King County House Sales – Feature Engineering & Price Prediction - TASK 08

In [1]:
```
# Step 1: Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
```

```
C:\Users\pramo\anaconda3\Lib\site-packages\pandas\core\arrays\masked.py:60: Use
rWarning: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.
3.5' currently installed).
  from pandas.core import (
```

In [2]:
```
# Step 2: Load dataset
df = pd.read_csv('kc_house_data.csv')
df.head()
```

Out[2]:

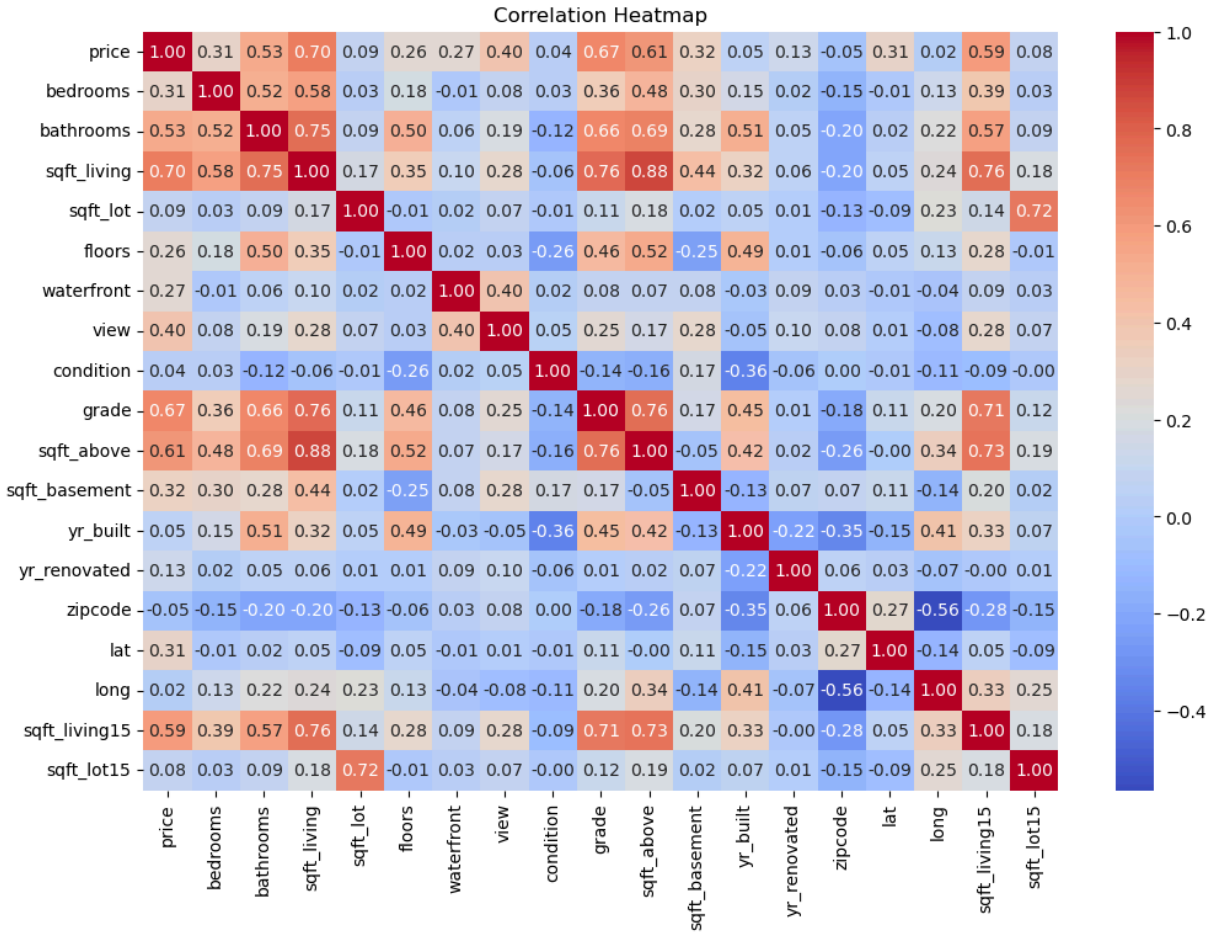
	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	wate
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	

5 rows × 21 columns

In [3]:
```
# Step 3: Drop ID and check for nulls
df = df.drop(['id', 'date'], axis=1)
df.isnull().sum()
```

Out[3]:
```
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated     0
zipcode          0
lat              0
long             0
sqft_living15    0
sqft_lot15       0
dtype: int64
```

In [4]:
```
# Step 4: Correlation heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



In [5]:
```
# Step 5: Define features and target
X = df.drop('price', axis=1)
y = df['price']
```

In [6]:
```
# Step 6: Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_s
```

In [7]:
```
# Step 7: Scale numeric features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [8]:
```
# Step 8: Train Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train_scaled, y_train)
preds = model.predict(X_test_scaled)
rmse = np.sqrt(mean_squared_error(y_test, preds))
print('Test RMSE: ${:,.2f}'.format(rmse))
```

```
Test RMSE: $148,896.95
```

In [9]:
```
# Step 9: Feature importance
importances = model.feature_importances_
indices = np.argsort(importances)[-15:]
plt.figure(figsize=(10, 6))
plt.title('Top 15 Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='skyblue')
plt.yticks(range(len(indices)), [X.columns[i] for i in indices])
plt.tight_layout()
plt.show()
```