'}},
         'ThroughputConfig': {'ThroughputMode': 'OnDemand'},
         'RoleArn': 'arn:aws:iam::004608622582:role/LabRole',
         'FeatureGroupStatus': 'Created',
         'OnlineStoreTotalSizeBytes': 0,
         'ResponseMetadata': {'RequestId': '5f0a3db2-6997-41ce-be82-7f490aad88f3',
          'HTTPStatusCode': 200,
          'HTTPHeaders': {'x-amzn-requestid': '5f0a3db2-6997-41ce-be82-7f490aad88f3',
           'content-type': 'application/x-amz-json-1.1',
           'content-length': '2248',
           'date': 'Fri, 24 May 2024 20:56:53 GMT'},
          'RetryAttempts': 0}}
```

In [ ]:
```
encoded_feature_group.describe()
```

Out[ ]:
```
{'FeatureGroupArn': 'arn:aws:sagemaker:us-east-1:004608622582:feature-group/encoded-feature-group-24-20-56-29',
 'FeatureGroupName': 'encoded-feature-group-24-20-56-29',
 'RecordIdentifierFeatureName': 'nbh_pol_new',
 'EventTimeFeatureName': 'event_time',
 'FeatureDefinitions': [{'FeatureName': 'nbh_pol_new',
   'FeatureType': 'String'},
  {'FeatureName': 'nbh_pol_encode', 'FeatureType': 'Fractional'},
  {'FeatureName': 'event_time', 'FeatureType': 'Fractional'}],
 'CreationTime': datetime.datetime(2024, 5, 24, 20, 56, 32, 482000, tzinfo=tzlocal()),
 'OnlineStoreConfig': {'EnableOnlineStore': True},
 'OfflineStoreConfig': {'S3StorageConfig': {'S3Uri': 's3://sagemaker-us-east-1-004608622582/sagemaker-featurestore-d
emo',
   'ResolvedOutputS3Uri': 's3://sagemaker-us-east-1-004608622582/sagemaker-featurestore-demo/004608622582/sagemaker/
us-east-1/offline-store/encoded-feature-group-24-20-56-29-1716584192/data'},
  'DisableGlueTableCreation': False,
  'DataCatalogConfig': {'TableName': 'encoded_feature_group_24_20_56_29_1716584192',
   'Catalog': 'AwsDataCatalog',
   'Database': 'sagemaker_featurestore'}},
 'ThroughputConfig': {'ThroughputMode': 'OnDemand'},
 'RoleArn': 'arn:aws:iam::004608622582:role/LabRole',
 'FeatureGroupStatus': 'Created',
 'OnlineStoreTotalSizeBytes': 0,
 'ResponseMetadata': {'RequestId': '62b190da-bd03-4690-800f-a1767f7729e3',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '62b190da-bd03-4690-800f-a1767f7729e3',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '1583',
   'date': 'Fri, 24 May 2024 20:57:01 GMT'},
  'RetryAttempts': 0}}
```

In [ ]:
```
neighborhood_feature_group.ingest(data_frame=new_df, max_workers=3, wait=True)
```

Out[ ]:

```
IngestionManagerPandas(feature_group_name='neighborhood-feature-group-24-20-56-29', feature_definitions={'nbh_pol':
{'FeatureName': 'nbh_pol', 'FeatureType': 'String'}, 'med_hse_val': {'FeatureName': 'med_hse_val', 'FeatureType': 'F
ractional'}, 'hse_med_age': {'FeatureName': 'hse_med_age', 'FeatureType': 'Fractional'}, 'tot_house': {'FeatureNam
e': 'tot_house', 'FeatureType': 'Fractional'}, 'tot_bed': {'FeatureName': 'tot_bed', 'FeatureType': 'Fractional'},
'loc_pol': {'FeatureName': 'loc_pol', 'FeatureType': 'Fractional'}, 'avg_house': {'FeatureName': 'avg_house', 'Featu
reType': 'Fractional'}, 'avg_bed': {'FeatureName': 'avg_bed', 'FeatureType': 'Fractional'}, 'avg_bed_per_house': {'F
eatureName': 'avg_bed_per_house', 'FeatureType': 'Fractional'}, 'ls_1_ocn': {'FeatureName': 'ls_1_ocn', 'FeatureTyp
e': 'Fractional'}, 'inland': {'FeatureName': 'inland', 'FeatureType': 'Fractional'}, 'nr_bay': {'FeatureName': 'nr_b
ay', 'FeatureType': 'Fractional'}, 'nr_ocn': {'FeatureName': 'nr_ocn', 'FeatureType': 'Fractional'}, 'bed_per_hse':
{'FeatureName': 'bed_per_hse', 'FeatureType': 'Fractional'}, 'event_time': {'FeatureName': 'event_time', 'FeatureTyp
e': 'Fractional'}}, sagemaker_fs_runtime_client_config=<botocore.config.Config object at 0x7f568d93b100>, sagemaker_
session=<sagemaker.session.Session object at 0x7f568d9d4730>, max_workers=3, max_processes=1, profile_name=None, _as
ync_result=<multiprocess.pool.MapResult object at 0x7f568d9661a0>, _processing_pool=<pool ProcessPool(ncpus=1)>, _fa
iled_indices=[])
```

In [ ]:

```
encoded_feature_group.ingest(data_frame=encode_df, max_workers=5, wait=True)
```

Out[ ]:

```
IngestionManagerPandas(feature_group_name='encoded-feature-group-24-20-56-29', feature_definitions={'nbh_pol_new':
{'FeatureName': 'nbh_pol_new', 'FeatureType': 'String'}, 'nbh_pol_encode': {'FeatureName': 'nbh_pol_encode', 'Featur
eType': 'Fractional'}, 'event_time': {'FeatureName': 'event_time', 'FeatureType': 'Fractional'}}, sagemaker_fs_runti
me_client_config=<botocore.config.Config object at 0x7f568d93b100>, sagemaker_session=<sagemaker.session.Session obj
ect at 0x7f568d9d4730>, max_workers=5, max_processes=1, profile_name=None, _async_result=<multiprocess.pool.MapResul
t object at 0x7f568d9663e0>, _processing_pool=<pool ProcessPool(ncpus=1)>, _failed_indices=[])
```

In [ ]:

```
### Grabbing the Record from the online store
record_identifier_value = 'Brooktree'

featurestore_runtime.get_record(
    FeatureGroupName=neighborhood_feature_group_name,
    RecordIdentifierValueAsString=record_identifier_value,
)
```

Out[ ]:

```
{'ResponseMetadata': {'RequestId': '524106e0-3df5-48df-bfe4-0b5ca3f12fc3',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '524106e0-3df5-48df-bfe4-0b5ca3f12fc3',
   'content-type': 'application/json',
   'content-length': '1054',
   'date': 'Fri, 24 May 2024 21:01:11 GMT'},
  'RetryAttempts': 0},
 'Record': [{'FeatureName': 'nbh_pol', 'ValueAsString': 'Brooktree'},
  {'FeatureName': 'med_hse_val', 'ValueAsString': '257400.0'},
  {'FeatureName': 'hse_med_age', 'ValueAsString': '9.0'},
  {'FeatureName': 'tot_house', 'ValueAsString': '1438.0'},
  {'FeatureName': 'loc_pol', 'ValueAsString': '182.0'},
  {'FeatureName': 'avg_house', 'ValueAsString': '532.5061475409836'},
  {'FeatureName': 'avg_bed', 'ValueAsString': '548.5381443298969'},
  {'FeatureName': 'avg_bed_per_house', 'ValueAsString': '1.0'},
  {'FeatureName': 'ls_1_ocn', 'ValueAsString': '1.0'},
  {'FeatureName': 'inland', 'ValueAsString': '0.0'},
  {'FeatureName': 'nr_bay', 'ValueAsString': '0.0'},
  {'FeatureName': 'nr_ocn', 'ValueAsString': '0.0'},
  {'FeatureName': 'event_time', 'ValueAsString': '1716584186.0'}]}
```

In [ ]:

```python
record_identifier_value = "Fisherman's Wharf"

featurestore_runtime.get_record(
    FeatureGroupName=neighborhood_feature_group_name,
    RecordIdentifierValueAsString=record_identifier_value,
)
```

Out[ ]:

```
{'ResponseMetadata': {'RequestId': '00c5eaad-81ae-41c5-a99d-c4084c43bded',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '00c5eaad-81ae-41c5-a99d-c4084c43bded',
   'content-type': 'application/json',
   'content-length': '1204',
   'date': 'Fri, 24 May 2024 21:01:13 GMT'},
  'RetryAttempts': 0},
 'Record': [{'FeatureName': 'nbh_pol', 'ValueAsString': "Fisherman's Wharf"},
  {'FeatureName': 'med_hse_val', 'ValueAsString': '500001.0'},
  {'FeatureName': 'hse_med_age', 'ValueAsString': '52.0'},
  {'FeatureName': 'tot_house', 'ValueAsString': '250.0'},
  {'FeatureName': 'tot_bed', 'ValueAsString': '317.0'},
  {'FeatureName': 'loc_pol', 'ValueAsString': '160.0'},
  {'FeatureName': 'avg_house', 'ValueAsString': '501.0'},
  {'FeatureName': 'avg_bed', 'ValueAsString': '535.3848987108655'},
  {'FeatureName': 'avg_bed_per_house', 'ValueAsString': '1.0'},
  {'FeatureName': 'ls_1_ocn', 'ValueAsString': '0.0'},
  {'FeatureName': 'inland', 'ValueAsString': '0.0'},
  {'FeatureName': 'nr_bay', 'ValueAsString': '1.0'},
  {'FeatureName': 'nr_ocn', 'ValueAsString': '0.0'},
  {'FeatureName': 'bed_per_hse', 'ValueAsString': '1.268'},
  {'FeatureName': 'event_time', 'ValueAsString': '1716584186.0'}]}
```

In [ ]:

```python
record_identifier_value = 'Los Osos'

featurestore_runtime.get_record(
    FeatureGroupName=neighborhood_feature_group_name,
    RecordIdentifierValueAsString=record_identifier_value,
)
```

```
Out[ ]: {'ResponseMetadata': {'RequestId': '010e5a93-7e39-4eca-95e4-337f31c914aa',
          'HTTPStatusCode': 200,
          'HTTPHeaders': {'x-amzn-requestid': '010e5a93-7e39-4eca-95e4-337f31c914aa',
           'content-type': 'application/json',
           'content-length': '1200',
           'date': 'Fri, 24 May 2024 21:01:15 GMT'},
          'RetryAttempts': 0},
         'Record': [{'FeatureName': 'nbh_pol', 'ValueAsString': 'Los Osos'},
          {'FeatureName': 'med_hse_val', 'ValueAsString': '221612.5'},
          {'FeatureName': 'hse_med_age', 'ValueAsString': '15.375'},
          {'FeatureName': 'tot_house', 'ValueAsString': '611.75'},
          {'FeatureName': 'tot_bed', 'ValueAsString': '642.5'},
          {'FeatureName': 'loc_pol', 'ValueAsString': '163.0'},
          {'FeatureName': 'avg_house', 'ValueAsString': '611.75'},
          {'FeatureName': 'avg_bed', 'ValueAsString': '642.5'},
          {'FeatureName': 'avg_bed_per_house', 'ValueAsString': '1.0'},
          {'FeatureName': 'ls_1_ocn', 'ValueAsString': '0.0'},
          {'FeatureName': 'inland', 'ValueAsString': '0.0'},
          {'FeatureName': 'nr_bay', 'ValueAsString': '0.0'},
          {'FeatureName': 'nr_ocn', 'ValueAsString': '1.0'},
          {'FeatureName': 'bed_per_hse', 'ValueAsString': '1.0502656313853698'},
          {'FeatureName': 'event_time', 'ValueAsString': '1716584186.0'}]}
```

```
In [ ]: a
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[45], line 1
----> 1 a

NameError: name 'a' is not defined
```

```
In [ ]: %%html

        <p><b>Shutting down your kernel for this notebook to release resources.</b></p>
        <button class="sm-command-button" data-commandlinker-command="kernelmenu:shutdown" style="display:none;">Shutdown Ker

        <script>
        try {
            els = document.getElementsByClassName("sm-command-button");
            els[0].click();
        }
```

```
catch(err) {
    // NoOp
}
</script>
```