In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split, cross_val_score,GridSearch
import plotly.express as px
import plotly.graph_objects as go
from sklearn.linear_model import Ridge
```

In [2]:
```
df=pd.read_csv("C:/Users/Qureshi/OneDrive/Desktop/Housing.csv")
```

In [3]:
```
df
```

Out[3]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheat
0	13300000	7420	4	2	3	yes	no	no	
1	12250000	8960	4	4	4	yes	no	no	
2	12250000	9960	3	2	2	yes	no	yes	
3	12215000	7500	4	2	2	yes	no	yes	
4	11410000	7420	4	1	2	yes	yes	yes	
...	...	...	...	...	...	...	...	...	
540	1820000	3000	2	1	1	yes	no	yes	
541	1767150	2400	3	1	1	no	no	no	
542	1750000	3620	2	1	1	yes	no	no	
543	1750000	2910	3	1	1	no	no	no	
544	1750000	3850	3	1	2	yes	no	no	

545 rows × 13 columns

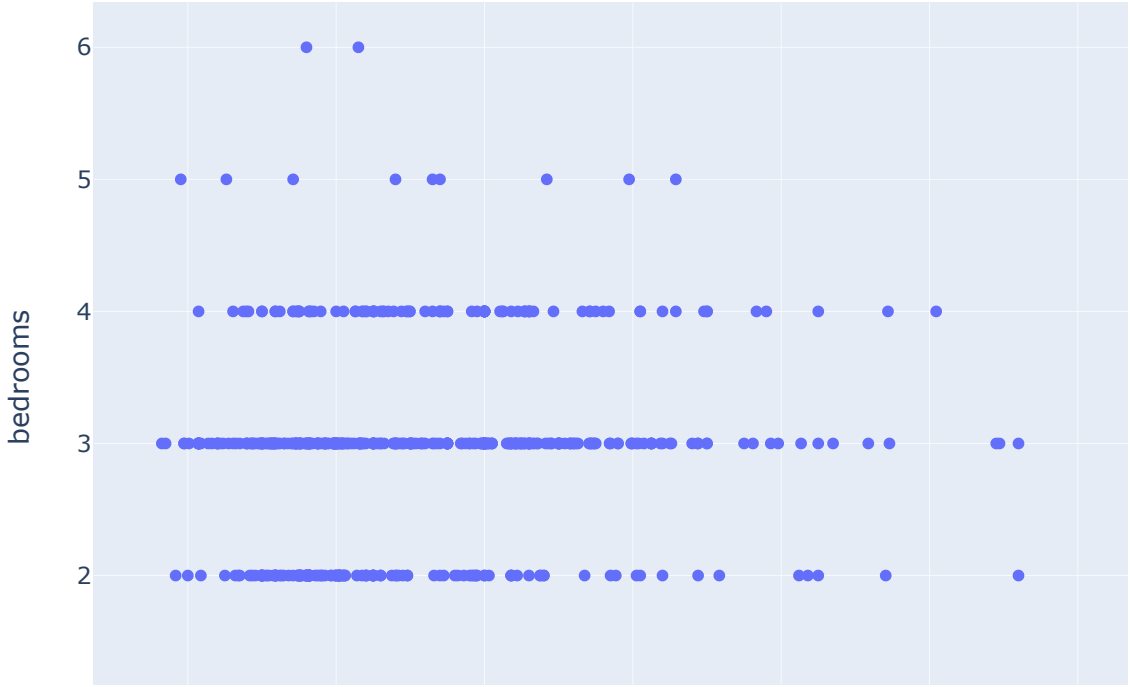
In [4]:
```
df.isnull().sum()
```

Out[4]:
```
price             0
area              0
bedrooms          0
bathrooms         0
stories           0
mainroad          0
guestroom         0
basement          0
hotwaterheating   0
airconditioning   0
parking           0
prefarea          0
furnishingstatus  0
dtype: int64
```

In [5]:
```
#duplicate values
duplicates= df.duplicated().sum()
duplicates
#no duplicates row
```

Out[5]: 0

In [6]:
```
px.scatter(df,x = 'area', y='bedrooms')
```



In [7]:
```
#categorical to integers
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [8]:
```
df['area']=le.fit_transform(df['area'])
df['bedrooms']=le.fit_transform(df['bedrooms'])
df['bathrooms']=le.fit_transform(df['bathrooms'])
df['mainroad']=le.fit_transform(df['mainroad'])
df['basement']=le.fit_transform(df['basement'])
df['parking']=le.fit_transform(df['parking'])
```

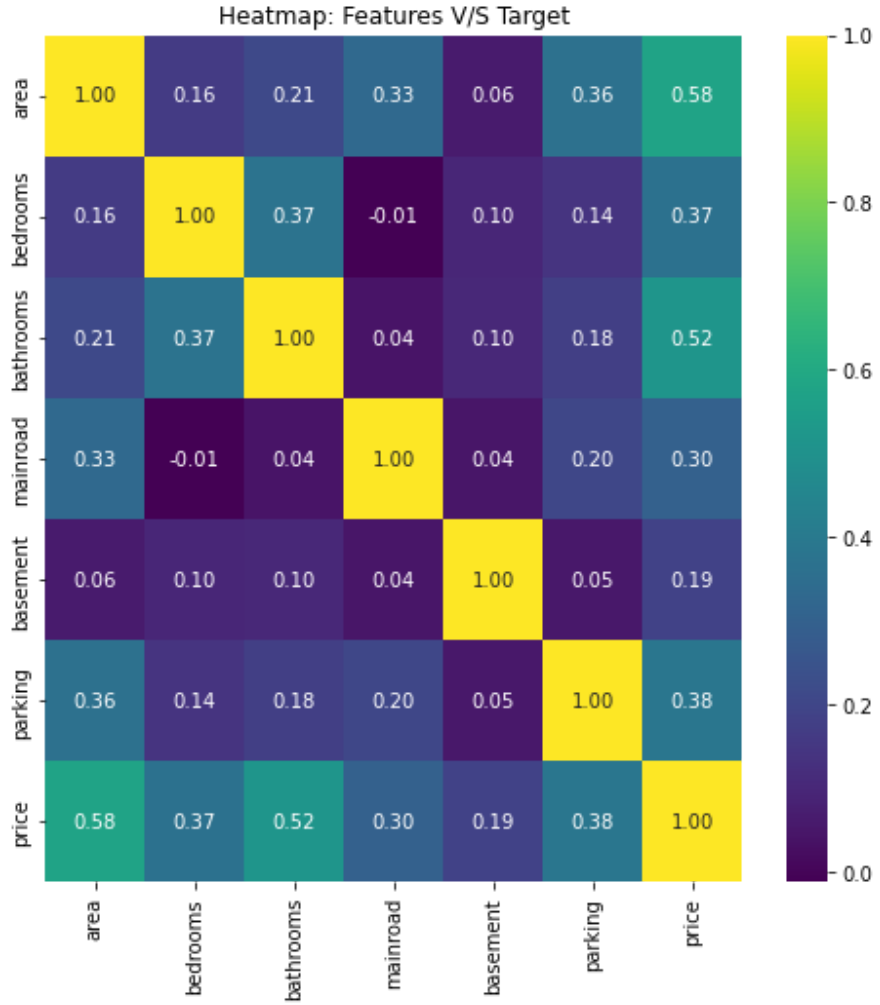
In [9]:
```
x = df[['area','bedrooms','bathrooms','mainroad','basement','parking']]
y = df[['price']]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42
features = df[['area','bedrooms','bathrooms','mainroad','basement','parking']]
target = df[['price']]
```

In [10]:
```
# Create a new DataFrame with only the selected columns
data_new= pd.concat([features, target], axis=1)
corr_matrix = data_new.corr()
corr_matrix
```

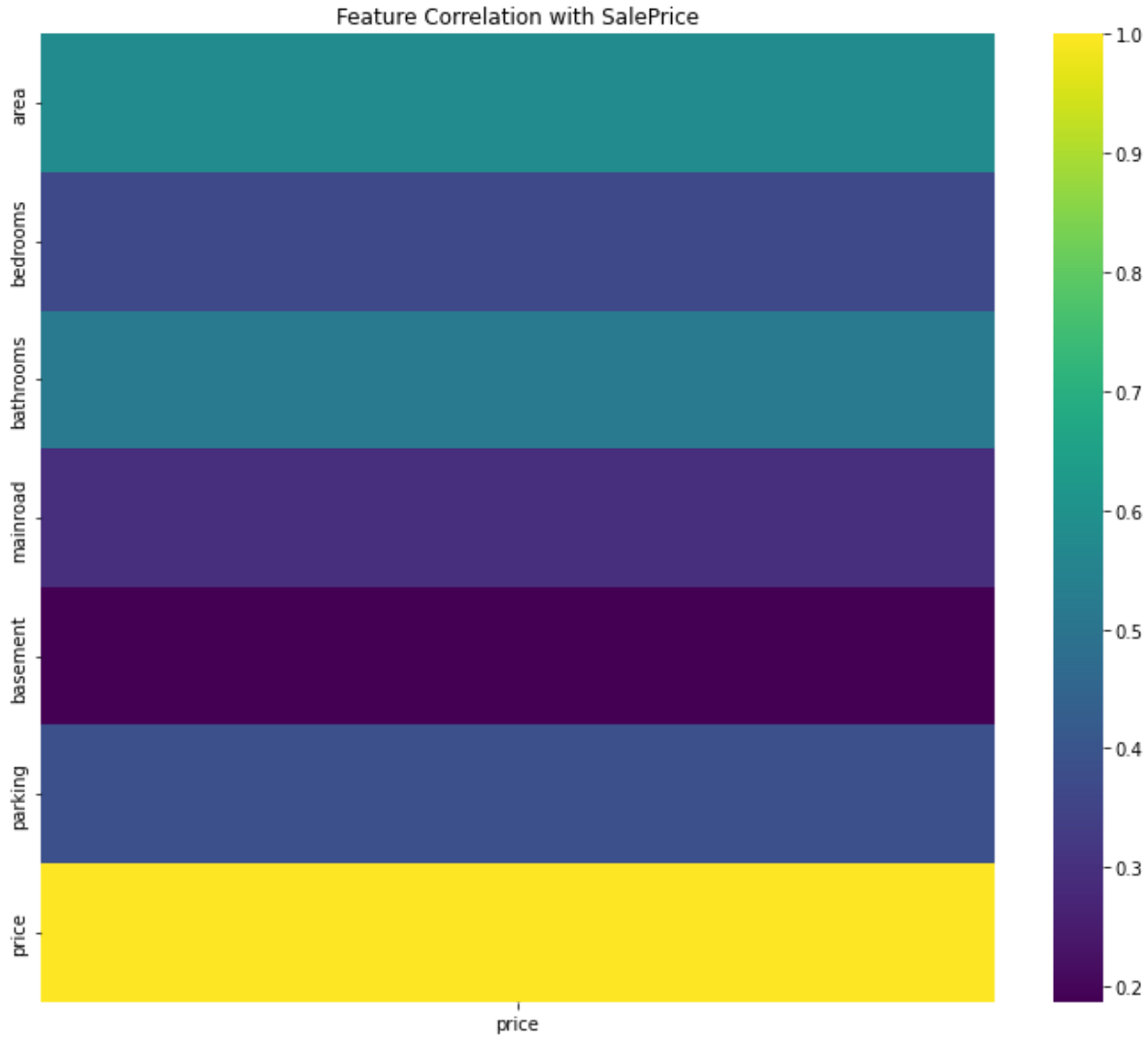
Out[10]:

	area	bedrooms	bathrooms	mainroad	basement	parking	price
area	1.000000	0.163235	0.211224	0.329097	0.060676	0.360925	0.575107
bedrooms	0.163235	1.000000	0.373930	-0.012033	0.097312	0.139270	0.366494
bathrooms	0.211224	0.373930	1.000000	0.042398	0.102106	0.177496	0.517545
mainroad	0.329097	-0.012033	0.042398	1.000000	0.044002	0.204433	0.296898
basement	0.060676	0.097312	0.102106	0.044002	1.000000	0.051497	0.187057
parking	0.360925	0.139270	0.177496	0.204433	0.051497	1.000000	0.384394
price	0.575107	0.366494	0.517545	0.296898	0.187057	0.384394	1.000000

In [11]:
```
plt.figure(figsize=(8, 8))
sns.heatmap(corr_matrix, annot=True, cmap='viridis', fmt=".2f")
plt.title("Heatmap: Features V/S Target")
plt.show()
```



In [12]:
```
#heatmaps
plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix[['price']], cmap='viridis')
plt.title("Feature Correlation with SalePrice")
plt.show()
```



In [13]:
```
#create a linear regression model
model = LinearRegression()
# Fit the model to the training data
model.fit(x_train, y_train)
```

Out[13]:
```
LinearRegression()
```

In [14]:
```
# Make predictions on the test data
y_pred = model.predict(x_test)
model.score(x_train,y_train)
```

Out[14]:
```
0.5763925019994289
```

In [15]:
```
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [16]:
```
print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared: {r2:.2f}")
```

```
Mean Squared Error: 2498570953172.19
R-squared: 0.51
```

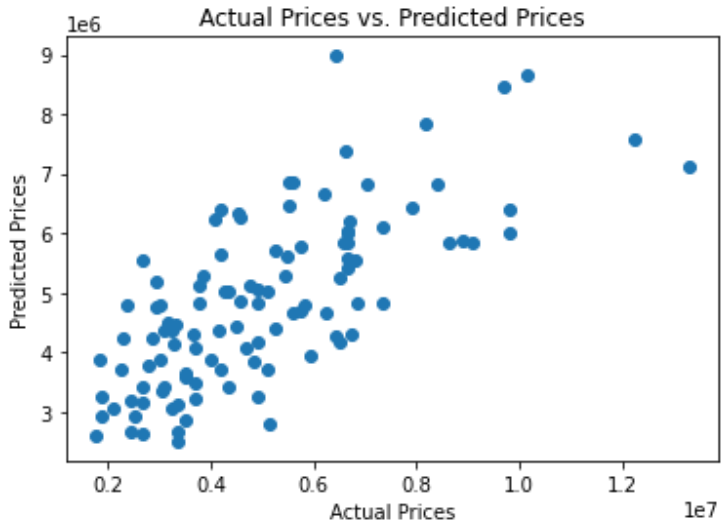
In [17]:
```
model.predict([[7420,4,2,0,1,0]])
```

```
C:\Users\Qureshi\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning:

X does not have valid feature names, but LinearRegression was fitted with featu
re names
```

Out[17]:
```
array([[72871543.50583129]])
```

In [18]:
```
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual Prices vs. Predicted Prices")
plt.show()
```



In [ ]:
```

```