Analytics_Regression

October 8, 2020

[1]:
```
import pandas as pd
```

[2]:
```
import os
os.chdir("/Emily/Teaching/Python/Code")
df_full = pd.read_csv('df_full_listing.csv', parse_dates = True, encoding =␣
 ↪"utf_8_sig")
var_need = ['id', 'latitude', 'longitude', 'zipcode',  'amenities',
            'room_type', 'bed_type', 'accommodates','bathrooms',
           'bedrooms', 'beds', 'reviews_per_month',
          'instant_bookable',  'cancellation_policy',␣
 ↪'require_guest_phone_verification',
         'calculated_host_listings_count',#'is_business_travel_ready','price',
           'calculated_host_listings_count_entire_homes',
       ␣
 ↪'calculated_host_listings_count_private_rooms','security_deposit','cleaning_fee']
df_slt = pd.DataFrame(df_full, columns = var_need)
```

[3]:
```
df_slt.head()
```

[3]:
```
     id  latitude  longitude  zipcode  \
0  1078  30.30123  -97.73674  78705.0
1  2265  30.27750  -97.71398  78702.0
2  5245  30.27577  -97.71379  78702.0
3  5456  30.26112  -97.73448  78702.0
4  5769  30.45596  -97.78370  78729.0

                                           amenities        room_type  \
0  {TV,Internet,Wifi,"Air conditioning",Kitchen,"...  Entire home/apt
1  {TV,"Cable TV",Internet,Wifi,"Air conditioning...  Entire home/apt
2  {TV,"Cable TV",Internet,Wifi,"Air conditioning...     Private room
3  {TV,Wifi,"Air conditioning",Kitchen,"Pets live...  Entire home/apt
4  {TV,"Cable TV",Internet,Wifi,"Air conditioning...     Private room

   bed_type  accommodates  bathrooms  bedrooms  beds  reviews_per_month  \
0  Real Bed             2        1.0       1.0   1.0               1.70
1  Real Bed             4        2.0       2.0   2.0               0.19
2  Real Bed             2        1.0       1.0   1.0               0.07
3  Real Bed             3        1.0       1.0   2.0               3.88
```

```
4  Real Bed             2        1.0       1.0   1.0               2.22

  instant_bookable          cancellation_policy  \
0                t                     flexible
1                f  strict_14_with_grace_period
2                f  strict_14_with_grace_period
3                f  strict_14_with_grace_period
4                f                     moderate

  require_guest_phone_verification  calculated_host_listings_count  \
0                                f                               2
1                                f                               3
2                                f                               3
3                                t                               1
4                                t                               1

   calculated_host_listings_count_entire_homes  \
0                                            2
1                                            2
2                                            2
3                                            1
4                                            0

   calculated_host_listings_count_private_rooms security_deposit cleaning_fee
0                                             0              NaN       $35.00
1                                             1          $500.00      $100.00
2                                             1          $500.00       $75.00
3                                             0          $100.00          NaN
4                                             1              NaN          NaN
```

[4]:
```
df_slt['instant_bookable'].value_counts()
```

[4]:
```
f    6160
t    5632
Name: instant_bookable, dtype: int64
```

[5]:
```
df_slt['cancellation_policy'].value_counts()
```

[5]:
```
strict_14_with_grace_period    4486
flexible                       3813
moderate                       3257
super_strict_30                 225
super_strict_60                   7
strict                            4
Name: cancellation_policy, dtype: int64
```

[6]:
```
df_slt = df_slt.dropna()
```

[7]:
```
#Turkey IQR
import numpy as np
```

```
def find_outliers_tukey(x):
    q1 = np.percentile(x, 25)
    q3 = np.percentile(x, 75)
    iqr = q3-q1
    floor = q1 - 1.5*iqr
    ceiling = q3 + 1.5*iqr
    outlier_indices = list(x.index[(x < floor)|(x > ceiling)])
    outlier_values = list(x[outlier_indices])

    return outlier_indices, outlier_values
```

[8]:
```
df_slt['require_guest_phone_verification'].value_counts()
```

[8]:
```
f    6572
t     331
Name: require_guest_phone_verification, dtype: int64
```

[9]:
```
df_slt.cleaning_fee.str.startswith('$').sum()
```

[9]:
```
6903
```

[10]:
```
df_slt['cleaning_fee'] = df_slt.cleaning_fee.str.slice(1,)
df_slt['cleaning_fee'] = df_slt['cleaning_fee'].str.replace(',', '')
df_slt['cleaning_fee'] = df_slt['cleaning_fee'].astype(float)
df_slt['cleaning_fee'].describe()
```

[10]:
```
count    6903.000000
mean       84.364914
std        77.284520
min         0.000000
25%        30.000000
50%        70.000000
75%       100.000000
max       704.000000
Name: cleaning_fee, dtype: float64
```

[11]:
```
tukey_indices, tukey_values = find_outliers_tukey(df_slt['cleaning_fee'])
df_slt = df_slt[~df_slt['cleaning_fee'].isin(tukey_values)]
df_slt['cleaning_fee'].describe()
```

[11]:
```
count    6473.000000
mean       69.927236
std        49.983657
min         0.000000
25%        30.000000
50%        60.000000
75%       100.000000
max       204.000000
Name: cleaning_fee, dtype: float64
```

[12]:
```
df_slt.security_deposit.str.startswith('$').sum()
```

[12]:
```
6473
```

[13]:
```
df_slt['security_deposit'] = df_slt.security_deposit.str.slice(1,)
df_slt['security_deposit'] = df_slt['security_deposit'].str.replace(',', '')
df_slt['security_deposit'] = df_slt['security_deposit'].astype(float)
df_slt['security_deposit'].describe()
```

[13]:
```
count    6473.000000
mean      255.555075
std       443.499436
min         0.000000
25%         0.000000
50%       150.000000
75%       300.000000
max      5000.000000
Name: security_deposit, dtype: float64
```

[14]:
```
tukey_indices, tukey_values = find_outliers_tukey(df_slt['security_deposit'])
df_slt = df_slt[~df_slt['security_deposit'].isin(tukey_values)]
df_slt['security_deposit'].describe()
```

[14]:
```
count    6056.000000
mean      171.544419
std       186.126374
min         0.000000
25%         0.000000
50%       100.000000
75%       270.000000
max       750.000000
Name: security_deposit, dtype: float64
```

[15]:
```
data_store = pd.HDFStore('property_cleaned.h5')
# Retrieve data using key
df_property = data_store['preprocessed_property']
data_store.close()
```

[16]:
```
df_property.head()
```

[16]:

	id	property_type
0	1078	Hotel
1	2265	House
2	5245	House
3	5456	Hotel
4	5769	House

[17]:
```
data_store = pd.HDFStore('price_cleaned.h5')
# Retrieve data using key
df_price = data_store['preprocessed_price']
data_store.close()
```

[18]:
```
df_price.head()
```

[18]:
```
     id  price
0  1078   85.0
1  2265  225.0
2  5245  100.0
3  5456   95.0
4  5769   40.0
```

[19]:
```
data_store = pd.HDFStore('Reviews_Clustered.h5')
# Retrieve data using key
df_review = data_store['df_review_clustered']
data_store.close()
df_review.head()
```

[19]:
```
     id Review
0  1078   Good
1  2265    Bad
2  5245    Bad
3  5456   Good
4  5769   Good
```

1 Sample Properties Nearby

[20]:
```
import numpy as np
```

[21]:
```
df_slt['zipcode'].value_counts()
```

[21]:
```
78704.0    1234
78702.0     916
78741.0     400
78701.0     385
78703.0     295
78745.0     294
78705.0     268
78751.0     253
78723.0     219
78721.0     141
78722.0     134
78757.0     130
78758.0     121
78746.0     106
78744.0     105
78734.0      93
78748.0      87
78756.0      79
78731.0      62
78749.0      57
78759.0      57
78752.0      56
```

```
78753.0      55
78724.0      52
78729.0      49
78727.0      48
78737.0      44
78747.0      34
78754.0      33
78750.0      31
78736.0      30
78728.0      30
78733.0      24
78732.0      20
78735.0      20
78717.0      18
78738.0      18
78725.0      13
78739.0      12
78730.0      10
78726.0       8
78660.0       5
78719.0       3
78620.0       2
78681.0       1
78669.0       1
78742.0       1
78652.0       1
78767.0       1
Name: zipcode, dtype: int64
```

[22]:
```
df_sample = df_slt[df_slt['zipcode'].isin([78751])]
df_sample.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 253 entries, 79 to 11689
Data columns (total 20 columns):
id                                                253 non-null int64
latitude                                          253 non-null float64
longitude                                         253 non-null float64
zipcode                                           253 non-null float64
amenities                                         253 non-null object
room_type                                         253 non-null object
bed_type                                          253 non-null object
accommodates                                      253 non-null int64
bathrooms                                         253 non-null float64
bedrooms                                          253 non-null float64
beds                                              253 non-null float64
reviews_per_month                                 253 non-null float64
instant_bookable                                  253 non-null object
```

```
cancellation_policy                               253 non-null object
require_guest_phone_verification                  253 non-null object
calculated_host_listings_count                    253 non-null int64
calculated_host_listings_count_entire_homes       253 non-null int64
calculated_host_listings_count_private_rooms      253 non-null int64
security_deposit                                  253 non-null float64
cleaning_fee                                      253 non-null float64
dtypes: float64(9), int64(5), object(6)
memory usage: 41.5+ KB
```

[23]:
```
df_sample = pd.merge(df_sample, df_price, on = 'id', how = 'inner')
df_sample = pd.merge(df_sample, df_property, on = 'id', how = 'inner')
df_sample = pd.merge(df_sample, df_review, on = 'id', how = 'inner')
df_sample.head()
```

[23]:

	id	latitude	longitude	zipcode	\
0	113779	30.30326	-97.73057	78751.0	
1	113791	30.30220	-97.73013	78751.0	
2	140474	30.30226	-97.73143	78751.0	
3	140504	30.30242	-97.72954	78751.0	
4	233930	30.30271	-97.73123	78751.0	

	amenities	room_type	\
0	{TV,"Cable TV",Internet,Wifi,"Air conditioning...	Entire home/apt	
1	{TV,"Cable TV",Wifi,"Air conditioning",Kitchen...	Private room	
2	{TV,"Cable TV",Internet,Wifi,"Air conditioning...	Entire home/apt	
3	{TV,"Cable TV",Internet,Wifi,"Air conditioning...	Entire home/apt	
4	{TV,"Cable TV",Internet,Wifi,"Air conditioning...	Entire home/apt	

	bed_type	accommodates	bathrooms	bedrooms	...	\
0	Real Bed	2	1.0	1.0	...	
1	Real Bed	2	1.0	1.0	...	
2	Real Bed	4	1.0	2.0	...	
3	Real Bed	2	1.0	1.0	...	
4	Real Bed	4	1.0	2.0	...	

	cancellation_policy	require_guest_phone_verification	\
0	super_strict_30	t	
1	strict_14_with_grace_period	f	
2	super_strict_30	t	
3	super_strict_30	t	
4	super_strict_30	t	

	calculated_host_listings_count	calculated_host_listings_count_entire_homes	\
0	9	9	
1	2	1	
2	9	9	
3	9	9	

```
4                                9                                                 9

  calculated_host_listings_count_private_rooms  security_deposit  \
0                                            0             200.0
1                                            1               0.0
2                                            0             200.0
3                                            0             200.0
4                                            0             200.0

   cleaning_fee  price  property_type  Review
0          60.0  119.0          House    Good
1         130.0   60.0          House    Good
2          60.0  136.0          House    Good
3          60.0  119.0          House    Good
4          60.0  159.0          House    Good

[5 rows x 23 columns]
```

[24]:
```
df_sample.property_type.value_counts()
```

[24]:
```
House          116
Apartment       68
Hotel           29
Other           22
Condominium     14
Name: property_type, dtype: int64
```

[25]:
```
df_sample = df_sample[df_sample['property_type'].isin(['House'])]
df_sample.room_type.value_counts()
```

[25]:
```
Entire home/apt    76
Private room       39
Shared room         1
Name: room_type, dtype: int64
```

[26]:
```
df_sample = df_sample[df_sample['room_type'] == 'Entire home/apt']
df_sample.bed_type.value_counts()
```

[26]:
```
Real Bed    76
Name: bed_type, dtype: int64
```

[27]:
```
df_sample['tv'] = df_sample['amenities'].str.contains('TV')
df_sample['tv'].value_counts()
```

[27]:
```
True     65
False    11
Name: tv, dtype: int64
```

[28]:
```
df_sample['washer'] = df_sample['amenities'].str.contains('[Ww]asher')
df_sample['dryer'] = df_sample['amenities'].str.contains('Dryer')
df_sample['washer'].value_counts()
```

[28]:
```
True     65
False    11
Name: washer, dtype: int64
```

[29]:
```
df_sample['dryer'].value_counts()
```

[29]:
```
True     60
False    16
Name: dryer, dtype: int64
```

[30]:
```
df_sample['parking'] = df_sample['amenities'].str.contains('Free parking on␣
 ↪premise')
df_sample['parking'].value_counts()
```

[30]:
```
True     68
False     8
Name: parking, dtype: int64
```

[31]:
```
df_sample['laptop'] = df_sample['amenities'].str.contains('[Ll]aptop friendly␣
 ↪workspace')
df_sample['laptop'].value_counts()
```

[31]:
```
True     62
False    14
Name: laptop, dtype: int64
```

[32]:
```
df_sample['instant_bookable'].value_counts()
```

[32]:
```
f    48
t    28
Name: instant_bookable, dtype: int64
```

[33]:
```
df_sample['cancellation_policy'].value_counts()
```

[33]:
```
strict_14_with_grace_period    36
moderate                       27
flexible                        8
super_strict_30                 5
Name: cancellation_policy, dtype: int64
```

[34]:
```
df_sample['require_guest_phone_verification'].value_counts()
```

[34]:
```
f    64
t    12
Name: require_guest_phone_verification, dtype: int64
```

[35]:
```
df_sample['Review'].value_counts()
```

[35]:
```
Good    56
Bad     20
Name: Review, dtype: int64
```

[36]:
```
df_sample[['id','price','property_type', 'washer', 'Review']].head()
```

[36]:
```
        id  price property_type  washer Review
0   113779  119.0         House    True   Good
2   140474  136.0         House    True   Good
3   140504  119.0         House    True   Good
4   233930  159.0         House    True   Good
6   347802  119.0         House   False   Good
```

[37]:
```
todummy_list = ['instant_bookable', 'cancellation_policy',␣
 ↪'Review','require_guest_phone_verification']
def dummy_df(df, todummy_list):
    for x in todummy_list:
        dummies = pd.get_dummies(df[x], prefix=x, dummy_na=False)
        df = df.drop(x, 1)
        df = pd.concat([df, dummies], axis=1)
    return df

df_R = dummy_df(df_sample, todummy_list)
df_R.head()
```

[37]:
```
        id  latitude  longitude  zipcode  \
0   113779  30.30326  -97.73057  78751.0
2   140474  30.30226  -97.73143  78751.0
3   140504  30.30242  -97.72954  78751.0
4   233930  30.30271  -97.73123  78751.0
6   347802  30.30232  -97.72633  78751.0

                                            amenities        room_type  \
0  {TV,"Cable TV",Internet,Wifi,"Air conditioning...  Entire home/apt
2  {TV,"Cable TV",Internet,Wifi,"Air conditioning...  Entire home/apt
3  {TV,"Cable TV",Internet,Wifi,"Air conditioning...  Entire home/apt
4  {TV,"Cable TV",Internet,Wifi,"Air conditioning...  Entire home/apt
6  {TV,Internet,Wifi,"Air conditioning",Kitchen,"...  Entire home/apt

   bed_type  accommodates  bathrooms  bedrooms  ...  instant_bookable_f  \
0  Real Bed             2        1.0       1.0  ...                   1
2  Real Bed             4        1.0       2.0  ...                   1
3  Real Bed             2        1.0       1.0  ...                   1
4  Real Bed             4        1.0       2.0  ...                   1
6  Real Bed             4        1.0       2.0  ...                   1

   instant_bookable_t  cancellation_policy_flexible  \
0                   0                             0
2                   0                             0
3                   0                             0
4                   0                             0
6                   0                             0

   cancellation_policy_moderate  \
```

```
0                              0
2                              0
3                              0
4                              0
6                              0

   cancellation_policy_strict_14_with_grace_period  \
0                                                0
2                                                0
3                                                0
4                                                0
6                                                1

   cancellation_policy_super_strict_30  Review_Bad  Review_Good  \
0                                    1           0            1
2                                    1           0            1
3                                    1           0            1
4                                    1           0            1
6                                    0           0            1

  require_guest_phone_verification_f  require_guest_phone_verification_t
0                                  0                                  1
2                                  0                                  1
3                                  0                                  1
4                                  0                                  1
6                                  0                                  1

[5 rows x 34 columns]
```

[38]:
```
df_R.columns.tolist()
```

[38]:
```
['id',
 'latitude',
 'longitude',
 'zipcode',
 'amenities',
 'room_type',
 'bed_type',
 'accommodates',
 'bathrooms',
 'bedrooms',
 'beds',
 'reviews_per_month',
 'calculated_host_listings_count',
 'calculated_host_listings_count_entire_homes',
 'calculated_host_listings_count_private_rooms',
 'security_deposit',
 'cleaning_fee',
```

```
 'price',
 'property_type',
 'tv',
 'washer',
 'dryer',
 'parking',
 'laptop',
 'instant_bookable_f',
 'instant_bookable_t',
 'cancellation_policy_flexible',
 'cancellation_policy_moderate',
 'cancellation_policy_strict_14_with_grace_period',
 'cancellation_policy_super_strict_30',
 'Review_Bad',
 'Review_Good',
 'require_guest_phone_verification_f',
 'require_guest_phone_verification_t']
```

2 Advanced Visualization before Regression

[39]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

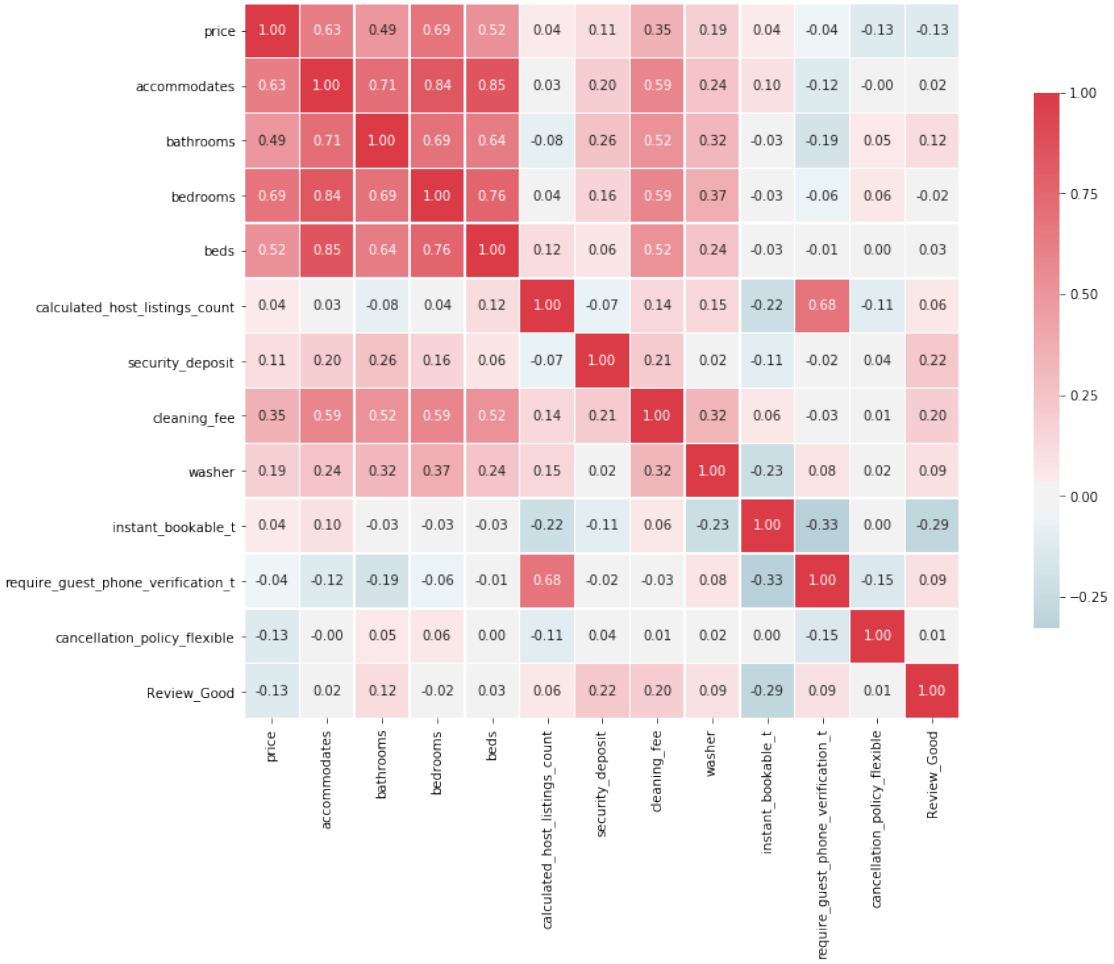
[40]:
```
def corr_heatmap(v):
    correlations = df_R[v].corr()

    # Create color map ranging between two colors
    cmap = sns.diverging_palette(220, 10, as_cmap=True)

    fig, ax = plt.subplots(figsize=(20,10))
    sns.heatmap(correlations, cmap=cmap, vmax=1.0, center=0, fmt='.2f',
                square=True, linewidths=.5, annot=True, cbar_kws={"shrink": .
 ↪75})
    plt.show();

v = ['price','accommodates', 'bathrooms', 'bedrooms','beds',␣
 ↪'calculated_host_listings_count', 'security_deposit',
 'cleaning_fee', 'washer','instant_bookable_t',␣
 ↪'require_guest_phone_verification_t',
 'cancellation_policy_flexible',
  'Review_Good']
#'tv',  'dryer',␣
 ↪,'calculated_host_listings_count_entire_homes','cancellation_policy_super_strict_60',
# 'calculated_host_listings_count_private_rooms', 'room_type_Entire home/apt',␣
 ↪'room_type_Private room',
```

```
# 'property_type_Apartment', 'property_type_Condominium',␣
 ↪'property_type_Hotel', 'property_type_House','bed_type_Real Bed',␣
 ↪'Review_Bad',
#'parking',␣
 ↪'laptop','cancellation_policy_moderate','cancellation_policy_strict_14_with_grace_period','
corr_heatmap(v)
```

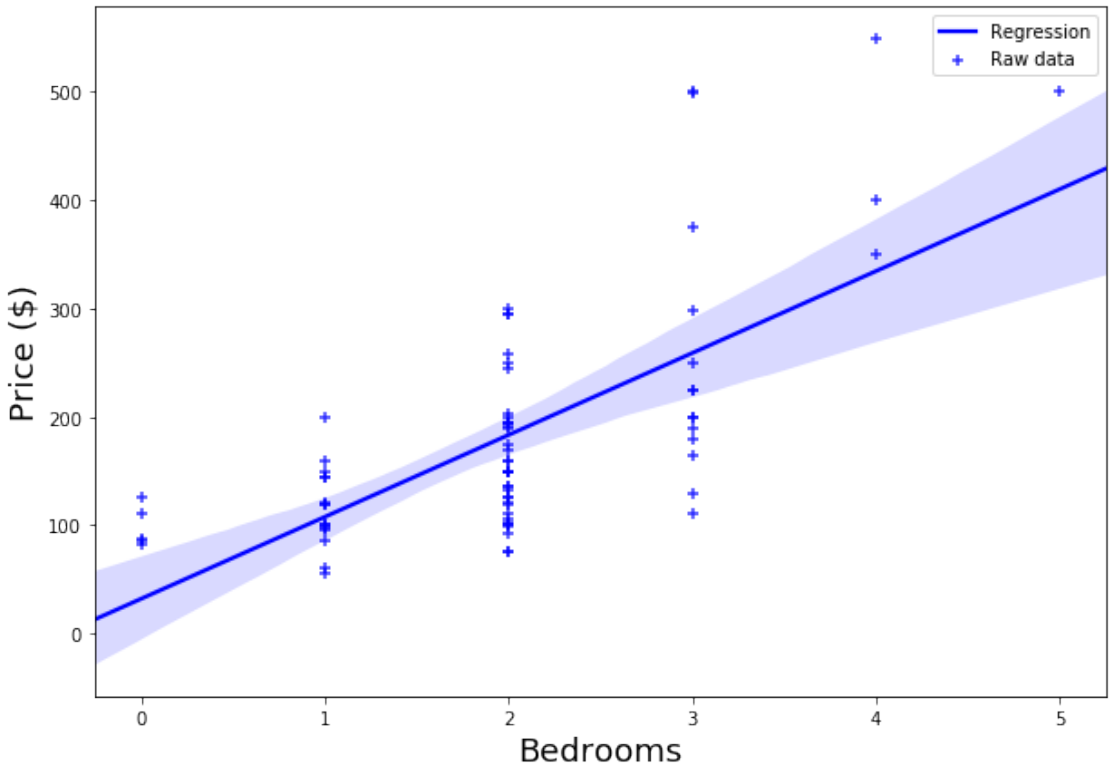


[41]:
```
# Regression plot using seaborn.
fig = plt.figure(figsize=(10,7))
sns.regplot(x = df_R.bedrooms, y = df_R.price, color='blue', marker='+')
#sns.regplot(x = df_R.calculated_host_listings_count, y = df_R.price,␣
 ↪color='magenta', marker='+')

# Legend, title and labels.
plt.legend(labels=['Regression','Raw data'])
#plt.title('Relationship between Prince and Accommodates')
plt.xlabel('Bedrooms', size=18)
```

```
plt.ylabel('Price ($)', size=18)
```

[41]:
```
Text(0, 0.5, 'Price ($)')
```



3 Regression from Statistics Perspective

[42]:
```
import statsmodels.formula.api as smf
```

[43]:
```
model = smf.ols(formula = 'price ~ bathrooms + bedrooms + beds +␣
 ↪security_deposit\
 + cleaning_fee + laptop + \
 + cancellation_policy_flexible +␣
 ↪cancellation_policy_strict_14_with_grace_period + Review_Good', data = df_R)
result_formula = model.fit()
result_formula.summary()
```

[43]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.525
Model:                            OLS   Adj. R-squared:                  0.461
```

```
Method:                 Least Squares   F-statistic:                     8.115
Date:                Thu, 08 Oct 2020   Prob (F-statistic):           5.41e-08
Time:                        20:56:06   Log-Likelihood:                -434.24
No. Observations:                  76   AIC:                             888.5
Df Residuals:                      66   BIC:                             911.8
Df Model:                           9
Covariance Type:            nonrobust
==================================================================================
                                                     coef    std err          t
P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept                                          66.9324     35.018      1.911
0.060      -2.983     136.848
laptop[T.True]                                    -14.3388     24.536     -0.584
0.561     -63.327      34.649
bathrooms                                          12.3581     22.222      0.556
0.580     -32.010      56.726
bedrooms                                           75.8539     17.367      4.368
0.000      41.179     110.528
beds                                               -0.6892     10.690     -0.064
0.949     -22.033      20.654
security_deposit                                    0.0288      0.055      0.525
0.601      -0.081       0.138
cleaning_fee                                       -0.1285      0.307     -0.419
0.677      -0.742       0.485
cancellation_policy_flexible                      -67.6647     32.034     -2.112
0.038    -131.622      -3.708
cancellation_policy_strict_14_with_grace_period   -14.3603     21.200     -0.677
0.501     -56.688      27.967
Review_Good                                       -30.9878     22.095     -1.402
0.165     -75.101      13.126
==============================================================================
Omnibus:                        8.977   Durbin-Watson:                   1.788
Prob(Omnibus):                  0.011   Jarque-Bera (JB):                8.595
Skew:                           0.729   Prob(JB):                       0.0136
Kurtosis:                       3.767   Cond. No.                     1.40e+03
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.4e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[44]:
```
model = smf.ols(formula = 'price ~ bedrooms + washer + accommodates + beds +␣
↪cancellation_policy_flexible + Review_Good', data = df_R)
result_formula = model.fit()
result_formula.summary()
```

[44]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.528
Model:                            OLS   Adj. R-squared:                  0.487
Method:                 Least Squares   F-statistic:                     12.88
Date:                Thu, 08 Oct 2020   Prob (F-statistic):           1.05e-09
Time:                        20:57:42   Log-Likelihood:                -433.99
No. Observations:                  76   AIC:                             882.0
Df Residuals:                      69   BIC:                             898.3
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
================
                                   coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       55.5779     30.328      1.833      0.071
-4.925     116.081
washer[T.True]                 -14.3357     27.388     -0.523      0.602
-68.973      40.301
bedrooms                        65.5358     18.107      3.619      0.001
29.413     101.659
accommodates                    13.0055      9.901      1.314      0.193
-6.747      32.758
beds                           -10.1579     12.352     -0.822      0.414
-34.800      14.484
cancellation_policy_flexible   -56.9252     28.856     -1.973      0.053
-114.490       0.640
Review_Good                    -27.1967     20.202     -1.346      0.183
-67.499      13.105
==============================================================================
Omnibus:                       13.654   Durbin-Watson:                   1.870
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               15.245
Skew:                           0.907   Prob(JB):                     0.000489
Kurtosis:                       4.236   Cond. No.                         25.6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
```

```
specified.
"""
```

4 Regression from Machine Learning perspective

[45]:
```
from sklearn import metrics
from sklearn.model_selection import train_test_split
```

[46]:
```
# A Simple Example
# define true and predicted response values
y_true = [100, 50, 30, 20]
y_pred = [90, 50, 50, 30]

# calculate MAE, MSE, RMSE
print(metrics.mean_absolute_error(y_true, y_pred))
print(metrics.mean_squared_error(y_true, y_pred))
print(np.sqrt(metrics.mean_squared_error(y_true, y_pred)))
```

```
10.0
150.0
12.24744871391589
```

[47]:
```
from sklearn.linear_model import LinearRegression
```

[48]:
```
# Out-of-Sample Evaluation
X = df_R[['bedrooms', 'accommodates', 'washer', 'beds',␣
 ↪'cancellation_policy_flexible', 'Review_Good']]
y = df_R.price

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)

# Instantiate model
lm2 = LinearRegression()

# Fit Model
lm2.fit(X_train, y_train)

# Predict
y_pred = lm2.predict(X_test)

# RMSE
print(np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
97.22236627049256
```

[49]:
```
# Out-of-Sample Evaluation, exclude 'Beds'
X = df_R[['bedrooms', 'washer','accommodates', 'cancellation_policy_flexible',␣
 ↪'Review_Good']]
y = df_R.price

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)

# Instantiate model
lm2 = LinearRegression()

# Fit Model
lm2.fit(X_train, y_train)

# Predict
y_pred = lm2.predict(X_test)

# RMSE
print(np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
93.07781085263811
```

[ ]:
```

```