Problem Description

Rents in Munich have increased drastically over the last years. It get's harder for small businesses to find a spot within the city that is a) not covered by a strong competitor already or b) too expensive. This analysis should provide people starting a business with the necessairy insights on where to start.

Data Description

We will use data on rental prices for the respective neighborhoods and and make use of the Foursquare API. The data for rental prices will be scraped from a website and Latitude and Longitude will be added with the help of a geolocation csv file.

Methdology

We will import relevant libraries first, scrape the data from different resources second. We will then do some basic ETL to merge and clean the data from the different datasets. Afterwards exploratory analysis will be performed to get an understanding of the data. We will then derive insights from rental prices, number of inhabitants per area and the number of venues for the respective area. Afterwards we will develop a simple regression model to get an understanding of how the variables impact the number of venues in a certain areas. Areas that are below the predicted number of venues will then be a good starting point to start a business.

Importing libraries and scraping the data

In [17]:

```
#Importing relevant libraries
import bs4 as bs
import urllib.request
import time
from datetime import datetime
import pandas as pd
import json
import requests
import os
import folium
import seaborn as sns
```

In [2]:

```
#Load data
url = 'https://suedbayerische-immobilien.de/Mietpreise-Muenchen-Stadtteile'
df=pd.read_html(url, header=0)[0]
df.columns=['Neighborhood','Rent per M2']
df
```

Out[2]:

	Neighborhood	Rent per M2
0	Altstadt – Lehel	1915
1	Schwabing	1876
2	Maxvorstadt – Universitätsviertel	1861
3	Ludwigsvorstadt – Isarvorstadt	1748
4	Maxvorstadt	1741
5	Schwabing-West	1731
6	Au – Haidhausen	1696
7	Altbogenhausen – Herzogpark	1691
8	Nymphenburg	1647
9	Schwanthalerhöhe	1605
10	Harlaching	1574
11	Neuhausen	1567
12	Sendling	1532
13	Untergiesing	1511
14	Alte Heide – Hirschau	1473
15	Obergiesing	1466
16	Thalkirchen – Obersendling – Forstenried – Für...	1459
17	Oberföhring – Englschalking	1421
18	Milbertshofen	1414
19	Moosach	1410
20	Laim	1407
21	Pasing – Obermenzing	1389
22	Hadern	1382
23	Trudering – Riem	1379
24	Sendling – Westpark	1375
25	Ramersdorf – Perlach	1369
26	Berg am Laim	1362
27	Freimann	1361
28	Johanneskirchen – Daglfing	1358
29	Am Hart	1348
30	Allach – Untermenzing	1285
31	Aubing – Lochhausen – Langwied	1278
32	Feldmoching – Hasenbergl	1219

In [3]:

```
df.info()
print ('---')
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33 entries, 0 to 32
Data columns (total 2 columns):
Neighborhood    33 non-null object
Rent per M2     33 non-null int64
dtypes: int64(1), object(1)
memory usage: 608.0+ bytes
---
```

Out[3]:

	Rent per M2
count	33.000000
mean	1510.606061
std	185.027353
min	1219.000000
25%	1375.000000
50%	1459.000000
75%	1647.000000
max	1915.000000

In [4]:

```
#Get Latitude and Longitude
geo_data = pd.read_excel("Geospatial_Coordinates.xls")
geo_data
df['Latitude']=geo_data['Latitude'].values
df['Longitude']=geo_data['Longitude'].values
df
```

Out[4]:

	Neighborhood	Rent per M2	Latitude	Longitude
0	Altstadt – Lehel	1915	48.139130	11.580220
1	Schwabing	1876	48.167450	11.570380
2	Maxvorstadt – Universitätsviertel	1861	48.147629	11.568050
3	Ludwigsvorstadt – Isarvorstadt	1748	48.132440	11.556090
4	Maxvorstadt	1741	48.151092	11.562418
5	Schwabing-West	1731	48.168271	11.569873
6	Au – Haidhausen	1696	48.128753	11.590536
7	Altbogenhausen – Herzogpark	1691	48.149398	11.603792
8	Nymphenburg	1647	48.158257	11.503297
9	Schwanthalerhöhe	1605	48.134230	11.539034
10	Harlaching	1574	48.088293	11.565078
11	Neuhausen	1567	48.156335	11.531386
12	Sendling	1532	48.118012	11.539083
13	Untergiesing	1511	48.114963	11.570189
14	Alte Heide – Hirschau	1473	48.174773	11.603891
15	Obergiesing	1466	48.111156	11.588909
16	Thalkirchen – Obersendling – Forstenried – Für...	1459	48.097802	11.520807
17	Oberföhring – Englschalking	1421	48.156458	11.642028
18	Milbertshofen	1414	48.182385	11.575043
19	Moosach	1410	48.179895	11.510571
20	Laim	1407	48.139551	11.502166
21	Pasing – Obermenzing	1389	48.147785	11.460701
22	Hadern	1382	48.118064	11.481842
23	Trudering – Riem	1379	48.123175	11.664078
24	Sendling – Westpark	1375	48.117984	11.516719
25	Ramersdorf – Perlach	1369	48.113917	11.615699
26	Berg am Laim	1362	48.123483	11.633451
27	Freimann	1361	48.191969	11.614384
28	Johanneskirchen – Daglfing	1358	48.148570	11.649227
29	Am Hart	1348	48.197245	11.571674
30	Allach – Untermenzing	1285	48.195994	11.457013
31	Aubing – Lochhausen – Langwied	1278	48.158437	11.414066

	Neighborhood	Rent per M2	Latitude	Longitude
32	Feldmoching – Hasenbergl	1219	48.213898	11.539273

In [5]:

```
#Load number of inhabitants per neighborhood
url = 'https://suedbayerische-immobilien.de/Einwohner-Muenchen-Stadtteile'
inhabitants=pd.read_html(url, header=0)[0]
inhabitants.columns=['Neighborhood','No of inhabitants', 'Percentage']
inhabitants['Neighborhood'] = inhabitants['Neighborhood'].str[2:]
inhabitants = inhabitants.drop('Percentage', 1)
inhabitants = inhabitants.drop(25)
inhabitants
```

Out[5]:

	Neighborhood	No of inhabitants
0	Ramersdorf - Perlach	108.244
1	Neuhausen - Nymphenburg	95.906
2	Thalkirchen - Obersendling - Forstenried - Für...	90.790
3	Bogenhausen	82.138
4	Milbertshofen - Am Hart	73.617
5	Pasing - Obermenzing	70.783
6	Schwabing - Freimann	69.676
7	Trudering - Riem	67.009
8	Schwabing West	65.892
9	Au - Haidhausen	59.752
10	Feldmoching - Hasenbergl	59.391
11	Sendling - Westpark	55.405
12	Laim	54.030
13	Untergiesing - Harlaching	51.937
14	Maxvorstadt	51.642
15	Moosach	51.537
16	Obergiesing - Fasanengarten	51.499
17	Ludwigsvorstadt - Isarvorstadt	50.620
18	Hadern	48.945
19	Berg am Laim	43.068
20	Aubing - Lochhausen - Langwied	42.305
21	Sendling	39.953
22	Allach - Untermenzing	30.737
23	Schwanthalerhöhe	29.663
24	Altstadt - Lehel	20.422

It seems like the list contains less values than the one created before. We will compare the dataframes and merge the data two have one combined dataset. Since the names differ too much for merging the dataframes right away, we do some magic within Excel in between an create a CSV-File.

In [10]:

```
#Merge Inhabitants data with with rental prices and coordinates
inhabitants_all_areas = pd.read_excel("Inhabitants_Per_Area.xls")
inhabitants_all_areas
df['Inhabitants']=inhabitants_all_areas['Inhabitants'].values
df
```

Out[10]:

	Neighborhood	Rent per M2	Latitude	Longitude	Inhabitants
0	Altstadt – Lehel	1915	48.139130	11.580220	2042
1	Schwabing	1876	48.167450	11.570380	69676
2	Maxvorstadt – Universitätsviertel	1861	48.147629	11.568050	0
3	Ludwigsvorstadt – Isarvorstadt	1748	48.132440	11.556090	50620
4	Maxvorstadt	1741	48.151092	11.562418	51642
5	Schwabing-West	1731	48.168271	11.569873	65892
6	Au – Haidhausen	1696	48.128753	11.590536	59752
7	Altbogenhausen – Herzogpark	1691	48.149398	11.603792	82138
8	Nymphenburg	1647	48.158257	11.503297	95906
9	Schwanthalerhöhe	1605	48.134230	11.539034	29663
10	Harlaching	1574	48.088293	11.565078	0
11	Neuhausen	1567	48.156335	11.531386	0
12	Sendling	1532	48.118012	11.539083	39953
13	Untergiesing	1511	48.114963	11.570189	51937
14	Alte Heide – Hirschau	1473	48.174773	11.603891	0
15	Obergiesing	1466	48.111156	11.588909	51499
16	Thalkirchen – Obersendling – Forstenried – Für...	1459	48.097802	11.520807	90790
17	Oberföhring – Englschalking	1421	48.156458	11.642028	0
18	Milbertshofen	1414	48.182385	11.575043	73617
19	Moosach	1410	48.179895	11.510571	51537
20	Laim	1407	48.139551	11.502166	54030
21	Pasing – Obermenzing	1389	48.147785	11.460701	70783
22	Hadern	1382	48.118064	11.481842	48945
23	Trudering – Riem	1379	48.123175	11.664078	67009
24	Sendling – Westpark	1375	48.117984	11.516719	55405
25	Ramersdorf – Perlach	1369	48.113917	11.615699	108244
26	Berg am Laim	1362	48.123483	11.633451	43068
27	Freimann	1361	48.191969	11.614384	0
28	Johanneskirchen – Daglfing	1358	48.148570	11.649227	0
29	Am Hart	1348	48.197245	11.571674	0
30	Allach – Untermenzing	1285	48.195994	11.457013	30737
31	Aubing – Lochhausen – Langwied	1278	48.158437	11.414066	42305

	Neighborhood	Rent per M2	Latitude	Longitude	Inhabitants
32	Feldmoching – Hasenbergl	1219	48.213898	11.539273	59391

In [14]:

```
#Dropping Rows that are zero
df = df[df.Inhabitants != 0]
df
```

Out[14]:

	Neighborhood	Rent per M2	Latitude	Longitude	Inhabitants
0	Altstadt – Lehel	1915	48.139130	11.580220	2042
1	Schwabing	1876	48.167450	11.570380	69676
3	Ludwigsvorstadt – Isarvorstadt	1748	48.132440	11.556090	50620
4	Maxvorstadt	1741	48.151092	11.562418	51642
5	Schwabing-West	1731	48.168271	11.569873	65892
6	Au – Haidhausen	1696	48.128753	11.590536	59752
7	Altbogenhausen – Herzogpark	1691	48.149398	11.603792	82138
8	Nymphenburg	1647	48.158257	11.503297	95906
9	Schwanthalerhöhe	1605	48.134230	11.539034	29663
12	Sendling	1532	48.118012	11.539083	39953
13	Untergiesing	1511	48.114963	11.570189	51937
15	Obergiesing	1466	48.111156	11.588909	51499
16	Thalkirchen – Obersendling – Forstenried – Für...	1459	48.097802	11.520807	90790
18	Milbertshofen	1414	48.182385	11.575043	73617
19	Moosach	1410	48.179895	11.510571	51537
20	Laim	1407	48.139551	11.502166	54030
21	Pasing – Obermenzing	1389	48.147785	11.460701	70783
22	Hadern	1382	48.118064	11.481842	48945
23	Trudering – Riem	1379	48.123175	11.664078	67009
24	Sendling – Westpark	1375	48.117984	11.516719	55405
25	Ramersdorf – Perlach	1369	48.113917	11.615699	108244
26	Berg am Laim	1362	48.123483	11.633451	43068
30	Allach – Untermenzing	1285	48.195994	11.457013	30737
31	Aubing – Lochhausen – Langwied	1278	48.158437	11.414066	42305
32	Feldmoching – Hasenbergl	1219	48.213898	11.539273	59391

The Data is now ready for Analysis

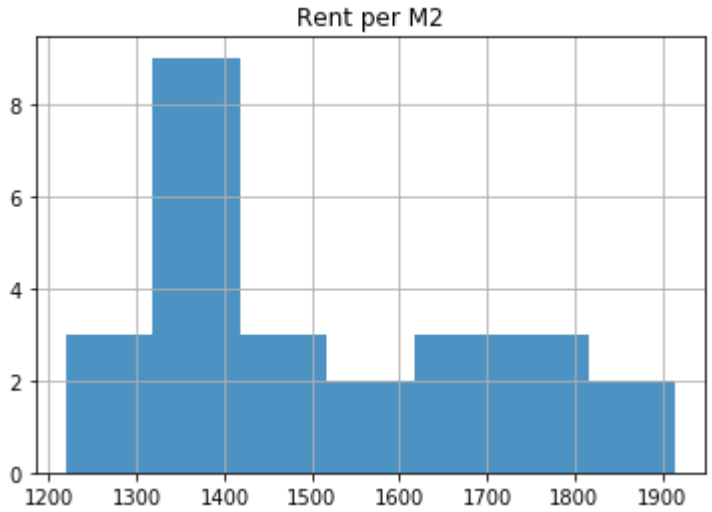
First Analysis of the Data

In [16]:

```
hist = df.hist(column='Rent per M2', bins=7, alpha = 0.8)
hist
```

Out[16]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x123c6a400
>]],
      dtype=object)
```



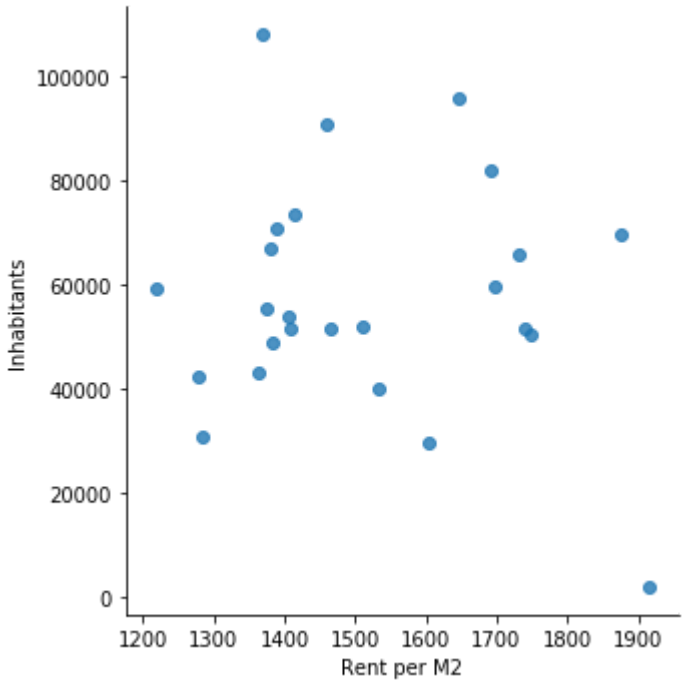
We can see a concentration of rental prices around 13.00-14.50 EUR, with 12 of 33 neighborhood in that bin

In [20]:

```
#How do rental prices and inhabitants per area correlate?
sns.lmplot('Rent per M2', 'Inhabitants', data=df, fit_reg=False)
```

Out[20]:

```
<seaborn.axisgrid.FacetGrid at 0x1a25bd7128>
```



Seems like there's no obvious relationship between the # of inhabitants and the Rent per M2

In [22]:

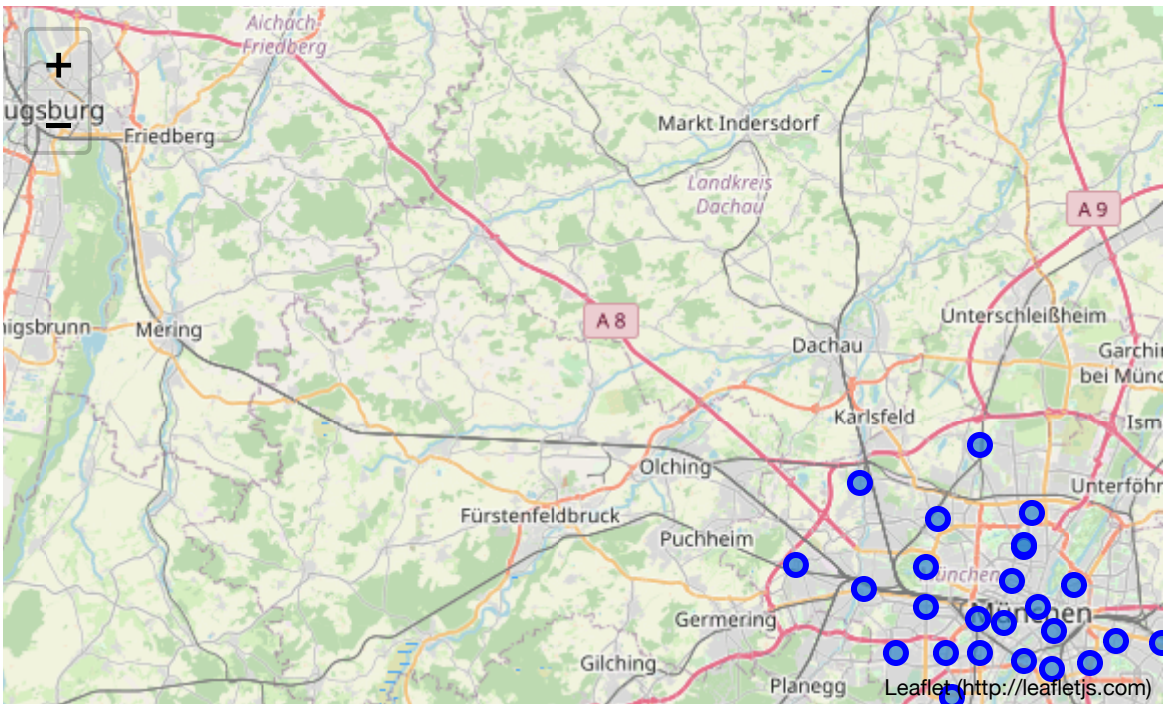
```
#Creating a map to get a visual overview of where the areas are
center_lat=df.Latitude.mean()
center_long=df.Longitude.mean()

# create map of Munich using latitude and longitude values
map_to = folium.Map(location=[center_lat, center_long], zoom_start=10)

# add markers to map
for lat, lng, neighborhood in zip(df['Latitude'], df['Longitude'], df['Neighborh
ood']):
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_to)

map_to
```

Out[22]:



In [34]:

```
#Connecting to the Foursquare API
CLIENT_ID = 'FUZFUOSOAV35IPWPTAYMF2HV3EEX3IQKUQ1JPPA1ZHYL4GMN'
CLIENT_SECRET = 'Y2S3SBIZMUD3V2L4Z5EFKLH2EINWA2SOELDPTKVD0HRFM4O2'
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: FUZFUOSOAV35IPWPTAYMF2HV3EEX3IQKUQ1JPPA1ZHYL4GMN
CLIENT_SECRET:Y2S3SBIZMUD3V2L4Z5EFKLH2EINWA2SOELDPTKVD0HRFM4O2
```

In [56]:

```
def getNearbyVenues(names, latitudes, longitudes):
    radius=500
    LIMIT=100
    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client
_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in
venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [58]:

```
munich_venues = getNearbyVenues(names=df['Neighborhood'],
                                latitudes=df['Latitude'],
                                longitudes=df['Longitude']
                               )
```

```
Altstadt — Lehel
Schwabing
Ludwigsvorstadt — Isarvorstadt
Maxvorstadt
Schwabing-West
Au — Haidhausen
Altbogenhausen — Herzogpark
Nymphenburg
Schwanthalerhöhe
Sendling
Untergiesing
Obergiesing
Thalkirchen — Obersendling — Forstenried — Fürstenried — Solln
Milbertshofen
Moosach
Laim
Pasing — Obermenzing
Hadern
Trudering — Riem
Sendling — Westpark
Ramersdorf — Perlach
Berg am Laim
Allach — Untermenzing
Aubing — Lochhausen — Langwied
Feldmoching — Hasenbergl
```

In [59]:

```
print('There are {} single categories.'.format(len(munich_venues['Venue Categor
y'].unique())))
```

```
There are 162 single categories.
```

In [60]:

```
venue_data_grouped = munich_venues.groupby('Neighborhood').count()
venue_data_grouped
```

Out[60]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Ven Catego
Neighborhood						
Allach – Untermenzing	7	7	7	7	7	7
Altbogenhausen – Herzogpark	20	20	20	20	20	20
Altstadt – Lehel	100	100	100	100	100	100
Au – Haidhausen	42	42	42	42	42	42
Aubing – Lochhausen – Langwied	7	7	7	7	7	7
Berg am Laim	8	8	8	8	8	8
Feldmoching – Hasenbergl	10	10	10	10	10	10
Hadern	9	9	9	9	9	9
Laim	21	21	21	21	21	21
Ludwigsvorstadt – Isarvorstadt	42	42	42	42	42	42
Maxvorstadt	48	48	48	48	48	48
Milbertshofen	18	18	18	18	18	18
Moosach	26	26	26	26	26	26
Nymphenburg	12	12	12	12	12	12
Obergiesing	17	17	17	17	17	17
Pasing – Obermenzing	63	63	63	63	63	63
Ramersdorf – Perlach	14	14	14	14	14	14
Schwabing	25	25	25	25	25	25
Schwabing-West	19	19	19	19	19	19
Schwanthalerhöhe	47	47	47	47	47	47
Sendling	47	47	47	47	47	47
Sendling – Westpark	5	5	5	5	5	5
Thalkirchen – Obersendling – Forstenried – Fürstenried – Solln	15	15	15	15	15	15

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Ven Categ
Neighborhood						
Trudering – Riem	12	12	12	12	12	12
Untergiesing	26	26	26	26	26	26

In [61]:

```
#Let's add the number of venues to our original dataframe
venue_data_grouped = venue_data_grouped.drop(['Neighborhood Latitude', 'Neighbor
hood Longitude','Venue Longitude','Venue Latitude', 'Venue Category'], axis=1)
venue_data_grouped
```

Out[61]:

	Venue
Neighborhood	
Allach – Untermenzing	7
Altbogenhausen – Herzogpark	20
Altstadt – Lehel	100
Au – Haidhausen	42
Aubing – Lochhausen – Langwied	7
Berg am Laim	8
Feldmoching – Hasenbergl	10
Hadern	9
Laim	21
Ludwigsvorstadt – Isarvorstadt	42
Maxvorstadt	48
Milbertshofen	18
Moosach	26
Nymphenburg	12
Obergiesing	17
Pasing – Obermenzing	63
Ramersdorf – Perlach	14
Schwabing	25
Schwabing-West	19
Schwanthalerhöhe	47
Sendling	47
Sendling – Westpark	5
Thalkirchen – Obersendling – Forstenried – Fürstenried – Solln	15
Trudering – Riem	12
Untergiesing	26

In [67]:

```
#...merging
df = pd.merge(df, venue_data_grouped, on='Neighborhood', how='outer')
df
```

Out[67]:

	Neighborhood	Rent per M2	Latitude	Longitude	Inhabitants	Venue
0	Altstadt – Lehel	1915	48.139130	11.580220	2042	100
1	Schwabing	1876	48.167450	11.570380	69676	25
2	Ludwigsvorstadt – Isarvorstadt	1748	48.132440	11.556090	50620	42
3	Maxvorstadt	1741	48.151092	11.562418	51642	48
4	Schwabing-West	1731	48.168271	11.569873	65892	19
5	Au – Haidhausen	1696	48.128753	11.590536	59752	42
6	Altbogenhausen – Herzogpark	1691	48.149398	11.603792	82138	20
7	Nymphenburg	1647	48.158257	11.503297	95906	12
8	Schwanthalerhöhe	1605	48.134230	11.539034	29663	47
9	Sendling	1532	48.118012	11.539083	39953	47
10	Untergiesing	1511	48.114963	11.570189	51937	26
11	Obergiesing	1466	48.111156	11.588909	51499	17
12	Thalkirchen – Obersendling – Forstenried – Für...	1459	48.097802	11.520807	90790	15
13	Milbertshofen	1414	48.182385	11.575043	73617	18
14	Moosach	1410	48.179895	11.510571	51537	26
15	Laim	1407	48.139551	11.502166	54030	21
16	Pasing – Obermenzing	1389	48.147785	11.460701	70783	63
17	Hadern	1382	48.118064	11.481842	48945	9
18	Trudering – Riem	1379	48.123175	11.664078	67009	12
19	Sendling – Westpark	1375	48.117984	11.516719	55405	5
20	Ramersdorf – Perlach	1369	48.113917	11.615699	108244	14
21	Berg am Laim	1362	48.123483	11.633451	43068	8
22	Allach – Untermenzing	1285	48.195994	11.457013	30737	7
23	Aubing – Lochhausen – Langwied	1278	48.158437	11.414066	42305	7
24	Feldmoching – Hasenbergl	1219	48.213898	11.539273	59391	10

In [68]:

```
#Let's look for correlations
df.corr()
```

Out[68]:

	Rent per M2	Latitude	Longitude	Inhabitants	Venue
Rent per M2	1.000000	-0.093397	0.327951	-0.112285	0.607084
Latitude	-0.093397	1.000000	-0.292908	-0.074140	-0.098452
Longitude	0.327951	-0.292908	1.000000	0.174815	0.071478
Inhabitants	-0.112285	-0.074140	0.174815	1.000000	-0.461057
Venue	0.607084	-0.098452	0.071478	-0.461057	1.000000

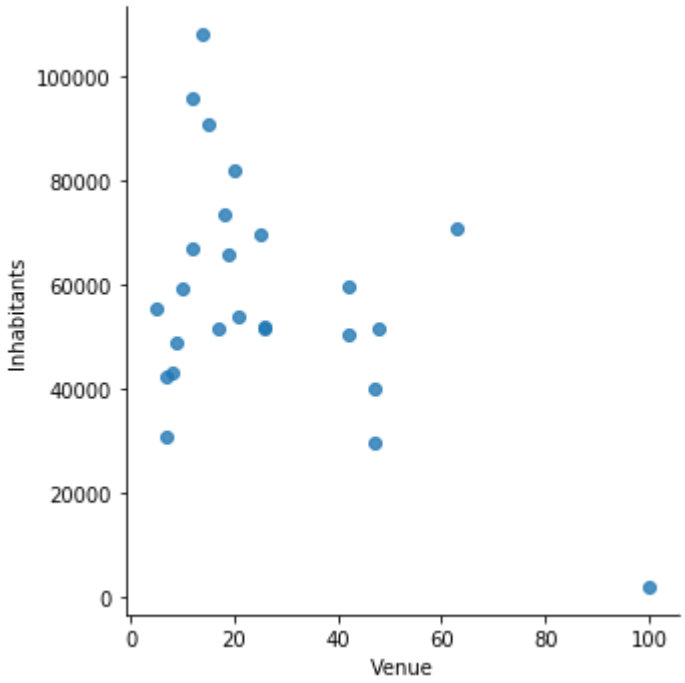
We can observe a strong correlation between Venues and the Rent per M2, as well as the number of Venues and Inhabitants. The 2nd corerlation was expected, since a bigger area of the city (=more inhabitants) typically offers more room and opportunities for businesses. The correlation betweend geospatial data will not be observed, since it's probably random or noise.

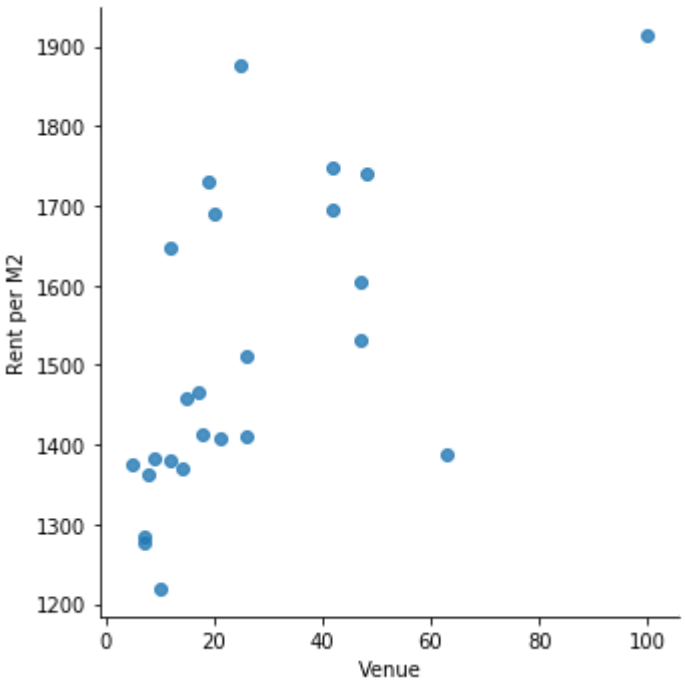
In [73]:

```
sns.lmplot('Venue', 'Inhabitants', data=df, fit_reg=False)
sns.lmplot('Venue', 'Rent per M2', data=df, fit_reg=False)
```

Out[73]:

```
<seaborn.axisgrid.FacetGrid at 0x1a262cc0f0>
```





In [75]:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

In [50]:

```
munich_onehot = pd.get_dummies(munich_venues[['Venue Category']], prefix="", pre
fix_sep="")
munich_onehot
```

Out[50]:

	Afghan Restaurant	American Restaurant	Arcade	Art Museum	Asian Restaurant	Athletics & Sports	Austrian Restaurant	BB Joi
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0
27	0	0	0	1	0	0	0	0
28	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...
630	0	0	0	0	0	0	0	0

	Afghan Restaurant	American Restaurant	Arcade	Art Museum	Asian Restaurant	Athletics & Sports	Austrian Restaurant	BB Joi
631	0	0	0	0	0	0	0	0
632	0	0	0	0	0	0	0	0
633	0	0	0	0	0	0	0	0
634	0	0	0	0	0	0	0	0
635	0	0	0	0	0	0	0	0
636	0	0	0	0	0	0	0	0
637	0	0	0	0	0	0	0	0
638	0	0	0	0	0	0	0	0
639	0	0	0	0	0	0	0	0
640	0	0	0	0	0	0	0	0
641	0	0	0	0	0	0	0	0
642	0	0	0	0	0	0	0	0
643	0	0	0	0	0	0	0	0
644	0	0	0	0	0	0	0	0
645	0	0	0	0	0	0	0	0
646	0	0	0	0	0	0	0	0
647	0	0	0	0	0	0	0	0
648	0	0	0	0	0	0	0	0
649	0	0	0	0	0	0	0	0
650	0	0	0	0	0	0	0	0
651	0	0	0	0	0	0	0	0
652	0	0	0	0	0	0	0	0
653	0	0	0	0	0	0	0	0
654	0	0	0	0	0	0	0	0
655	0	0	0	0	0	0	0	0
656	0	0	0	0	0	0	0	0
657	0	0	0	0	0	0	0	0
658	0	0	0	0	0	0	0	0
659	0	0	0	0	0	0	0	0

660 rows × 162 columns

Results

It seems like there is a strong correlation between the number of venues and inhabitants / rental prices, even though not all areas (e.g. Alstdat) show a strong correlation between the # of inhabitants and the # of venues. The person starting the business should look for an area with a high number of inhabitants and relativley high rental prices. This could e.g. be Schwabing.

Discussion

We only took a few factors (rental prices and inhabitants) into consideration. There might be many more factors affecting the number of venues, e.g. number of tourists or people working in a certain area. Also data was not normalized. The analysis is therefore just considered a training and additional analysis should be performed to gather additional insights as well as to improve the information at hand for decision making. We also haven't looked into the nature of the venues (e.g. restaurants vs. sports clubs) or the style of e.g. a restaurant.

Conclusion

First steps were made to get an understanding of which areas offer potential for future business owners. The analysis should be considered as a starting point and the two factors that were identified are probably not enough to get a full understanding of what areas offer potential.