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

In [4]:
```
# Read dataset
df = pd.read_csv('assig5/data/seattle-house-prices.csv')
df.head()
```

Out[4]:

| | price | bedrooms | bathrooms | sqft_living | sqft_lot | yr_built | lat | long |
|---|---|---|---|---|---|---|---|---|
| **0** | 538000 | 3 | 2.25 | 2570 | 7242 | 1951 | 47.7210 | -122.319 |
| **1** | 180000 | 2 | 1.00 | 770 | 10000 | 1933 | 47.7379 | -122.233 |
| **2** | 604000 | 4 | 3.00 | 1960 | 5000 | 1965 | 47.5208 | -122.393 |
| **3** | 510000 | 3 | 2.00 | 1680 | 8080 | 1987 | 47.6168 | -122.045 |
| **4** | 1230000 | 4 | 4.50 | 5420 | 101930 | 2001 | 47.6561 | -122.005 |

In [5]:
```
# Summary of columns, values, and data types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19451 entries, 0 to 19450
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   price        19451 non-null  int64  
 1   bedrooms     19451 non-null  int64  
 2   bathrooms    19451 non-null  float64
 3   sqft_living  19451 non-null  int64  
 4   sqft_lot     19451 non-null  int64  
 5   yr_built     19451 non-null  int64  
 6   lat          19451 non-null  float64
 7   long         19451 non-null  float64
dtypes: float64(3), int64(5)
memory usage: 1.2 MB
```

In [6]:
```
df.shape
```

Out[6]:
```
(19451, 8)
```

In [9]:
```
num_houses = df.shape[0]
```

In [10]:
```
num_houses
```

Out[10]:
```
19451
```

In [11]:
```
num_features = df.shape[1]
```

In [12]:
```
num_features
```

Out[12]:
```
8
```

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19451 entries, 0 to 19450
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   price        19451 non-null  int64  
 1   bedrooms     19451 non-null  int64  
 2   bathrooms    19451 non-null  float64
 3   sqft_living  19451 non-null  int64  
 4   sqft_lot     19451 non-null  int64  
 5   yr_built     19451 non-null  int64  
 6   lat          19451 non-null  float64
 7   long         19451 non-null  float64
dtypes: float64(3), int64(5)
memory usage: 1.2 MB
```

In [14]:
```
# Compute correlation matrix
corr_matrix = df.corr()

# Display just house value correlations
corr_matrix["long"].sort_values(ascending= False)
```

Out[14]:
```
long           1.000000
yr_built       0.407168
sqft_living    0.240437
sqft_lot       0.233369
bathrooms      0.221788
bedrooms       0.134575
price          0.020092
lat           -0.135695
Name: long, dtype: float64
```

In [15]:
```
# Import library
from sklearn.preprocessing import StandardScaler

# Define feature list
feature_list =  ['long', 'lat', 'yr_built',
                 'sqft_lot', 'sqft_living', 'bathrooms', 'bedrooms']

# Define features and labels
X = df[feature_list]
y = df['price']
```

In [16]:
```
# Standarize data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
df_scaled = pd.DataFrame(X_scaled, columns=feature_list)
df_scaled
```

Out[16]:

| | long | lat | yr_built | sqft_lot | sqft_living | bathrooms | bedrooms |
|---|---|---|---|---|---|---|---|
| **0** | -0.747062 | 1.161695 | -0.681874 | -0.188509 | 0.532437 | 0.175445 | -0.408062 |
| **1** | -0.135791 | 1.283471 | -1.294903 | -0.122212 | -1.428055 | -1.448357 | -1.511064 |
| **2** | -1.273039 | -0.280879 | -0.205073 | -0.242402 | -0.131952 | 1.149726 | 0.694941 |
| **3** | 1.200476 | 0.410865 | 0.544185 | -0.168365 | -0.436917 | -0.149315 | -0.408062 |
| **4** | 1.484788 | 0.694047 | 1.020985 | 2.087617 | 3.636549 | 3.098289 | 0.694941 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **19446** | -1.386764 | 0.126241 | 1.259385 | -0.331488 | -0.839907 | 0.500206 | -0.408062 |
| **19447** | -0.938973 | 1.005332 | 1.293443 | -0.335406 | -0.600292 | 0.500206 | -0.408062 |
| **19448** | -1.052697 | -0.353656 | 1.463728 | -0.222859 | 0.249255 | 0.500206 | 0.694941 |
| **19449** | 1.029889 | -0.182161 | 1.123157 | -0.305190 | -0.524050 | 0.500206 | -0.408062 |
| **19450** | -0.604906 | 0.247296 | 1.259385 | -0.336728 | -1.155764 | -1.773117 | -1.511064 |

19451 rows × 7 columns

In [17]:
```
from sklearn.model_selection import train_test_split

# Split data
X_train, X_test, y_train, y_test = train_test_split(df_scaled, y, test_size=0.2, random_state=42)
```

In [18]:
```
from sklearn.linear_model import LinearRegression

# Define model
lin_reg = LinearRegression()

# Fit model to data
lin_reg.fit(X_train, y_train)
```

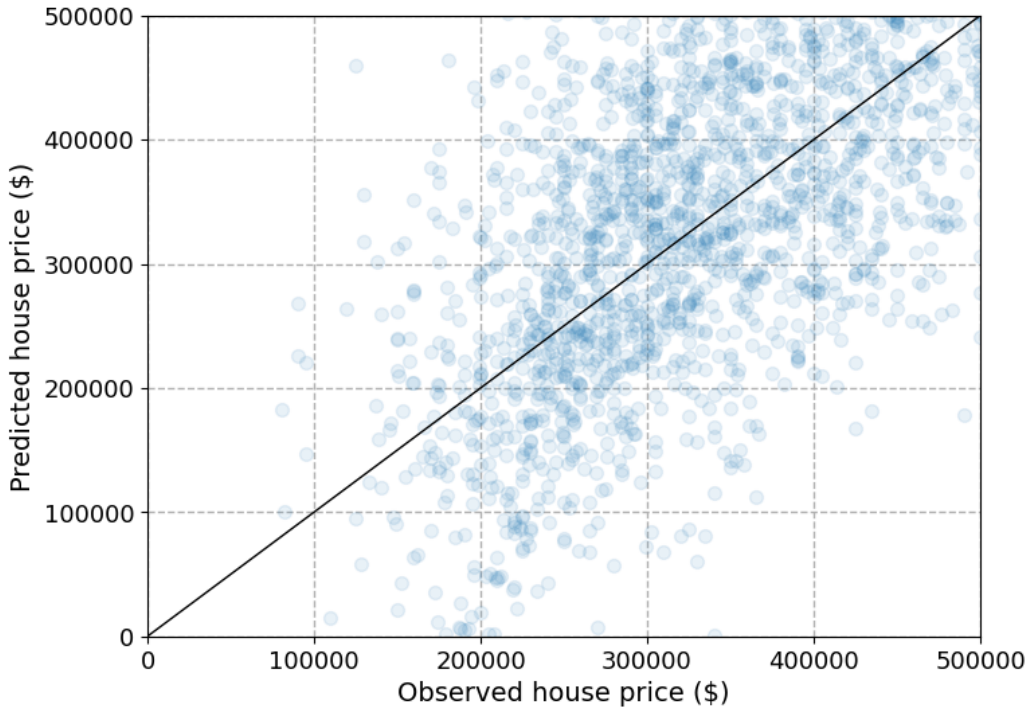
Out[18]:
```
LinearRegression()
```
**In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.**
**On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.**

In [19]:
```
from sklearn.metrics import mean_squared_error

# Predict test labels
predictions = lin_reg.predict(X_test)

# Compute mean-squared-error
lin_mse = mean_squared_error(y_test, predictions)
lin_rmse = np.sqrt(lin_mse)
lin_rmse
```

Out[19]:
```
239438.7716731389
```

In [20]:
```
# Plot
fig, ax = plt.subplots(figsize=(8, 6))
ax.scatter(y_test, predictions, alpha=0.1, s=50, zorder=2)
ax.plot([0,500000], [0, 500000], color='k', lw=1, zorder=3)
ax.set_ylabel('Predicted house price ($)', fontsize=14)
ax.set_xlabel('Observed house price ($)', fontsize=14)
ax.tick_params(axis='both', which='major', labelsize=13)
ax.grid(ls='dashed', lw=1, zorder=1)
ax.set_ylim(0,500000)
ax.set_xlim(0,500000)
```

Out[20]:
```
(0.0, 500000.0)
```




In [ ]: