ADS

Assignment 2

In [27]:

```
# General libraries
import pandas as pd
import numpy as np

# Visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Modelling libraries

## Data splitting
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

## Models
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor
from sklearn.ensemble import RandomForestClassifier, RandomForestRegresso
from sklearn.svm import SVC, SVR
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5)

# Import metrics

## Classification metrics
from sklearn.metrics import classification_report

## Regression metrics
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_s
```

In [28]:

```
# Import the csv file
RealEstate = pd.read_csv('RealEstate.csv')
```

In [29]:

```
RealEstate.head()
```

Out[29]:

	Price	sqft	beds	bath	age	stories	vacant
0	163.00	2727	5	3	8	2	0
1	88.00	1069	3	2	5	1	0
2	165.00	2846	4	3	56	2	0
3	150.00	1790	4	3	36	2	0
4	159.95	2200	4	2	28	1	0

In [30]:

```
RealEstate.describe()
```

Out[30]:

	Price	sqft	beds	bath	age	stories
count	550.000000	550.000000	550.000000	550.000000	550.000000	550.000000
mean	113.234859	1614.318182	3.196364	2.067273	24.105455	1.216364
std	50.694375	527.829782	0.691864	0.672801	19.769523	0.412140

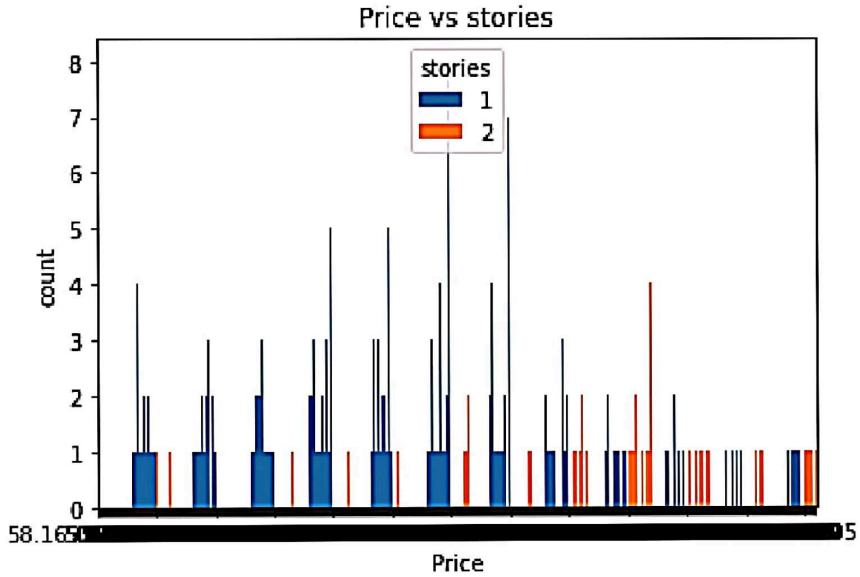
min	50.005000	751.000000	2.000000	1.000000	0.000000	1.000000
25%	80.428500	1196.250000	3.000000	2.000000	9.000000	1.000000
50%	101.596500	1518.500000	3.000000	2.000000	18.000000	1.000000
75%	130.000000	1900.000000	4.000000	2.000000	38.000000	1.000000
max	460.000000	4278.000000	6.000000	5.000000	96.000000	2.000000

In [31]:
```
RealEstate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550 entries, 0 to 549
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Price    550 non-null    float64
 1   sqft     550 non-null    int64  
 2   beds     550 non-null    int64  
 3   bath     550 non-null    int64  
 4   age      550 non-null    int64  
 5   stories  550 non-null    int64  
 6   vacant   550 non-null    int64  
dtypes: float64(1), int64(6)
memory usage: 30.2 KB
```

In [32]:
```
#How is the price related to stories
sns.countplot(x='Price' , hue = 'stories', data = RealEstate).set(title='
```

Out[32]: [Text(0.5, 1.0, 'Price vs stories')]

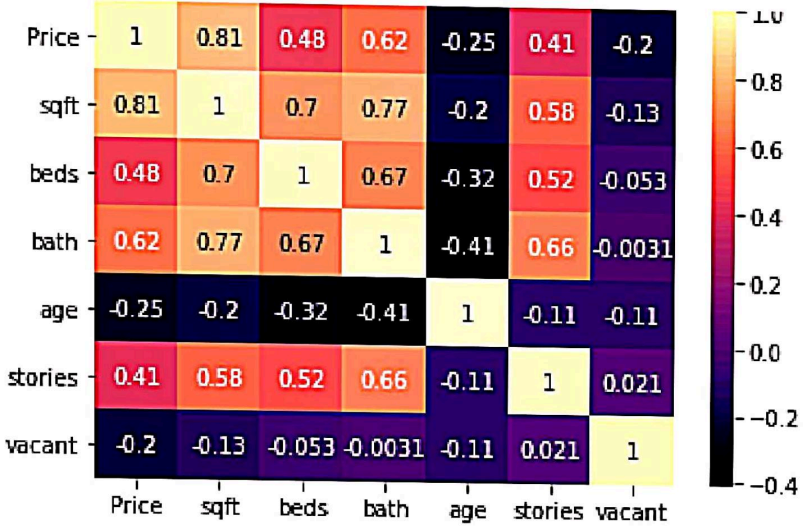


In [33]:
```
# Heatmap

# Corr amtrix
corr_matrix = RealEstate.corr()

# Plot heatmap
sns.heatmap(corr_matrix, cmap='magma', annot=True)
```

Out[33]: <AxesSubplot:>



In [34]:
```
RealEstate.duplicated().sum()
```

Out[34]: 3

In [35]:
```
RealEstate.isnull().sum()
```

Out[35]:
```
Price      0
sqft       0
beds       0
bath       0
age        0
stories    0
vacant     0
dtype: int64
```

In [36]:
```
#Feature engineering
ReslEstate_drop=["vacant"]
RealEstate.drop(ReslEstate_drop, axis=1, inplace=True)
```

In [37]:
```
RealEstate.head()
```

Out[37]:

	Price	sqft	beds	bath	age	stories
0	163.00	2727	5	3	8	2
1	88.00	1069	3	2	5	1
2	165.00	2846	4	3	56	2
3	150.00	1790	4	3	36	2
4	159.95	2200	4	2	28	1

In [38]:
```
RealEstate.duplicated().sum()
```

Out[38]: 3

In [39]:
```
RealEstate.drop_duplicates(subset=None, keep=False,inplace=False)
```

Out[39]:

	Price	sqft	beds	bath	age	stories

0	163.0000	2727	5	3	8	2
1	88.0000	1069	3	2	5	1
2	165.0000	2846	4	3	56	2
3	150.0000	1790	4	3	36	2
4	159.9500	2200	4	2	28	1
...	...	...	...	...	...	...
545	160.3950	2559	5	3	4	2
546	324.6195	2891	4	4	5	2
547	102.6750	1342	4	2	49	1
548	63.2700	1122	3	1	48	1
549	120.9900	1577	4	2	7	2

545 rows × 6 columns

In [40]:
```
RealEstate.duplicated().sum()
```

Out[40]: 3