WQD7005 - Data Mining

MIDTERM EXAM

Matrix Number : 17043640

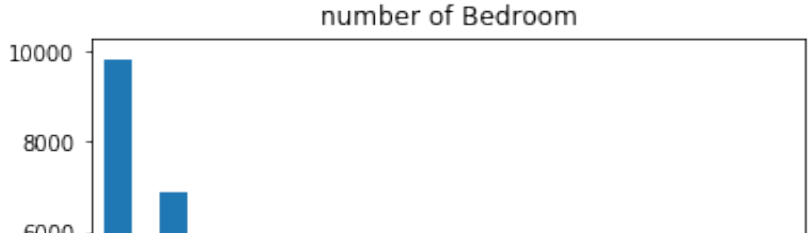
Name : Gunasegarran Magadevan

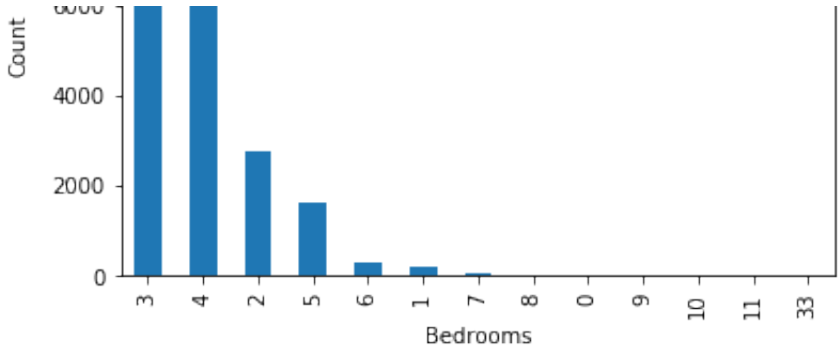
In [1]:
```
# Import packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# Assign url of file: url
url = 'https://files.osf.io/v1/resources/bvn42/providers/osfstorage

# Read file into a DataFrame: df
df = pd.read_csv(url, sep=",")

df.to_csv('HouseData.csv')

# Print the head of the DataFrame
print(df.head())

print(df.describe())

# Plot first column of df
df['bedrooms'].value_counts().plot(kind='bar')
plt.title('number of Bedroom')
plt.xlabel('Bedrooms')
plt.ylabel('Count')
sns.despine
plt.savefig('number of Bedroom.jpg')
```

```
           id             date     price  bedrooms  bathrooms  sqf
t_living  \
0  7129300520  20141013T000000  221900.0         3       1.00
1180
1  6414100192  20141209T000000  538000.0         3       2.25
2570
2  5631500400  20150225T000000  180000.0         2       1.00
770
3  2487200875  20141209T000000  604000.0         4       3.00
1960
4  1954400510  20150218T000000  510000.0         3       2.00
1680

   sqft_lot  floors  waterfront  view  ...  grade  sqft_above  sqf
t_basement  \
```

```
0      5650     1.0           0     0  ...      7        1180
0
1      7242     2.0           0     0  ...      7        2170
400
2     10000     1.0           0     0  ...      6         770
0
3      5000     1.0           0     0  ...      7        1050
910
4      8080     1.0           0     0  ...      8        1680
0

   yr_built  yr_renovated  zipcode      lat     long  sqft_living1
5  \
0      1955             0    98178  47.5112 -122.257           134
0
1      1951          1991    98125  47.7210 -122.319           169
0
2      1933             0    98028  47.7379 -122.233           272
0
3      1965             0    98136  47.5208 -122.393           136
0
4      1987             0    98074  47.6168 -122.045           180
0

   sqft_lot15
0        5650
1        7639
2        8062
3        5000
4        7503

[5 rows x 21 columns]
                 id         price      bedrooms     bathrooms   sq
ft_living  \
count  2.161300e+04  2.161300e+04  21613.000000  21613.000000  216
13.000000
mean   4.580302e+09  5.400881e+05      3.370842      2.114757   20
79.899736
std    2.876566e+09  3.671272e+05      0.930062      0.770163    9
18.440897
min    1.000102e+06  7.500000e+04      0.000000      0.000000    2
90.000000
25%    2.123049e+09  3.219500e+05      3.000000      1.750000   14
27.000000
50%    3.904930e+09  4.500000e+05      3.000000      2.250000   19
10.000000
75%    7.308900e+09  6.450000e+05      4.000000      2.500000   25
50.000000
max    9.900000e+09  7.700000e+06     33.000000      8.000000  135
40.000000

           sqft_lot        floors    waterfront          view
condition  \
count  2.161300e+04  21613.000000  21613.000000  21613.000000  216
13.000000
mean   1.510697e+04      1.494309      0.007542      0.234303
3.409430
```

```
std    4.142051e+04      0.539989      0.086517      0.766318
0.650743
min    5.200000e+02      1.000000      0.000000      0.000000
1.000000
25%    5.040000e+03      1.000000      0.000000      0.000000
3.000000
50%    7.618000e+03      1.500000      0.000000      0.000000
3.000000
75%    1.068800e+04      2.000000      0.000000      0.000000
4.000000
max    1.651359e+06      3.500000      1.000000      4.000000
5.000000

              grade    sqft_above  sqft_basement      yr_built  yr
_renovated  \
count  21613.000000  21613.000000   21613.000000  21613.000000  21
613.000000
mean       7.656873   1788.390691     291.509045   1971.005136
84.402258
std        1.175459    828.090978     442.575043     29.373411
401.679240
min        1.000000    290.000000       0.000000   1900.000000
0.000000
25%        7.000000   1190.000000       0.000000   1951.000000
0.000000
50%        7.000000   1560.000000       0.000000   1975.000000
0.000000
75%        8.000000   2210.000000     560.000000   1997.000000
0.000000
max       13.000000   9410.000000    4820.000000   2015.000000   2
015.000000

            zipcode           lat          long  sqft_living15
sqft_lot15
count  21613.000000  21613.000000  21613.000000   21613.000000   2
1613.000000
mean   98077.939805     47.560053   -122.213896    1986.552492   1
2768.455652
std       53.505026      0.138564      0.140828     685.391304   2
7304.179631
min    98001.000000     47.155900   -122.519000     399.000000
651.000000
25%    98033.000000     47.471000   -122.328000    1490.000000
5100.000000
50%    98065.000000     47.571800   -122.230000    1840.000000
7620.000000
75%    98118.000000     47.678000   -122.125000    2360.000000   1
0083.000000
max    98199.000000     47.777600   -121.315000    6210.000000  87
1200.000000
```

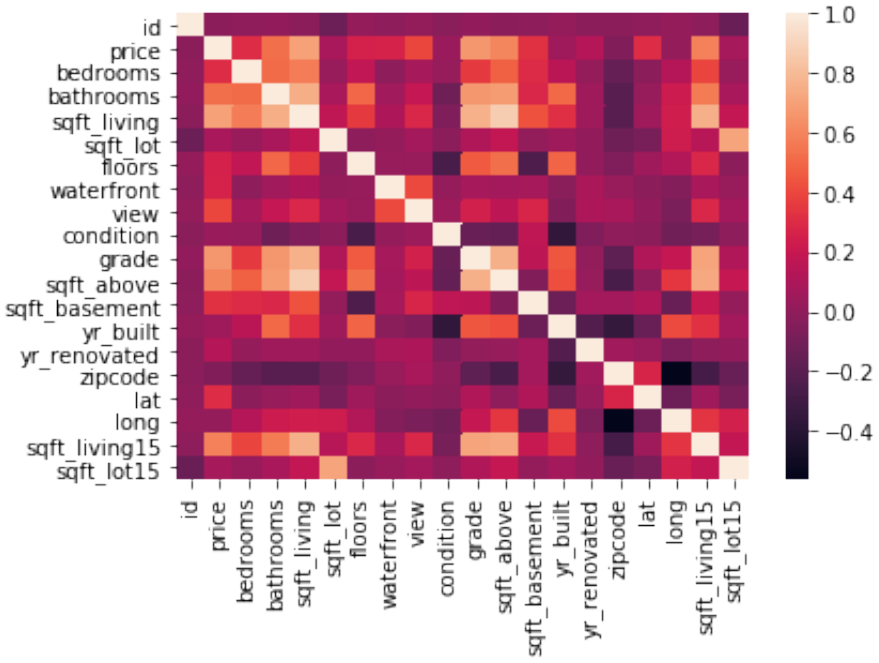




In [2]:
```
# Finding the correlation of the dataset

correlations = df.corr()['bedrooms'].drop('bedrooms')
print(correlations)
```

```
id              0.001286
price           0.308350
bathrooms       0.515884
sqft_living     0.576671
sqft_lot        0.031703
floors          0.175429
waterfront     -0.006582
view            0.079532
condition       0.028472
grade           0.356967
sqft_above      0.477600
sqft_basement   0.303093
yr_built        0.154178
yr_renovated    0.018841
zipcode        -0.152668
lat            -0.008931
long            0.129473
sqft_living15   0.391638
sqft_lot15      0.029244
Name: bedrooms, dtype: float64
```

In [3]:
```
# Draw a heatmap and obtaining a detailed diagram of the correlatio

import seaborn as sns
sns.heatmap(df.corr())
plt.show()
plt.savefig('correlations.jpg', dpi=400)
```




```
<Figure size 432x288 with 0 Axes>
```

In [4]:
```
# Obtaining the features that have a correlation that is above the

def get_features(correlation_threshold):
    abs_corrs = correlations.abs()
    high_correlations = abs_corrs[abs_corrs > correlation_threshold
    return high_correlations
```

In [5]:
```
# taking features with correlation more than 0.08 as input x and 'b
features = get_features(0.08)
print(features)
x = df[features]
y = df['bedrooms']
```

```
['price', 'bathrooms', 'sqft_living', 'floors', 'grade', 'sqft_abo
ve', 'sqft_basement', 'yr_built', 'zipcode', 'long', 'sqft_living1
5']
```

In [6]:
```
# Obtaining the new dataset after eliminating the columns correlati
new_df= df.loc[:, features]
print(new_df)
print(new_df.columns)
```

```
          price  bathrooms  sqft_living  floors  grade  sqft_above
\
0      221900.0       1.00         1180     1.0      7        1180
1      538000.0       2.25         2570     2.0      7        2170
2      180000.0       1.00          770     1.0      6         770
3      604000.0       3.00         1960     1.0      7        1050
4      510000.0       2.00         1680     1.0      8        1680
...         ...        ...          ...     ...    ...         ...
21608  360000.0       2.50         1530     3.0      8        1530
21609  400000.0       2.50         2310     2.0      8        2310
21610  402101.0       0.75         1020     2.0      7        1020
21611  400000.0       2.50         1600     2.0      8        1600
21612  325000.0       0.75         1020     2.0      7        1020

       sqft_basement  yr_built  zipcode     long  sqft_living15
0                  0      1955    98178 -122.257           1340
1                400      1951    98125 -122.319           1690
2                  0      1933    98028 -122.233           2720
3                910      1965    98136 -122.393           1360
4                  0      1987    98074 -122.045           1800
...              ...       ...      ...      ...            ...
21608              0      2009    98103 -122.346           1530
21609              0      2014    98146 -122.362           1830
21610              0      2009    98144 -122.299           1020
21611              0      2004    98027 -122.069           1410
21612              0      2008    98144 -122.299           1020

[21613 rows x 11 columns]
Index(['price', 'bathrooms', 'sqft_living', 'floors', 'grade', 'sq
ft_above',
       'sqft_basement', 'yr_built', 'zipcode', 'long', 'sqft_livin
g15'],
      dtype='object')
```

In [7]:
```
col_missing = new_df.isnull().sum()
print(col_missing)
```

```
price            0
bathrooms        0
sqft_living      0
floors           0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
zipcode          0
long             0
sqft_living15    0
dtype: int64
```

In [8]:
```
# Since there is no missing data, we will deliberately add missing
```

```
import random
new_df2 = new_df.to_csv(r'HouseData.csv',index=False)
new_df3 = pd.read_csv(r'HouseData.csv')

nan_percent = {'id':0.10, 'price':0.10, 'bathrooms':0.10, 'sqft_liv
                        'waterfront':0.10, 'view':0.10, 'condition'
                                    'sqft_basement':0.10, 'yr_built
                                    'lat':0.10, 'long':0.10, 'sqft_
for col in new_df3:
    for i, row_value in new_df3[col].iteritems():
        if random.random() <= nan_percent[col]:
            new_df3[col][i] = np.nan
new_df3.to_csv(r'HouseDataMissing.csv',index=False)
new_df4 = pd.read_csv(r'HouseDataMissing.csv')
new_df4.to_csv('HouseDataMissing.csv')
print(new_df4)
```

```
/Users/gunasegarranmagadevan/opt/anaconda3/lib/python3.7/site-pack
ages/ipykernel_launcher.py:14: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy)

/Users/gunasegarranmagadevan/opt/anaconda3/lib/python3.7/site-pack
ages/pandas/core/indexing.py:670: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy)
  self._setitem_with_indexer(indexer, value)
```

	price	bathrooms	sqft_living	floors	grade	sqft_above
0	221900.0	NaN	1180.0	1.0	7.0	1180.0
1	538000.0	2.25	2570.0	2.0	7.0	2170.0
2	180000.0	1.00	770.0	1.0	6.0	770.0
3	604000.0	3.00	1960.0	1.0	7.0	1050.0
4	510000.0	2.00	1680.0	1.0	8.0	1680.0
...	...	...	...	...	...	...
21608	360000.0	2.50	1530.0	3.0	8.0	1530.0
21609	400000.0	2.50	2310.0	2.0	8.0	2310.0
21610	402101.0	0.75	1020.0	2.0	7.0	NaN
21611	400000.0	2.50	1600.0	NaN	8.0	1600.0
21612	325000.0	0.75	1020.0	2.0	7.0	NaN

	sqft_basement	yr_built	zipcode	long	sqft_living15
0	0.0	1955.0	NaN	-122.257	1340.0
1	400.0	1951.0	98125.0	-122.319	NaN
2	0.0	1933.0	98028.0	-122.233	2720.0
3	910.0	1965.0	98136.0	-122.393	NaN
4	0.0	NaN	98074.0	-122.045	1800.0
...	...	...	...	...	...

```
21608            0.0    2009.0  98103.0 -122.346         1530.0
21609            0.0    2014.0  98146.0 -122.362         1830.0
21610            0.0    2009.0      NaN -122.299         1020.0
21611            0.0    2004.0  98027.0 -122.069         1410.0
21612            0.0    2008.0  98144.0 -122.299         1020.0

[21613 rows x 11 columns]
```

In [9]:
```
# Determining the number of missing values in each column

col_missing = new_df4.isnull().sum()
print(col_missing)
```

```
price            2196
bathrooms        2220
sqft_living      2201
floors           2160
grade            2171
sqft_above       2229
sqft_basement    2151
yr_built         2260
zipcode          2161
long             2071
sqft_living15    2106
dtype: int64
```

In [10]:
```
# Imputing the missing values with the mean of each column

new_df7 = new_df4.fillna(new_df4.mean())
print(new_df7)
```

```
          price  bathrooms  sqft_living    floors  grade   sqft_ab
ove  \
0      221900.0   2.114732       1180.0  1.000000    7.0  1180.000
000
1      538000.0   2.250000       2570.0  2.000000    7.0  2170.000
000
2      180000.0   1.000000        770.0  1.000000    6.0   770.000
000
3      604000.0   3.000000       1960.0  1.000000    7.0  1050.000
000
4      510000.0   2.000000       1680.0  1.000000    8.0  1680.000
000
...         ...        ...          ...       ...    ...
...
21608  360000.0   2.500000       1530.0  3.000000    8.0  1530.000
000
21609  400000.0   2.500000       2310.0  2.000000    8.0  2310.000
000
21610  402101.0   0.750000       1020.0  2.000000    7.0  1785.880
623
21611  400000.0   2.500000       1600.0  1.494422    8.0  1600.000
000
21612  325000.0   0.750000       1020.0  2.000000    7.0  1785.880
623

       sqft_basement     yr_built       zipcode     long  sqft_liv
ing15
0                0.0  1955.000000  98077.989153 -122.257    1340.0
00000
1              400.0  1951.000000  98125.000000 -122.319    1985.5
04639
2                0.0  1933.000000  98028.000000 -122.233    2720.0
00000
3              910.0  1965.000000  98136.000000 -122.393    1985.5
04639
4                0.0  1971.031571  98074.000000 -122.045    1800.0
00000
...              ...          ...           ...      ...
...
21608            0.0  2009.000000  98103.000000 -122.346    1530.0
00000
21609            0.0  2014.000000  98146.000000 -122.362    1830.0
00000
21610            0.0  2009.000000  98077.989153 -122.299    1020.0
00000
21611            0.0  2004.000000  98027.000000 -122.069    1410.0
00000
21612            0.0  2008.000000  98144.000000 -122.299    1020.0
00000

[21613 rows x 11 columns]
```

In [11]:
```
# Before data integration
url_house = 'https://files.osf.io/v1/resources/bvn42/providers/osfs

house_df = pd.read_csv(url_house, sep=",")
print(house_df)
house_column = list(house_df.columns)
```

```
                id             date     price  bedrooms  bathrooms
\
0       7129300520  20141013T000000  221900.0         3       1.00
1       6414100192  20141209T000000  538000.0         3       2.25
2       5631500400  20150225T000000  180000.0         2       1.00
3       2487200875  20141209T000000  604000.0         4       3.00
4       1954400510  20150218T000000  510000.0         3       2.00
...            ...              ...       ...       ...        ...
21608    263000018  20140521T000000  360000.0         3       2.50
21609   6600060120  20150223T000000  400000.0         4       2.50
21610   1523300141  20140623T000000  402101.0         2       0.75
21611    291310100  20150116T000000  400000.0         3       2.50
21612   1523300157  20141015T000000  325000.0         2       0.75

       sqft_living  sqft_lot  floors  waterfront  view  ...  grade
\
0             1180      5650     1.0           0     0  ...      7
1             2570      7242     2.0           0     0  ...      7
2              770     10000     1.0           0     0  ...      6
3             1960      5000     1.0           0     0  ...      7
4             1680      8080     1.0           0     0  ...      8
...            ...       ...     ...         ...   ...  ...    ...
21608         1530      1131     3.0           0     0  ...      8
21609         2310      5813     2.0           0     0  ...      8
21610         1020      1350     2.0           0     0  ...      7
21611         1600      2388     2.0           0     0  ...      8
21612         1020      1076     2.0           0     0  ...      7

       sqft_above  sqft_basement  yr_built  yr_renovated  zipcode
lat  \
0            1180              0      1955             0    98178
47.5112
1            2170            400      1951          1991    98125
47.7210
2             770              0      1933             0    98028
47.7379
3            1050            910      1965             0    98136
47.5208
4            1680              0      1987             0    98074
47.6168
...           ...            ...       ...           ...      ...
...
21608        1530              0      2009             0    98103
47.6993
21609        2310              0      2014             0    98146
47.5107
21610        1020              0      2009             0    98144
47.5944
21611        1600              0      2004             0    98027
47.5345
```

```
21612        1020              0      2008             0    98144
47.5941

          long  sqft_living15  sqft_lot15
0     -122.257           1340        5650
1     -122.319           1690        7639
2     -122.233           2720        8062
3     -122.393           1360        5000
4     -122.045           1800        7503
...        ...            ...         ...
21608 -122.346           1530        1509
21609 -122.362           1830        7200
21610 -122.299           1020        2007
21611 -122.069           1410        1287
21612 -122.299           1020        1357

[21613 rows x 21 columns]
```

In [12]:

```
# Data Integration
import pandas as pd

print(house_column)

house_df = pd.read_csv(url_house, sep=",")
new_df5 = pd.merge(df, house_df, on = (['id', 'date', 'price', 'bat
                                        'sqft_lot', 'floors', 'wate
                                        'sqft_above', 'sqft_basemen
                                        'lat', 'long', 'sqft_living
new_df6 = new_df5.loc[:, ['id', 'date', 'price', 'bedrooms_y', 'bat
                          'waterfront', 'view', 'condition', 'grade
                          'yr_renovated', 'zipcode', 'lat', 'long',
print(new_df6)
```

```
['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living', 's
qft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade', 's
qft_above', 'sqft_basement', 'yr_built', 'yr_renovated', 'zipcode'
, 'lat', 'long', 'sqft_living15', 'sqft_lot15']
               id             date     price  bedrooms_y  bathroom
s  \
0      7129300520  20141013T000000  221900.0           3       1.0
0
1      6414100192  20141209T000000  538000.0           3       2.2
5
2      5631500400  20150225T000000  180000.0           2       1.0
0
3      2487200875  20141209T000000  604000.0           4       3.0
0
4      1954400510  20150218T000000  510000.0           3       2.0
0
...           ...              ...       ...         ...        ..
.
21608   263000018  20140521T000000  360000.0           3       2.5
0
21609  6600060120  20150223T000000  400000.0           4       2.5
0
21610  1523300141  20140623T000000  402101.0           2       0.7
5
```

```
21611   291310100  20150116T000000  400000.0           3       2.5
0
21612  1523300157  20141015T000000  325000.0           2       0.7
5

       sqft_living  sqft_lot  floors  waterfront  view  ...  grade
\
0             1180      5650     1.0           0     0  ...      7
1             2570      7242     2.0           0     0  ...      7
2              770     10000     1.0           0     0  ...      6
3             1960      5000     1.0           0     0  ...      7
4             1680      8080     1.0           0     0  ...      8
...            ...       ...     ...         ...   ...  ...    ...
21608         1530      1131     3.0           0     0  ...      8
21609         2310      5813     2.0           0     0  ...      8
21610         1020      1350     2.0           0     0  ...      7
21611         1600      2388     2.0           0     0  ...      8
21612         1020      1076     2.0           0     0  ...      7

       sqft_above  sqft_basement  yr_built  yr_renovated  zipcode
lat  \
0            1180              0      1955             0    98178
47.5112
1            2170            400      1951          1991    98125
47.7210
2             770              0      1933             0    98028
47.7379
3            1050            910      1965             0    98136
47.5208
4            1680              0      1987             0    98074
47.6168
...           ...            ...       ...           ...      ...
...
21608        1530              0      2009             0    98103
47.6993
21609        2310              0      2014             0    98146
47.5107
21610        1020              0      2009             0    98144
47.5944
21611        1600              0      2004             0    98027
47.5345
21612        1020              0      2008             0    98144
47.5941

          long  sqft_living15  sqft_lot15
0     -122.257           1340        5650
1     -122.319           1690        7639
2     -122.233           2720        8062
3     -122.393           1360        5000
4     -122.045           1800        7503
...        ...            ...         ...
21608 -122.346           1530        1509
21609 -122.362           1830        7200
21610 -122.299           1020        2007
21611 -122.069           1410        1287
21612 -122.299           1020        1357
```

```
[21613 rows x 21 columns]
```

In [13]:
```
# Before data transformation
print(new_df7)
print(list(new_df7.columns))
```

```
          price  bathrooms  sqft_living    floors  grade   sqft_ab
ove  \
0      221900.0   2.114732       1180.0  1.000000    7.0  1180.000
000
1      538000.0   2.250000       2570.0  2.000000    7.0  2170.000
000
2      180000.0   1.000000        770.0  1.000000    6.0   770.000
000
3      604000.0   3.000000       1960.0  1.000000    7.0  1050.000
000
4      510000.0   2.000000       1680.0  1.000000    8.0  1680.000
000
...         ...        ...          ...       ...    ...
...
21608  360000.0   2.500000       1530.0  3.000000    8.0  1530.000
000
21609  400000.0   2.500000       2310.0  2.000000    8.0  2310.000
000
21610  402101.0   0.750000       1020.0  2.000000    7.0  1785.880
623
21611  400000.0   2.500000       1600.0  1.494422    8.0  1600.000
000
21612  325000.0   0.750000       1020.0  2.000000    7.0  1785.880
623

       sqft_basement     yr_built       zipcode     long  sqft_liv
ing15
0                0.0  1955.000000  98077.989153 -122.257    1340.0
00000
1              400.0  1951.000000  98125.000000 -122.319    1985.5
04639
2                0.0  1933.000000  98028.000000 -122.233    2720.0
00000
3              910.0  1965.000000  98136.000000 -122.393    1985.5
04639
4                0.0  1971.031571  98074.000000 -122.045    1800.0
00000
...              ...          ...           ...      ...
...
21608            0.0  2009.000000  98103.000000 -122.346    1530.0
00000
21609            0.0  2014.000000  98146.000000 -122.362    1830.0
00000
21610            0.0  2009.000000  98077.989153 -122.299    1020.0
00000
21611            0.0  2004.000000  98027.000000 -122.069    1410.0
00000
21612            0.0  2008.000000  98144.000000 -122.299    1020.0
00000

[21613 rows x 11 columns]
```

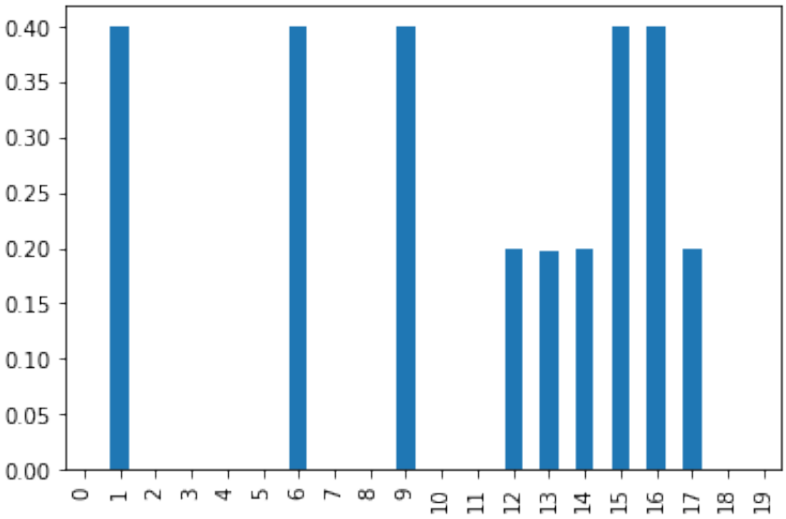
```
['price', 'bathrooms', 'sqft_living', 'floors', 'grade', 'sqft_abo
ve', 'sqft_basement', 'yr_built', 'zipcode', 'long', 'sqft_living1
5']
```

In [14]:
```
# Normalize the dataset
import pandas as pd
from sklearn import preprocessing

x = new_df7.values
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
new_df5 = pd.DataFrame(x_scaled)
new_df5.columns = ['price', 'bathrooms', 'sqft_living', 'floors', '
print(new_df5)
```

```
          price  bathrooms  sqft_living    floors     grade  sqft_
above  \
0      0.019266   0.264342     0.067170  0.000000  0.500000    0.0
89602
1      0.060721   0.281250     0.172075  0.400000  0.500000    0.1
99115
2      0.013770   0.125000     0.036226  0.000000  0.416667    0.0
44248
3      0.069377   0.375000     0.126038  0.000000  0.500000    0.0
75221
4      0.057049   0.250000     0.104906  0.000000  0.583333    0.1
44912
...         ...        ...          ...       ...       ...
...
21608  0.037377   0.312500     0.093585  0.800000  0.583333    0.1
28319
21609  0.042623   0.312500     0.152453  0.400000  0.583333    0.2
14602
21610  0.042898   0.093750     0.055094  0.400000  0.500000    0.1
56624
21611  0.042623   0.312500     0.098868  0.197769  0.583333    0.1
36062
21612  0.032787   0.093750     0.055094  0.400000  0.500000    0.1
56624

       sqft_basement  yr_built   zipcode      long  sqft_living15
0           0.000000  0.478261  0.388834  0.217608       0.161934
1           0.082988  0.443478  0.626263  0.166113       0.273017
2           0.000000  0.286957  0.136364  0.237542       0.399415
3           0.188797  0.565217  0.681818  0.104651       0.273017
4           0.000000  0.617666  0.368687  0.393688       0.241094
...              ...       ...       ...       ...            ...
21608       0.000000  0.947826  0.515152  0.143688       0.194631
21609       0.000000  0.991304  0.732323  0.130399       0.246257
21610       0.000000  0.947826  0.388834  0.182724       0.106866
21611       0.000000  0.904348  0.131313  0.373754       0.173980
21612       0.000000  0.939130  0.722222  0.182724       0.106866

[21613 rows x 11 columns]
```

In [15]:
```
# Plotting the normalized dataframe

## Taking the first 20 in the 'floors' column
new_df5['floors'].head(20).plot(kind='bar')
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x7faddcd7efd0>

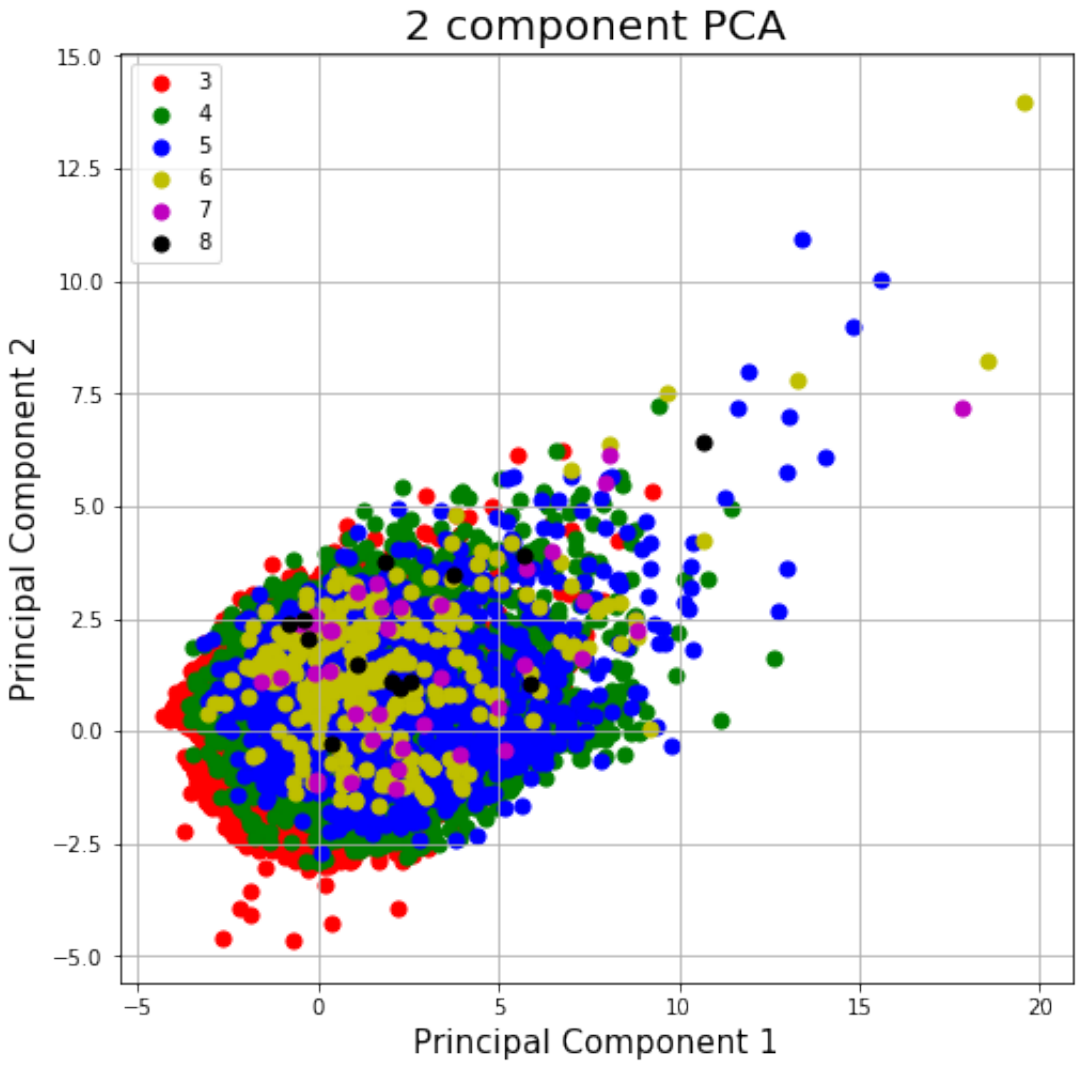


In [16]:
```
# Standardize the data

from sklearn.preprocessing import StandardScaler
features = ['price', 'bathrooms', 'sqft_living', 'floors', 'grade',
new_df6[['price', 'bathrooms', 'sqft_living', 'floors', 'grade', 's
new_df6 = new_df6.astype({"bedrooms_y": str})
# Separating out the features
x = new_df6.loc[:, features].values
# Separating out the target
y = new_df6.loc[:,['bedrooms_y']].values
# Standardizing the features
x = StandardScaler().fit_transform(x)
```

In [17]:
```
# Dimension reduction process using Principle Component Analysis (P

## PCA Projection to 2D
### 9 columns (4D) into 2D
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal compo
print(principalDf)
finalDf = pd.concat([principalDf, new_df6[['bedrooms_y']]], axis =
print(finalDf)
```

```
       principal component 1  principal component 2
0                  -2.780171               0.330768
1                  -0.089641               0.918673
2                  -2.516141              -0.636841
3                  -1.007941               2.025100
4                  -0.222785              -1.047851
...                      ...                    ...
21608               0.237680              -1.073149
21609               0.594853              -0.181787
21610              -2.124710              -0.814835
21611               0.334607              -2.057799
21612              -2.197749              -0.871952

[21613 rows x 2 columns]
       principal component 1  principal component 2 bedrooms_y
0                  -2.780171               0.330768          3
1                  -0.089641               0.918673          3
2                  -2.516141              -0.636841          2
3                  -1.007941               2.025100          4
4                  -0.222785              -1.047851          3
...                      ...                    ...        ...
21608               0.237680              -1.073149          3
21609               0.594853              -0.181787          4
21610              -2.124710              -0.814835          2
21611               0.334607              -2.057799          3
21612              -2.197749              -0.871952          2

[21613 rows x 3 columns]
```

In [18]:
```
# Visualize 2D Projection
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 component PCA', fontsize = 20)
bedrooms = ['3', '4', '5','6','7','8']
colors = ['r', 'g', 'b', 'y', 'm', 'k']
for bedrooms1, color in zip(bedrooms,colors):
    indicesToKeep = finalDf['bedrooms_y'] == bedrooms1
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(bedrooms)
ax.grid()

plt.savefig('2 component PCA.jpg')
```



In [ ]: