House price in recent five years (2013 - 2017) in Fremont, California

Xinmiao Tan

Objective

1. Time series: House price change in recent five years (2013-2017) in Fremont, California.
2. Geographic: House price geographical distribution in Fremont, California

Methods

1. Crawl date from Redfin (last sold price, last sold date, geo-position, square feet, etc.)
2. Clean data
3. Data analysis: Time series & Spatial

Part 1: Crawl Data from Redfin

Main package used: requests, BeautifulSoup

1. Start from a selected URL.

2. Download and save URL content. (Package used: requests)

```
#download redfin content, return content
def download_content(url):
    headers = requests.utils.default_headers()
    headers.update({'User-Agent': 'Mozilla/5.0',})
    response = requests.get(url, headers=headers)
    content = response.text.encode("utf8")
    return content
```

3. Extract useful data from content downloaded and save metadata as a dictionary in local.

   a. Package used: BeautifulSoup.

   b. Data Extracted: geo_position, last sold price, last sold date, sqft, etc.

Part 1 output Example:


```
{"geo_position": "37.5137352;-121.9723055", "geo_region": "US-CA", "state": "CA", "city": "Fremont", "zipcode":
94538, "beds": 3.0, "baths": 2.0, "sqft": 1204, "url": "https://www.redfin.com/CA/Fremont/5756-Antone-Rd-94538/
home/1535962", "title": "5756 ANTONE Rd, Fremont, CA 94538 - 3 beds/2 baths", "description": "3 bed, 2 bath,
1204 sq. ft. house located at 5756 ANTONE Rd, Fremont, CA 94538 sold for $760,000 on Apr 20, 2016. MLS#
40731726. Great home located in a quiet neighborhood! Spacious living room, fa...", "street_address": "5756
Antone Rd", "neighbours": ["/CA/Fremont/42849-Everglades-Park-Dr-94538/home/546318", "/CA/Fremont/42907-
Everglades-Park-Dr-94538/home/761966", "/CA/Fremont/42857-Everglades-Park-Dr-94538/home/1113899", "/CA/Fremont/
5716-Antone-Rd-94538/home/1124946", "/CA/Fremont/5765-Antone-Rd-94538/home/1264703", "/CA/Fremont/5756-Butano-
Park-Dr-94538/home/1394311", "/CA/Fremont/5735-Antone-Rd-94538/home/1475938", "/CA/Fremont/42841-Everglades-
Park-Dr-94538/home/1691213", "/CA/Fremont/42881-Everglades-Park-Dr-94538/home/1737933", "/CA/Fremont/5740-
Antone-Rd-94538/home/1879190", "/CA/Fremont/5764-Butano-Park-Dr-94538/home/589141", "/CA/Fremont/5749-Antone-
Rd-94538/home/801697", "/CA/Fremont/42873-Everglades-Park-Dr-94538/home/1113920", "/CA/Fremont/5741-Antone-
Rd-94538/home/1124960", "/CA/Fremont/42923-Everglades-Park-Dr-94538/home/1336680", "/CA/Fremont/Undisclosed-
address-94538/home/1428181", "/CA/Fremont/5772-Antone-Rd-94538/home/1572983", "/CA/Fremont/5748-Antone-
Rd-94538/home/1695829", "/CA/Fremont/5764-Antone-Rd-94538/home/1851836", "/CA/Fremont/5724-Antone-Rd-94538/
home/1922502", "/CA/Fremont/5748-Butano-Park-Dr-94538/home/655918", "/CA/Fremont/5085-Banff-Park-Ct-94538/home/
829698", "/CA/Fremont/5732-Antone-Rd-94538/home/1123473", "/CA/Fremont/5780-Butano-Park-Dr-94538/home/1124968",
"/CA/Fremont/5757-Antone-Rd-94538/home/1358394", "/CA/Fremont/5725-Antone-Rd-94538/home/1432831", "/CA/Fremont/
5780-Antone-Rd-94538/home/1602983", "/CA/Fremont/42865-Everglades-Park-Dr-94538/home/1698069", "/CA/Fremont/
5788-Antone-Rd-94538/home/1869036", "/CA/Fremont/42915-Everglades-Park-Dr-94538/home/2028730"], "sale_price":
760000.0, "sale_date": "2016-04-26", "estimate_price": 1047482, "style": "Single Family Residential",
"year_built": "1963", "key": "*CA*Fremont*5756-Antone-Rd-94538*home*1535962"}
```

4. Extract nearby homes using the [neighbors] in the metadata.

```
#start
record_list = load_recorded()
if record_list[-1] == ".DS_Store":
    record_list.remove(".DS_Store")
current_queue = deque()
metadata_map = {}
for item in record_list:
     if item != '.DS_Store':
         metadata_map[item] = read_from_file(item)

if len(record_list) == 0:
    content = download_content(SEED_URL)
    metadata = parse_content(content)
    key = metadata["key"]
    persist_to_file(key, content, metadata)
    current_queue.extend([neighbour for neighbour in metadata["neighbours"] if neighbour not in metadata_map and metadata["city"] == "Fremont"])
    metadata_map[key] = metadata
    print("Start processing")
else:
    i = 0
    while i < len(record_list):
        key = record_list[i]
        metadata = read_from_file(key)
        current_queue.extend([neighbour for neighbour in metadata["neighbours"] if neighbour not in metadata_map and metadata["city"] == "Fremont
        i = i + 1

#current_queue.extendleft([INSERT_URL])
while len(current_queue) >= 1:
    current_key = current_queue.popleft()
    if current_key in metadata_map:
        continue
    content = download_content(get_url_from_key(current_key))
    metadata = parse_content(content)
    persist_to_file(current_key, content, metadata)
    metadata_map[current_key] = metadata
    current_queue.extendleft([neighbour for neighbour in metadata["neighbours"] if neighbour not in metadata_map and metadata["city"] == "Fremont
    delay = random.randint(DELAY_RANGE[0], DELAY_RANGE[1])
    print("Finished processing %s , waitting for %d seconds" % (current_key, delay))
    time.sleep(delay)
```

Part 2: Clean Data

1. Exclude records without incomplete information: last sold price, last sold date, geo position, property type.

2. Include records with last sold date between 2013 and 2017.

3. Include records with property type in single family, townhouse and condo

```
metadata_map = {}
metadata_map_for_analysis = {}
#read from metadata
#exclude data without geo_position, last-sold price, last-sold date, property type
for item in record_list:
    #read meta data
    metadata_map[item] = read_from_meta(item)
    if 'sale_price' in metadata_map[item] and \
                    metadata_map[item]['sale_price'] >0 and \
                    'sale_date' in metadata_map[item] and \
                    metadata_map[item]['city'] == 'Fremont' and \
                    metadata_map[item]['sqft'] != '-' and \
                    'zipcode' in metadata_map[item] and \
                    'geo_position' in metadata_map[item] and \
                    metadata_map[item]['sale_date'][:4] in {'2013','2014','2015','2016','2017'} and \
                    metadata_map[item]['style'] in {'Single Family Residential', 'Townhouse', 'Condo/Co-op'}:
        #create metadata_map_for_analysis
        metadata_map_for_analysis[item]=metadata_map[item]
        metadata_map_for_analysis[item]["year"]=metadata_map[item]['sale_date'][:4]
        metadata_map_for_analysis[item]["month"] = metadata_map[item]['sale_date'][5:7]
        metadata_map_for_analysis[item]["year_month"] = metadata_map[item]['sale_date'][:7]
        metadata_map_for_analysis[item]['price_per_sqft'] = metadata_map[item]['sale_price'] / metadata_map[item]['sqft']
        sep = metadata_map_for_analysis[item]['geo_position'].index(';')
        metadata_map_for_analysis[item]['lat'] = float(metadata_map[item]['geo_position'][0:sep])
        metadata_map_for_analysis[item]['lon'] = float(metadata_map[item]['geo_position'][sep+1:])

def save_metadata(dataname, data):
    meta_data_path = "%s/all*%s" % (HOME_DIR, dataname)
    with open(meta_data_path, "w") as f:
        json.dump(data, f)

save_metadata('metadata_map',metadata_map)
save_metadata('metadatamap_for_analysis',metadata_map_for_analysis)
```

Part 2 Output:

Combine all the single meta date into one meta data.

The final output of this step is a dictionary.

The key of this dictionary is the URL link of each house.


```
{"*CA*Fremont*10-Lima-Ter-94539*home*2009670": {"geo_position": "37.546375;-121.9333117", "geo_region": "US-CA",
"state": "CA", "city": "Fremont", "zipcode": 94539, "beds": 3.0, "baths": 2.0, "sqft": 1755, "url": "https://
www.redfin.com/CA/Fremont/10-Lima-Ter-94539/home/2009670", "title": "10 Lima Ter, Fremont, CA - 3 beds/2 baths",
"description": "3 bed, 2 bath, 1755 sq. ft. house located at 10 Lima Ter, Fremont, CA 94539 sold for $993,000 on May
1, 2013. View sales history, tax history, home value estimates, and overhead views. APN 51304610...",
"street_address": "10 Lima Ter", "neighbours": ["/CA/Fremont/60-Mission-Ridge-Ct-94539/home/654389", "/CA/Fremont/63-
Mission-Ridge-Ct-94539/home/1079400", "/CA/Fremont/87-Mission-Ridge-Ct-94539/home/1081398", "/CA/Fremont/71-Lima-
Ter-94539/home/1081422", "/CA/Fremont/30-Lima-Ter-94539/home/1300596", "/CA/Fremont/92-Mission-Cielo-Ave-94539/home/
1475125", "/CA/Fremont/50-Lima-Ter-94539/home/1683960", "/CA/Fremont/40-Lima-Ter-94539/home/1787255", "/CA/Fremont/36-
Mission-Ridge-Ct-94539/home/1816752", "/CA/Fremont/101-Lima-Ter-94539/home/1977678", "/CA/Fremont/70-Lima-Ter-94539/
home/741254", "/CA/Fremont/21-Lima-Ter-94539/home/1079410", "/CA/Fremont/75-Mission-Ridge-Ct-94539/home/1081406", "/
CA/Fremont/Mission-Blvd-94539/home/1090463", "/CA/Fremont/15-Mission-Ridge-Ct-94539/home/1317683", "/CA/Fremont/20-
Lima-Ter-94539/home/1492393", "/CA/Fremont/90-Lima-Ter-94539/home/1737238", "/CA/Fremont/48-Mission-Ridge-Ct-94539/
home/1790577", "/CA/Fremont/51-Mission-Ridge-Ct-94539/home/1829291", "/CA/Fremont/121-Lima-Ter-94539/home/126891955",
"/CA/Fremont/24-Mission-Ridge-Ct-94539/home/1079387", "/CA/Fremont/61-Lima-Ter-94539/home/1079419", "/CA/Fremont/27-
Mission-Ridge-Ct-94539/home/1081414", "/CA/Fremont/31-Lima-Ter-94539/home/1235229", "/CA/Fremont/39-Mission-Ridge-
Ct-94539/home/1466771", "/CA/Fremont/41-Lima-Ter-94539/home/1602361", "/CA/Fremont/81-Lima-Ter-94539/home/1771241", "/
CA/Fremont/12-Mission-Ridge-Ct-94539/home/1795778", "/CA/Fremont/Undisclosed-address-94539/home/1959349", "/CA/
Fremont/122-Lima-Ter-94539/home/126894947"], "sale_price": 993000.0, "sale_date": "2013-05-01", "estimate_price":
1620059, "style": "Single Family Residential", "year_built": "1984", "key": "*CA*Fremont*10-Lima-
Ter-94539*home*2009670", "year": "2013", "month": "05", "year_month": "2013-05", "price_per_sqft": 565.8119658119658,
"lat": 37.546375, "lon": -121.9333117}, "*CA*Fremont*10-Yampa-Way-94539*home*1393462": {"state": "CA", "city":
"Fremont", "zipcode": 94539, "beds": 4.0, "baths": 2.0, "sqft": 1655, "url": "https://www.redfin.com/CA/Fremont/10-
Yampa-Way-94539/home/1393462", "title": "10 Yampa Way, Fremont, CA - 4 beds/2 baths", "description": "4 bed, 2 bath,
1655 sq. ft. house located at 10 Yampa Way, Fremont, CA 94539 sold for $125,000 on Apr 17, 2013. View sales history,
tax history, home value estimates, and overhead views. APN 519166...", "street_address": "10 Yampa Way",
"geo_position": "37.4633607;-121.913465", "neighbours": ["/CA/Fremont/48928-Tonalea-St-94539/home/545775", "/CA/
Fremont/48931-Tonalea-St-94539/home/644094", "/CA/Fremont/100-Yampa-Way-94539/home/1093831", "/CA/Fremont/48921-
Tonalea-St-94539/home/1093846", "/CA/Fremont/40-Yampa-Way-94539/home/1239626", "/CA/Fremont/48915-Yampa-Ct-94539/home/
1638089", "/CA/Fremont/48942-Lady-Fern-Cmn-94539/home/18978103", "/CA/Fremont/48942-Lady-Fern-Cmn-94539/home/
```

Part 3: Data Analysis

Main package used: numpy, spicy.interpolate, pylab, matplotlib.pyplot, mpl_toolkits.mplot3d

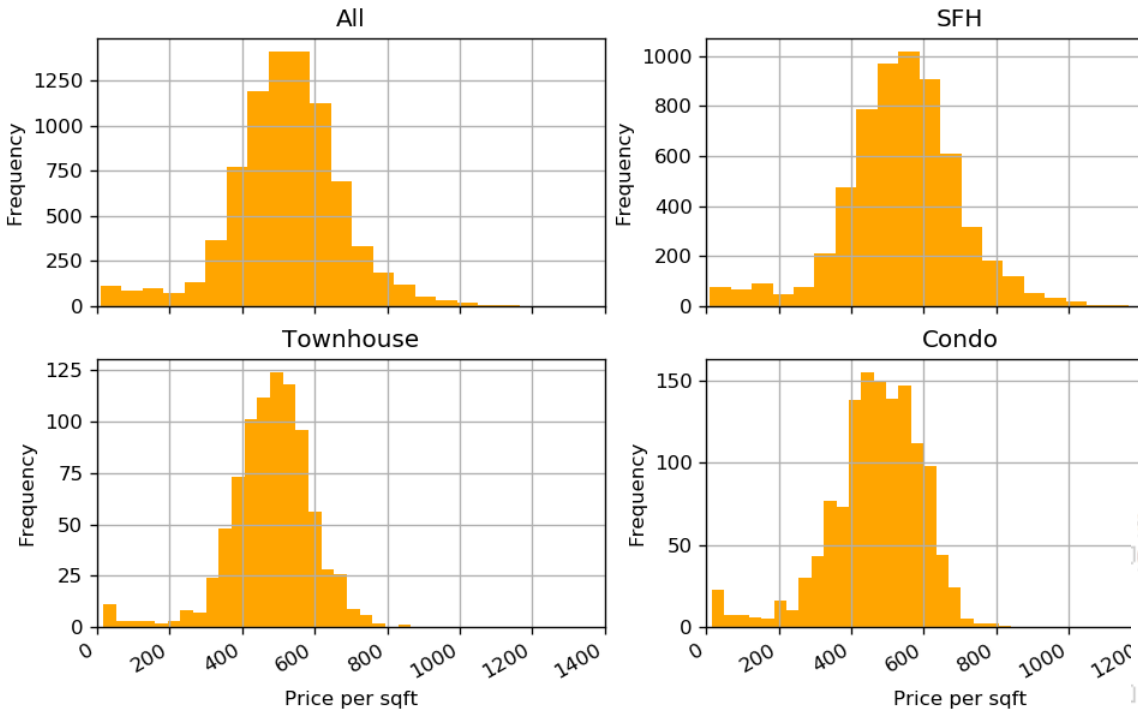
1. Descriptive

Table 3.1.a. Numbers of Records for Different Property Types

	Number of Records	Average Price per Sqft	Median Price per Sqft	Standard Deviation
Single Family	6,063	$537.21	$541.9	161.17
Townhouse	864	$479.24	$486.35	115.53
Condo	1,314	$465.62	$479.37	130.1
All	**8,241**	**$519.72**	**$523.7**	**155.16**

```
metadata_map_for_analysis_sfh ={}
metadata_map_for_analysis_townhouse={}
metadata_map_for_analysis_condo={}

for item in metadata_map_for_analysis:
    if metadata_map_for_analysis[item]['style'] == 'Single Family Residential':
        metadata_map_for_analysis_sfh[item] = metadata_map_for_analysis[item]
    if metadata_map_for_analysis[item]['style'] == 'Townhouse':
        metadata_map_for_analysis_townhouse[item] = metadata_map_for_analysis[item]
    if metadata_map_for_analysis[item]['style'] == 'Condo/Co-op':
        metadata_map_for_analysis_condo[item] = metadata_map_for_analysis[item]

def cal(data,var,cal):
    var_list = []
    for item in data:
        var_list.append(data[item][var])
    return(cal(var_list))
```

Figure 3.1.b. Distribution of Price per square feet for Different Property Types




```
#Histgram: check the distribution of price_per_sqft,
def dist_var(data, var):
    dist_var = []
    for item in data:
        dist_var.append(data[item][var])
    return dist_var

def check_dist(dist_var,style,loc):
    plb.subplot(2,2,loc)
    plb.hist(dist_var,color='orange', bins=24)
    plb.title(style)
    plb.xlabel("Price per sqft")
    plb.xlim([0,1400])
    plb.ylabel("Frequency")
    plb.grid(True)
    plb.show()
    plb.pause(3)
```

2. Time series

How house price change in the recent five years?

Table 3.2.a. Average Price per square feet for Different Property Types from 2013 - 2017

Number of Records	Year 2013 (N)	Year 2013 ($)	Year 2014 (N)	Year 2014 ($)	Year 2015 (N)	Year 2015 ($)	Year 2016 (N)	Year 2016 ($)	Year 2017 (N)	Year 2017 ($)
SFH	1216	$421.16	1182	$488.44	1098	$558.95	1363	$573.1	1204	$641.87
Townhouse	145	$377.42	182	$426.72	169	$487.85	171	$515.02	197	$564.28
Condo	273	$331.5	238	$416.69	241	$496.18	317	$520.98	245	$560.92
All	**1634**	**$402.3**	**1602**	**$470.77**	**1508**	**$540.95**	**1851**	**$558.81**	**1646**	**$620.53**

```
# Table data records counts/price for different property style
def time_serious_data_count(data):
    years = {}
    for item in data:
        year = data[item]['year']
        years[year] = years.get(year, 0) + 1
    return years

def time_serious_price_mean(data):
    price_map = {}
    year = ['2013','2014','2015','2016','2017']
    for yr in year:
        price = []
        for item in data:
            if data[item]['year'] == yr:
                price.append(data[item]['price_per_sqft'])
        price_map[yr]=round(np.mean(price),2)
    return price_map
```

Figure 3.2.b. Average Price per square feet using Different Interpolation from 2013 - 2017

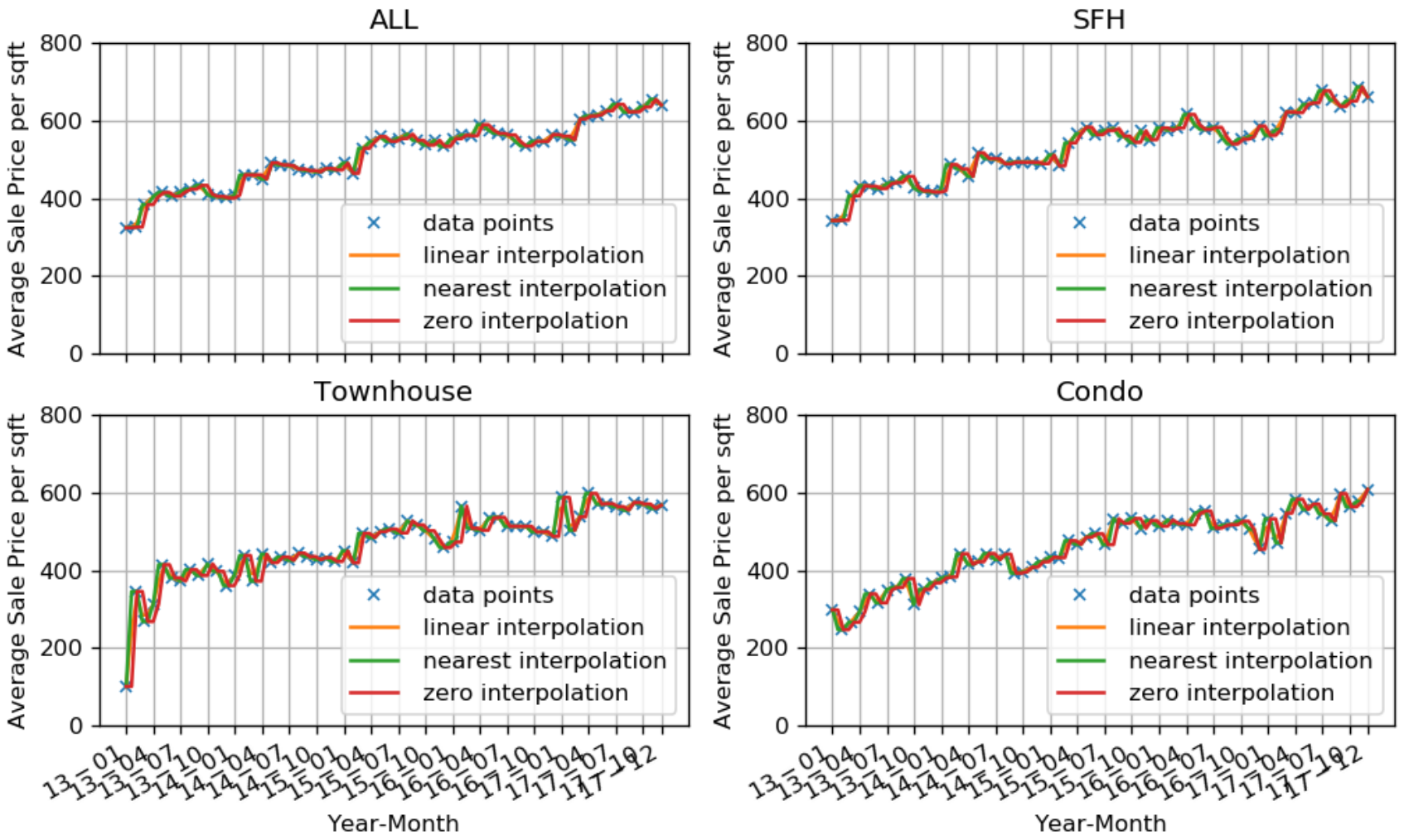
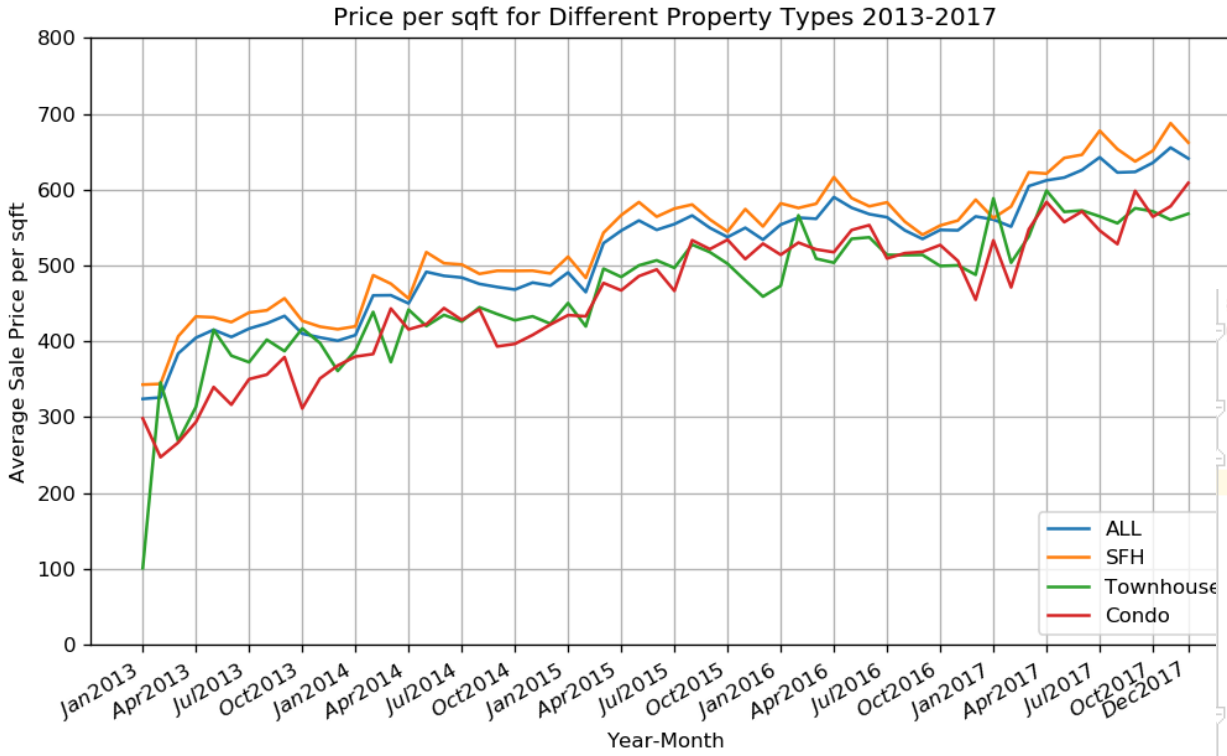


Figure 3.2.c. Average Price per square feet for Different Property Types from 2013 - 2017




```python
# Put all property type into one figure, using data points method
def year_month_dist_for4(data,style):
    x = []
    y = []
    for var in data:
        x.append((int(var[3])-3)*12+int(var[5:]))
        y.append(data[var]['price_per_sqft'])

    plb.plot(x, y, ms=5, label = style)
    plb.title("Price per sqft for Different Property Types 2013-2017")
    plb.xlabel("Year-Month")
    plb.ylabel("Average Sale Price per sqft")
    plb.grid(True)
    plb.legend(loc='lower right')
    plb.ylim([0,800])

    plb.xticks([1, 4, 7, 10,
                13, 16, 19, 22,
                25, 28, 31, 34,
                37, 40, 43, 46,
                49, 52, 55, 58, 60],
               ['$Jan 2013$','$Apr 2013$','$Jul 2013$','$Oct 2013$',
                '$Jan 2014$','$Apr 2014$','$Jul 2014$','$Oct 2014$',
                '$Jan 2015$','$Apr 2015$','$Jul 2015$','$Oct 2015$',
                '$Jan 2016$','$Apr 2016$','$Jul 2016$','$Oct 2016$',
                '$Jan 2017$','$Apr 2017$','$Jul 2017$','$Oct 2017$',
                '$Dec 2017$'])
```

3. Spatial

How house prices differ geographically?

Which Fremont communities increase the most, and which increase the least?

Figure 3.3.a. Fremont Map

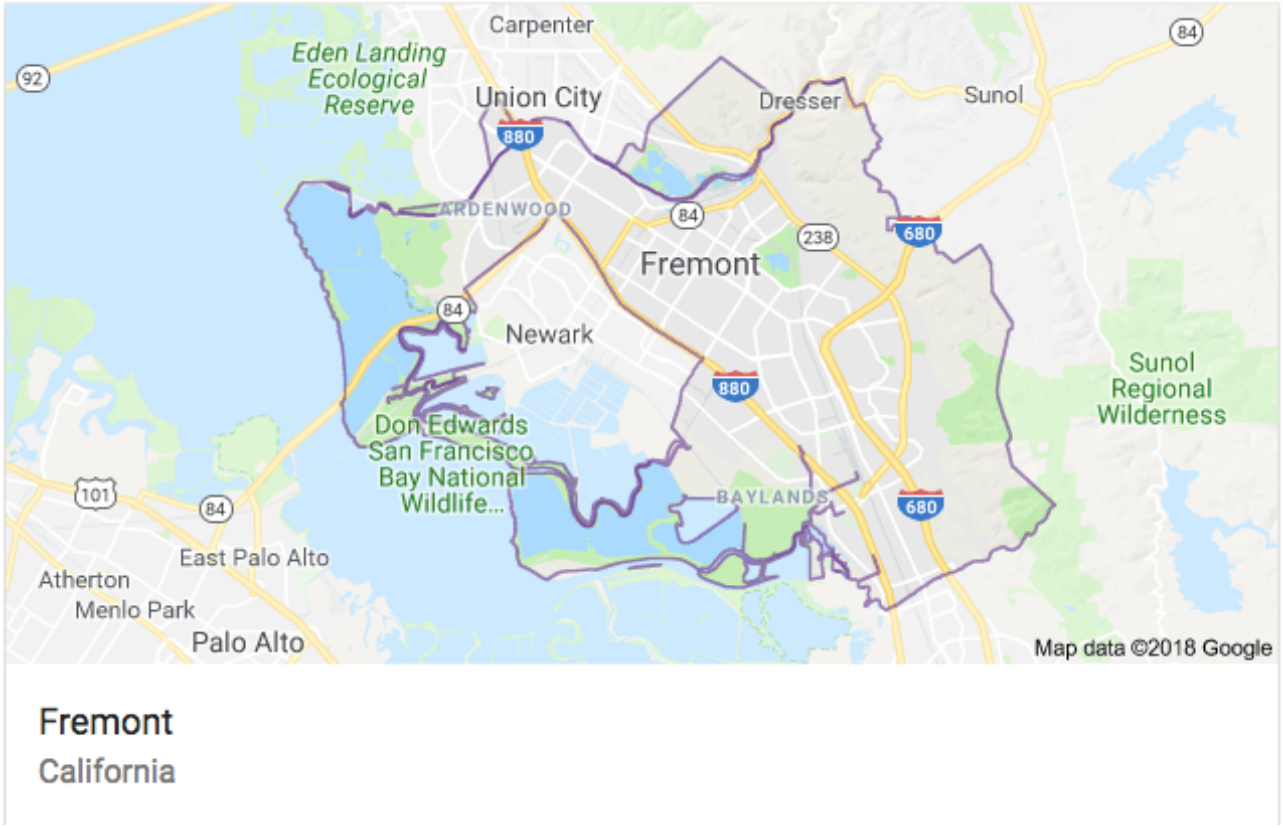
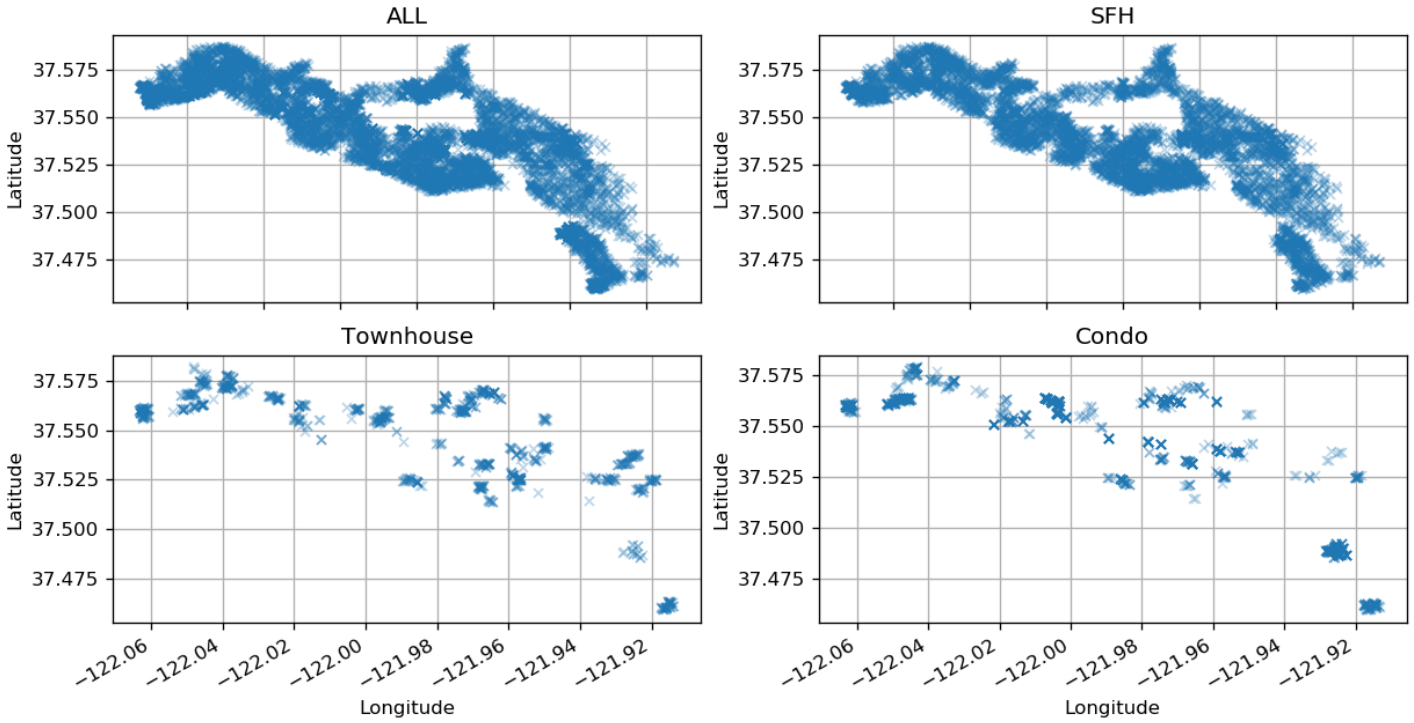
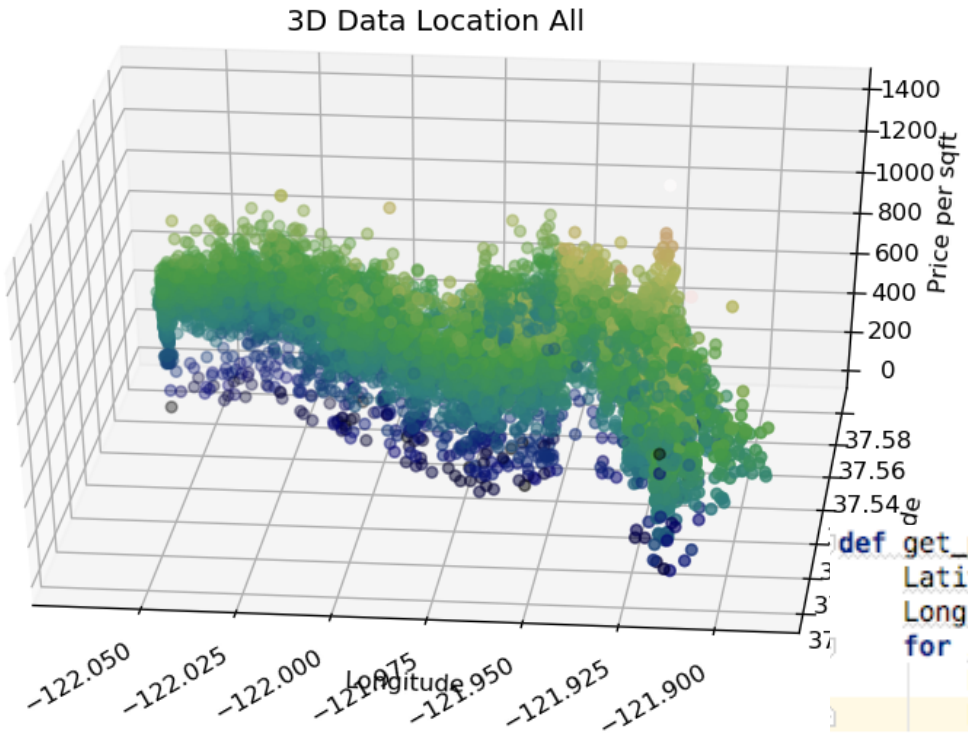


Figure 3.3.b. 2D Data Points Location




```
def get_points2d(data,style,loc):
    Latitude = []
    Longitude = []
    for item in data:
        Latitude.append(data[item]["lat"])
        Longitude.append(data[item]["lon"])

    plb.subplot(2,2,loc)
    plb.plot(Longitude, Latitude, 'x', ms=5, label='data points',
    plb.title(style)
    plb.xlabel('Longitude')
    plb.ylabel('Latitude')
    plb.grid()
    plb.show()
    plb.pause(2)
```

Figure 3.3.c. 3D Price per square feet



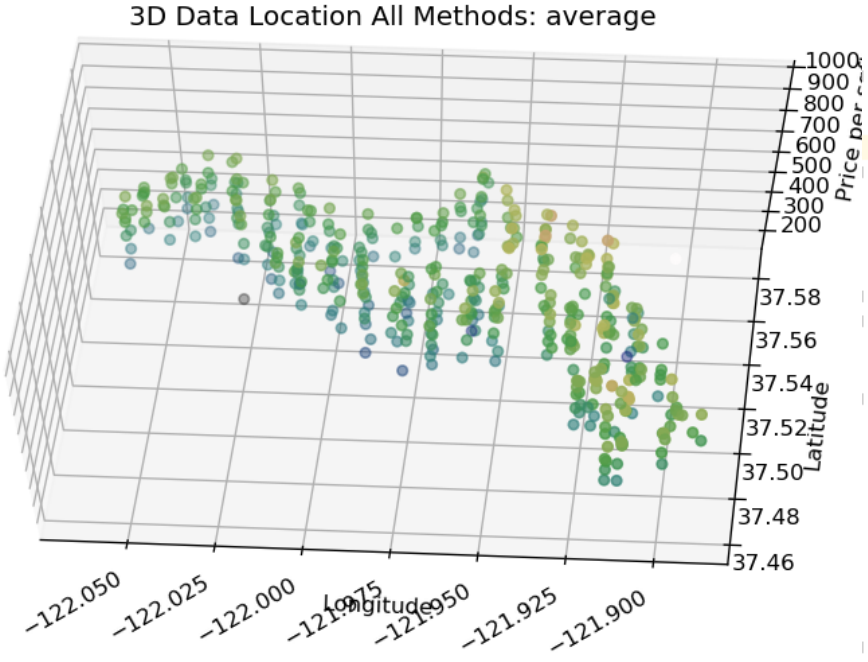

```
def get_points2d(data,style,loc):
    Latitude = []
    Longitude = []
    for item in data:
        Latitude.append(data[item]["lat"])
        Longitude.append(data[item]["lon"])

    plb.subplot(2,2,loc)
    plb.plot(Longitude, Latitude, 'x', ms=5, label='data points', alph
    plb.title(style)
    plb.xlabel('Longitude')
    plb.ylabel('Latitude')
    plb.grid()
    plb.show()
    plb.pause(2)
```

Group houses within 0.01 Longitude and Latitude to reduce data points:

—New Price per square feet is the average price per square from houses from the same group

—New latitude and longitude is the average latitude and longitude from houses from the same group

Figure 3.3.d. 3D Price per square feet




```python
def new_geo_group_func(data):
    #get the lat and lon
    Latitude = []
    Longitude = []
    for item in data:
        Latitude.append(data[item]["lat"])
        Longitude.append(data[item]["lon"])
    #add new lat/lon group
    new_geo_group = {}
    for item in data:
        new_geo_group[item] = data[item]
        new_geo_group[item]["group_lat"] = int((data[item]["lat"]-np.min(Latitude))/0.01)
        new_geo_group[item]["group_lon"] = int((data[item]["lon"] - np.min(Longitude)) / 0.01)
        new_geo_group[item]["group"] = str(data[item]["year"]) + '-' + \
                                       str(int((data[item]["lat"]-np.min(Latitude))/0.01)) + \
                                       "-"+ str(int((data[item]["lon"] - np.min(Longitude)) / 0.01))

    group_map = var_check(new_geo_group, 'group')
    metadata_map_new_geo = {}
    for group in group_map:
        metadata_map_new_geo_prep = {}
        house_price = []
        latitude = []
        longitude = []
        for item in new_geo_group:
            if new_geo_group[item]['group'] == group:
                house_price.append(new_geo_group[item]['price_per_sqft'])
                latitude.append(new_geo_group[item]['lat'])
                longitude.append(new_geo_group[item]['lon'])
        price_mean = np.mean(house_price)
        lat_mean = np.mean(latitude)
        lon_mean = np.mean(longitude)
        metadata_map_new_geo_prep['price_per_sqft'] = price_mean
        metadata_map_new_geo_prep['lat'] = lat_mean
        metadata_map_new_geo_prep['lon'] = lon_mean
        metadata_map_new_geo[group]= metadata_map_new_geo_prep
        metadata_map_new_geo[group]['year']= group[:4]
        metadata_map_new_geo[group]['group'] = group
        metadata_map_new_geo[group]['geo_group'] = group[5:]
    return metadata_map_new_geo
```

Figure 3.3.e. 2D Price per square feet

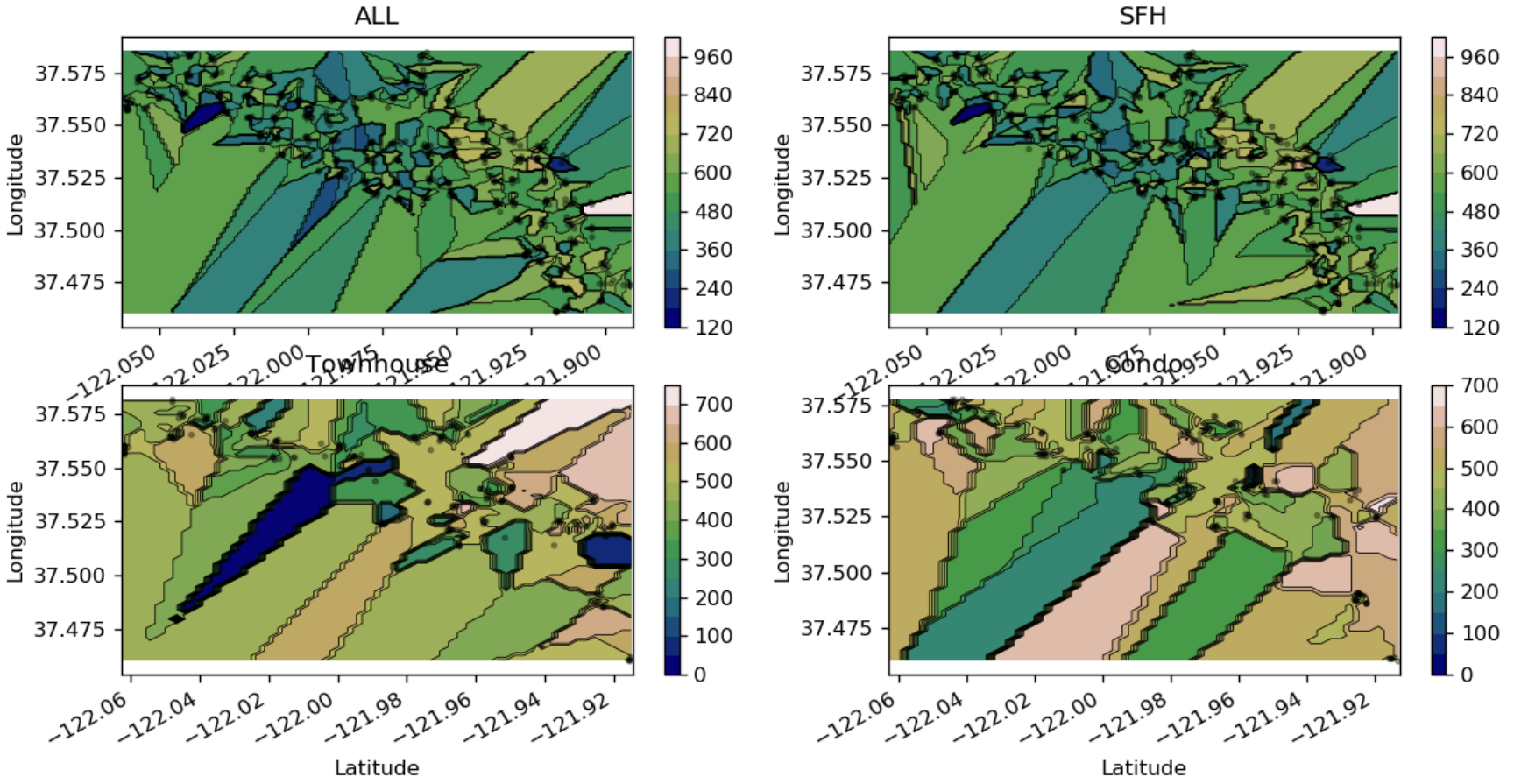
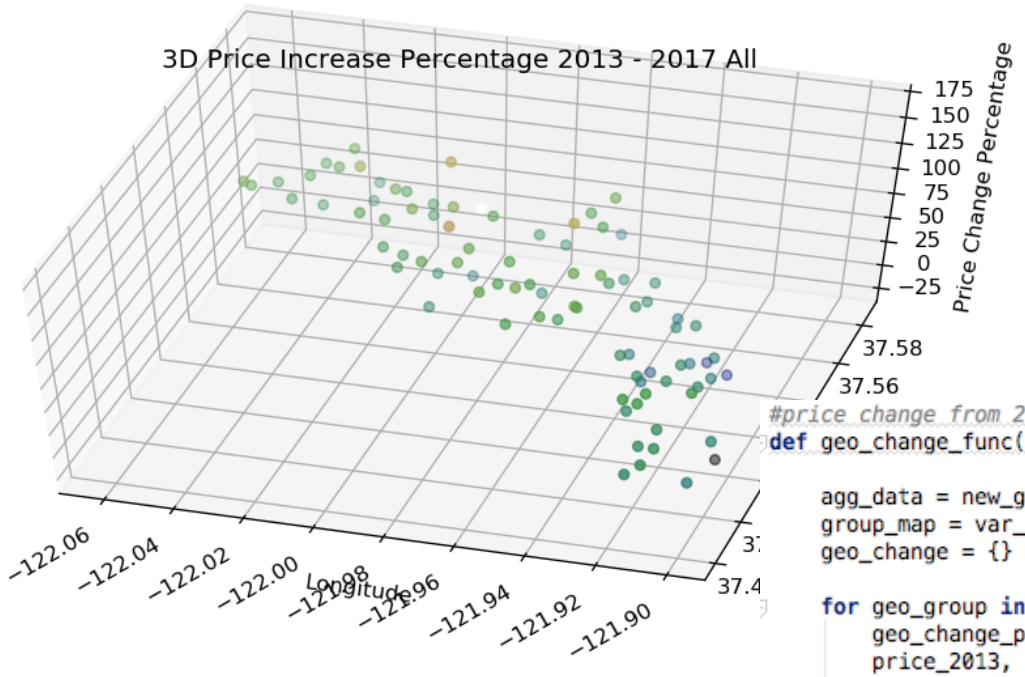
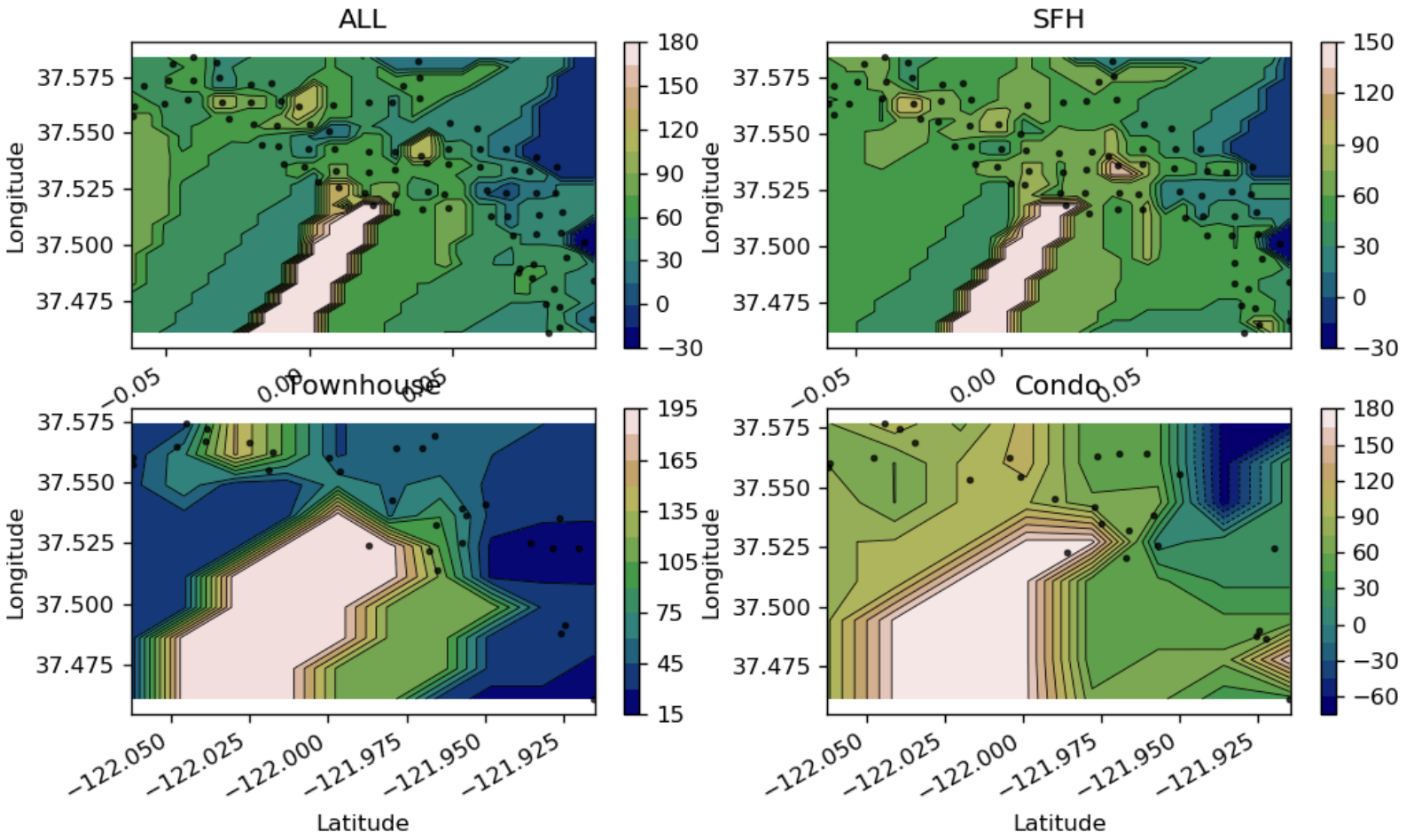


Figure 3.3.f. 3D Price increase percentage from 2013 Jan to 2017 Dec




```python
#price change from 2013 to 2017 for different geo-group
def geo_change_func(data):

    agg_data = new_geo_group_func(data)
    group_map = var_check(agg_data, 'geo_group')
    geo_change = {}

    for geo_group in group_map:
        geo_change_prep = {}
        price_2013, price_2017 = None, None
        for item in agg_data:
            if agg_data[item]['geo_group'] == geo_group and agg_data[item]['year'] in {'2013
                price_2013 = agg_data[item]['price_per_sqft']
                lat_2013 = agg_data[item]['lat']
                lon_2013 = agg_data[item]['lon']
            if agg_data[item]['geo_group'] == geo_group and agg_data[item]['year'] in {'2017
                price_2017 = agg_data[item]['price_per_sqft']
                lat_2017 = agg_data[item]['lat']
                lon_2017 = agg_data[item]['lon']
        if price_2013 is not None and price_2017 is not None:
            price_perc = round((price_2017-price_2013)/price_2013*100,2)
            lat_mean = (lat_2013+lat_2017)/2
            lon_mean = (lon_2013+lon_2017)/2
            geo_change_prep['price_perc'] = price_perc
            geo_change_prep['lat'] = lat_mean
            geo_change_prep['lon'] = lon_mean
            geo_change[geo_group]= geo_change_prep
            geo_change[geo_group]['year'] = '2013-2017'
    return geo_change
```

Figure 3.3.g. 2D Price changed percentage from 2013 Jan to 2017 Dec



Take Home Message:

1. Price increase about 50% in the recent 5 years.
2. Price is relatively low from Oct - Jan, and price is relatively high from April - June
3. South east part of Fremont has higher price (Irvington and Mission San Jose).
4. North west part of Fremont (Ardenwood) has increased the price more significantly in the recent five years, while South east part of Fremont (Irvington and Mission San Jose) has the lowest percentage of increasing.
5. The price for different property types (Single family, townhouse, and condo) behaviors similar to time and geographic factors.