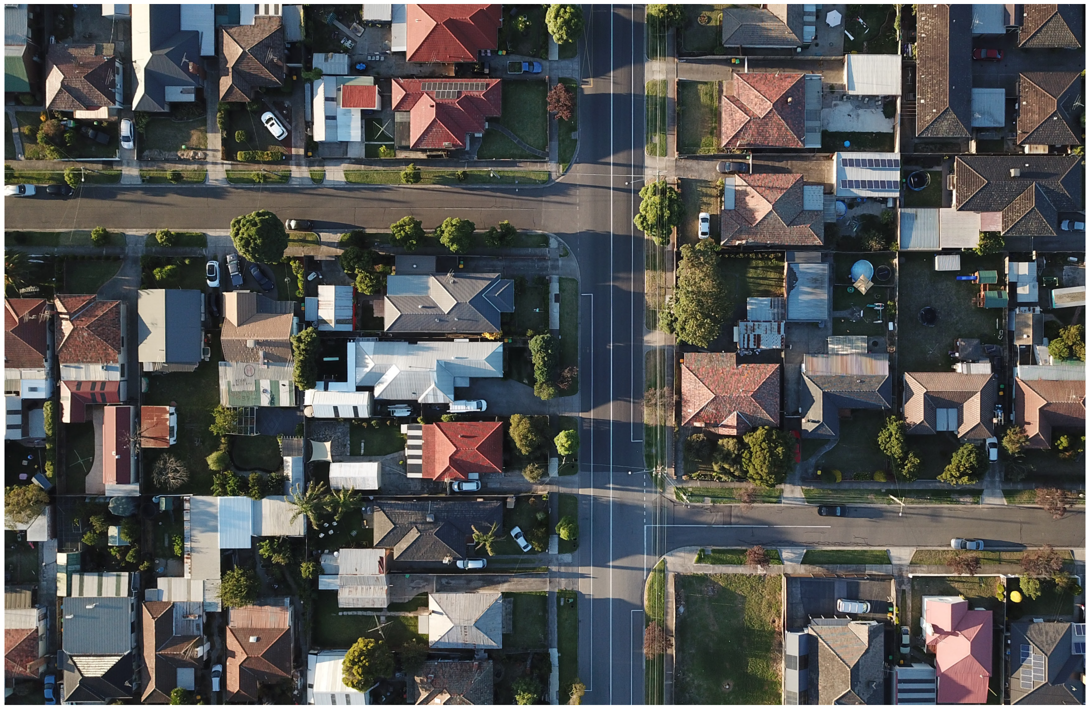


King County Housing with Multiple Linear Regression

Authors: Diane Tunnicliffe, Dana Rausch, Matthew Lipman

Notebook 1: Business Problem and Data Understading

This notebook contains an introduction to our project, our business problem, the full pr
how all our data were obtained, and an exploration of our data with EDA.

Overview

We have been tasked with analyzing the data of houses in King County. Our goal is to
predictions about the sale price of houses based on certain variables or features, so th
be used to make profitable decisions by a housing development company. After carefu
consideration and evaluation of our data, and many iterations of our linear regression
have determined that sqare-feet of living space, building grade, and proximity to top sc
coffee shops, and churches of scientology all are correlated with a higher selling price
in King County.

Business Problem

Our stakeholders in a housing development company are searching for the qualities th
higher home sale prices. We will be reviewing building grade, square-footage of living
location-related factors such as proximity to schools, coffee shops, parks, and sciento

churches to determine which factors are highly correlated with home sale prices.

Hypotheses

Null hypothesis (H0): There is no relationship between our features and our target varia
Alternative hypothesis (Ha): There is a relationship between our features and our target
price.

We will be using a significance level (alpha) of 0.05 to make our determination, and will
final recommendations accordingly.

Data Understanding

We utilized a few different data sources for our model so that we could obtain a compr
accurate prediction of home prices.

- King County House Data: a dataset that we were provided at the onset of the proje contains data for 21,597 homes built in King County from 1900 to 2015. Each hom contains information regarding features such as number of bedrooms/bathrooms, floors, square footage, zip code, condition, and more.
- Urban Institute Education Data: The Urban Institute is a nonprofit research organiz Education Data Explorer "...harmonizes data from all major federal datasets, incluc Department of Education Common Core of Data, the US Department of Education Data Collection, the US Department of Education EDFacts, the US Census Bureau Income and Poverty Estimates, the US Department of Education Integrated Posts Education Data System, the US Department of Education College Scorecard, and Historical Geographic Information System." Custom-generated report provides de such as name and location (lat,long) of school, zip code, and which school district to.
- Niche.com: school rankings for top King County school districts.
- Yelp API: Used to obtain the top-rated coffee shops for King County.
- Web-scraped data from KingCounty.gov parks website (https://www.kingcounty.gov/services/parks-recreation/parks/parks-and-natural-lands/parksatoz.aspx (https://www.kingcounty.gov/services/parks-recreation/park natural-lands/parksatoz.aspx))
- Scientology church location information from scientology-seattle.org.
- Building grade categorical descriptions from https://info.kingcounty.gov/assessor/esales/Glossary.aspx?type=r (https://info.kingcounty.gov/assessor/esales/Glossary.aspx?type=r).

In [227]:
```
importing the packages we will be using for this project
port pandas as pd
setting pandas display to avoid scientific notation in my datafra
.options.display.float_format = '{:.2f}'.format
port numpy as np
port matplotlib.pyplot as plt
port seaborn as sns
port sklearn

om bs4 import BeautifulSoup
port json
port requests

port folium

port haversine as hs

port statsmodels.api as sm
om statsmodels.formula.api import ols
om statsmodels.stats import diagnostic as diag
om statsmodels.stats.outliers_influence import variance_inflation

om sklearn.metrics import r2_score
om sklearn.linear_model import LinearRegression
om sklearn.neighbors import NearestNeighbors
om sklearn.model_selection import train_test_split
om sklearn.metrics import mean_squared_error, r2_score, mean_abso

port scipy.stats as stats

port pylab

atplotlib inline
```

King County House Data

In [228]:
```
# reading the csv file
df = pd.read_csv('data/kc_house_data.csv')
# previewing the DataFrame
df.head()
```

Out[228]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floo
0	7129300520	10/13/2014	221900.00	3	1.00	1180	5650	1.(
1	6414100192	12/9/2014	538000.00	3	2.25	2570	7242	2.(
2	5631500400	2/25/2015	180000.00	2	1.00	770	10000	1.(
3	2487200875	12/9/2014	604000.00	4	3.00	1960	5000	1.(
4	1954400510	2/18/2015	510000.00	3	2.00	1680	8080	1.(

5 rows × 21 columns

In [229]:
```
# getting info for DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21597 entries, 0 to 21596
Data columns (total 21 columns):
id                21597 non-null int64
date              21597 non-null object
price             21597 non-null float64
bedrooms          21597 non-null int64
bathrooms         21597 non-null float64
sqft_living       21597 non-null int64
sqft_lot          21597 non-null int64
floors            21597 non-null float64
waterfront        19221 non-null float64
view              21534 non-null float64
condition         21597 non-null int64
grade             21597 non-null int64
sqft_above        21597 non-null int64
sqft_basement     21597 non-null object
yr_built          21597 non-null int64
yr_renovated      17755 non-null float64
zipcode           21597 non-null int64
lat               21597 non-null float64
long              21597 non-null float64
sqft_living15     21597 non-null int64
sqft_lot15        21597 non-null int64
dtypes: float64(8), int64(11), object(2)
memory usage: 3.5+ MB
```

In [230]:
```
df.shape
```

Out[230]:
```
(21597, 21)
```

In [231]:
```
df.price.describe()
```

Out[231]:
```
count      21597.00
mean      540296.57
std       367368.14
min        78000.00
25%       322000.00
50%       450000.00
75%       645000.00
max      7700000.00
Name: price, dtype: float64
```

This dataset contains a wide price range for houses from 78,000 dollars all the way up
million dollars. The mean house price is 540,297 dollars, while the median house price
dollars.

In [232]:
```python
# checking the dispersion of years built
df.yr_built.describe()
```

Out[232]:
```
count    21597.00
mean      1971.00
std         29.38
min       1900.00
25%       1951.00
50%       1975.00
75%       1997.00
max       2015.00
Name: yr_built, dtype: float64
```

In [233]:
```python
# getting counts for each value in condition column
df['condition'].value_counts()
```

Out[233]:
```
3    14020
4     5677
5     1701
2      170
1       29
Name: condition, dtype: int64
```

In [234]:
```python
# getting counts for each value in zipcode column
df['zipcode'].value_counts()
```

Out[234]:
```
98103    602
98038    589
98115    583
98052    574
98117    553
        ...
98102    104
98010    100
98024     80
98148     57
98039     50
Name: zipcode, Length: 70, dtype: int64
```

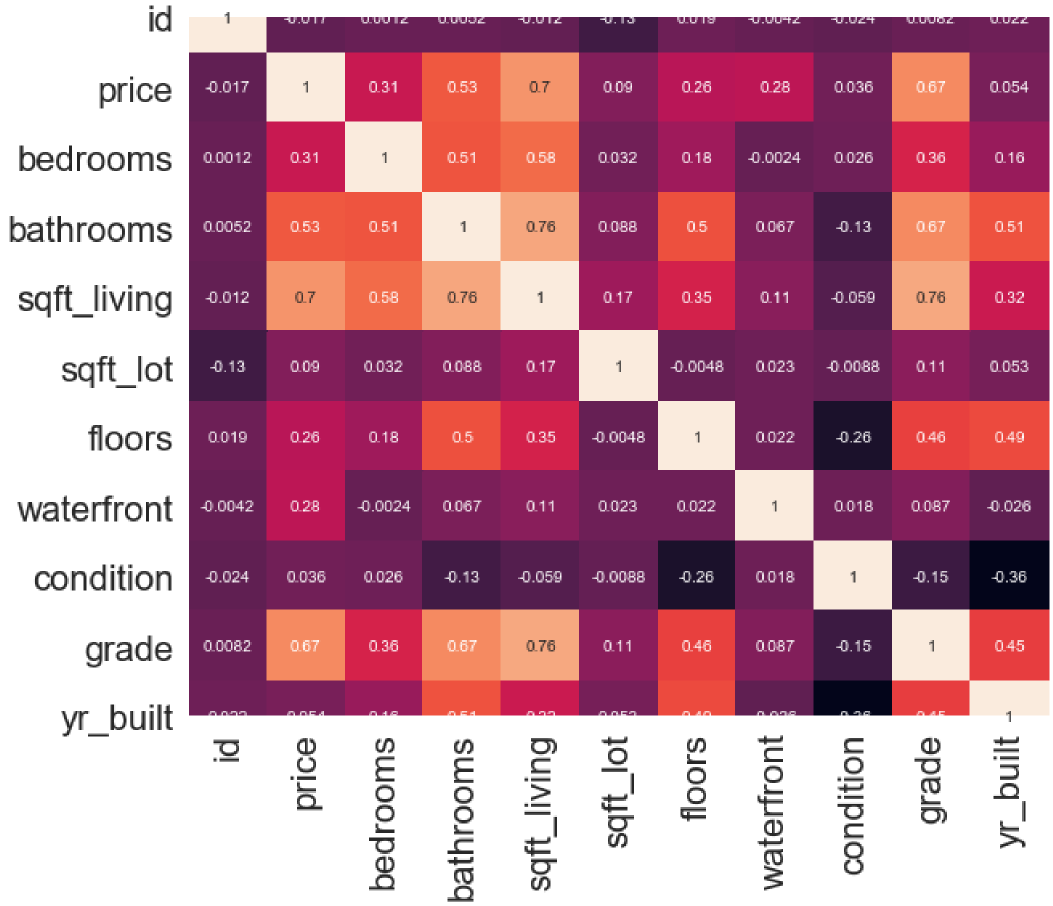
In [235]:
```python
# getting descriptive statistics for square footage
df['sqft_living'].describe()
```

Out[235]:
```
count    21597.00
mean      2080.32
std        918.11
min        370.00
25%       1430.00
50%       1910.00
75%       2550.00
max      13540.00
Name: sqft_living, dtype: float64
```

The mean square-feet of living space is 2,080 square feet, but there are houses as sma
and as large as 13,540 sqft in this dataset.

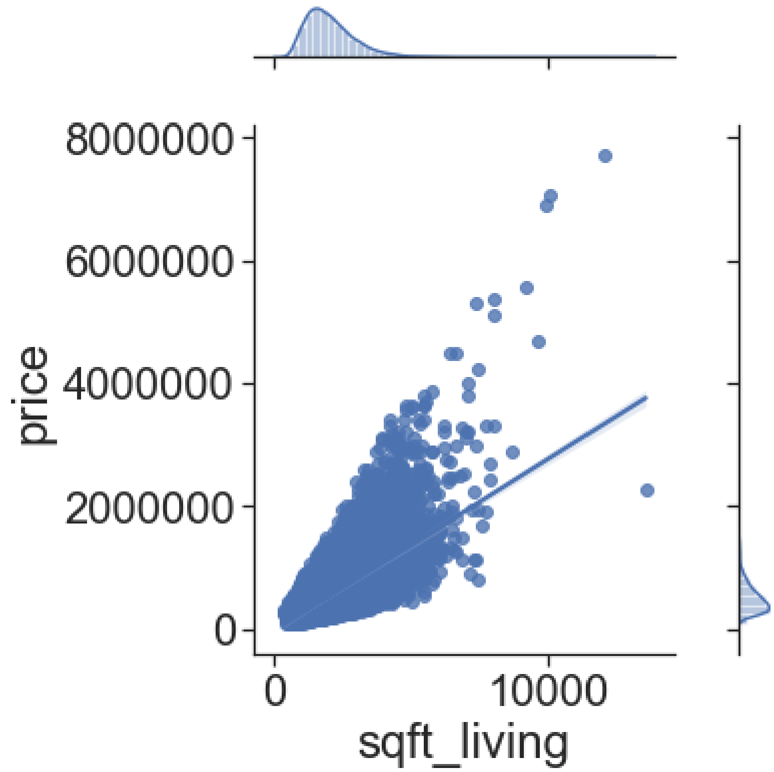
In [236]:
```
# remove unwanted columns
drop_vars = ['date', 'view', 'sqft_above', 'sqft_basement', 'yr_
             'zipcode', 'lat', 'long', 'sqft_living15', 'sqft_lo
df_corr = df.drop(columns=drop_vars)

# generate heatmap to display correlations
corr = df_corr.corr()
f, ax = plt.subplots(figsize=(12, 8))
sns.heatmap(corr, annot=True);
```



In [334]:

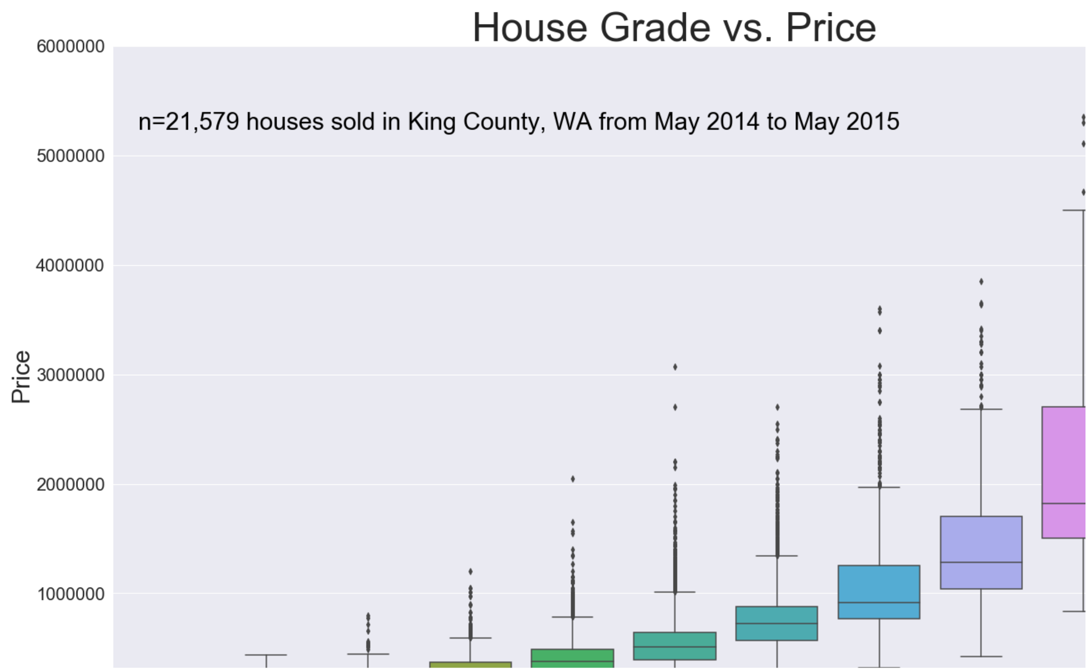
```
# examining the relationship between sqft_living and price
sns.jointplot('sqft_living','price', data=df, kind='reg')
plt.tight_layout()
plt.savefig('./visualizations/sqft_reg.png');
```



The visualization above demonstrates that there seems to be a relatively strong linear r
between square feet of living space and the price of a house.

In [238]:

```
#grade
plt.figure(figsize=(25,15))
sns.set(font_scale=2)
pal = sns.color_palette("husl", 8)
ax = sns.boxplot(x="grade", y="price", data=df_train)
ax.set_title('House Grade vs. Price', fontsize=50)
ax.set_ylabel('Price', fontsize=30)
ax.set_xlabel('Grade', fontsize=30)
ax.set_ylim(bottom=0, top=6000000)
ax.text(.7, .9, 'n=21,579 houses sold in King County, WA from May
        color='black', fontsize=30,
        horizontalalignment='right',
        verticalalignment='top',
        transform=ax.transAxes);
plt.savefig('./visualizations/grade.png');
```



When we look at grade, we can see that as the categorical building grade designation the house price does indeed rise as well. This makes sense, as the definition for a build 13 is, "Generally custom designed and built. Mansion level. Large amount of highest q work, wood trim, marble, entry ways etc." We can see in the boxplots above that the m price for a home with a grade of 13 is far above even the max value for any other grade the definition of a building grade of 3 is, "Falls short of minimum building standards. No or inferior structure." We can see this clearly demonstrated in the selling prices of hous lower end of grade.

In [239]:
```
df.grade.value_counts()
```
Out[239]:
```
7     8974
8     6065
9     2615
6     2038
10    1134
11     399
5      242
12      89
4       27
13      13
3        1
Name: grade, dtype: int64
```

The most common building grade is a 7, which is defined as, "Average grade of constr
design."

Urban Institute Education Data

King County Schools

We began by calculcating the distance from each home to a school, to see if there was
connection between school proximity and house price.

In [240]:
```
# loading and previewing school data
schools = pd.read_csv('data/EducationDataPortal_11.22.2020_schoo
schools.head()
```
Out[240]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude	lo
0	2015	530000100376	Black Diamond Elementary	Washington	Enumclaw School District	98010	47.31	
1	2015	530000100377	Byron Kibler Elementary School	Washington	Enumclaw School District	98022	47.21	
2	2015	530000100379	Enumclaw Sr High School	Washington	Enumclaw School District	98022	47.19	
3	2015	530000100382	Southwood Elementary School	Washington	Enumclaw School District	98022	47.19	
4	2015	530000100383	Westwood Elementary School	Washington	Enumclaw School District	98022	47.23	

In [241]:
```
# getting value counts for school county codes
schools.county_code.value_counts()
```

Out[241]:
```
53033.00    518
53053.00    284
53061.00    223
53063.00    175
53011.00    135
53077.00    113
53035.00     80
53067.00     79
53073.00     69
53005.00     61
53025.00     55
53015.00     48
53057.00     48
53041.00     46
53065.00     42
53027.00     41
53007.00     39
53021.00     36
53047.00     33
53071.00     30
53009.00     29
53029.00     26
53075.00     26
53039.00     22
53045.00     22
53017.00     21
53037.00     20
53049.00     20
53043.00     16
53031.00     15
53001.00     15
53055.00     14
53019.00     12
53003.00     12
53059.00     11
53051.00      9
53013.00      4
53023.00      2
53069.00      2
Name: county_code, dtype: int64
```

In [242]:
```python
# filtering dataframe to show only King County schools
# King County's county code is 53033 as per county website
schools = schools.loc[schools['county_code']==53033]
schools.head()
```

Out[242]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude	lo
0	2015	530000100376	Black Diamond Elementary	Washington	Enumclaw School District	98010	47.31	
1	2015	530000100377	Byron Kibler Elementary School	Washington	Enumclaw School District	98022	47.21	
2	2015	530000100379	Enumclaw Sr High School	Washington	Enumclaw School District	98022	47.19	
3	2015	530000100382	Southwood Elementary School	Washington	Enumclaw School District	98022	47.19	
4	2015	530000100383	Westwood Elementary School	Washington	Enumclaw School District	98022	47.23	

In [243]:
```python
schools.shape
```

Out[243]: (518, 11)

In [244]:
```python
# resetting index after filtering
schools.reset_index(inplace=True)
schools.head()
```

Out[244]:

	index	year	ncessch	school_name	state_name	lea_name	zip_location	latitu
0	0	2015	530000100376	Black Diamond Elementary	Washington	Enumclaw School District	98010	47
1	1	2015	530000100377	Byron Kibler Elementary School	Washington	Enumclaw School District	98022	47
2	2	2015	530000100379	Enumclaw Sr High School	Washington	Enumclaw School District	98022	47
3	3	2015	530000100382	Southwood Elementary School	Washington	Enumclaw School District	98022	47
4	4	2015	530000100383	Westwood Elementary School	Washington	Enumclaw School District	98022	47

In [245]:
```
# dropping extra index column
schools.drop(columns='index', inplace=True, axis=1)
```

In [246]:
```
schools.head()
```

Out[246]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude	lo
0	2015	530000100376	Black Diamond Elementary	Washington	Enumclaw School District	98010	47.31	
1	2015	530000100377	Byron Kibler Elementary School	Washington	Enumclaw School District	98022	47.21	
2	2015	530000100379	Enumclaw Sr High School	Washington	Enumclaw School District	98022	47.19	
3	2015	530000100382	Southwood Elementary School	Washington	Enumclaw School District	98022	47.19	
4	2015	530000100383	Westwood Elementary School	Washington	Enumclaw School District	98022	47.23	

In [247]:
```
# checking for duplicates
schools.school_name.duplicated().sum()
```

Out[247]: 11

In [248]:
```
# showing duplicates for school name
schools.loc[schools.school_name.duplicated()==True]
```

Out[248]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude
28	2015	530030002904	Special Ed School	Washington	Auburn School District	98002	47.31
123	2015	530354000522	Cascade Middle School	Washington	Highline School District	98146	47.50
125	2015	530354000524	Chinook Middle School	Washington	Highline School District	98188	47.44
160	2015	530354003373	Gateway to College	Washington	Highline School District	98146	47.50
203	2015	530396000628	Panther Lake Elementary School	Washington	Kent School District	98031	47.41
321	2015	530591001993	Sunrise Elementary	Washington	Northshore School District	98052	47.73
333	2015	530723001071	Hazelwood Elementary School	Washington	Renton School District	98056	47.54
337	2015	530723001076	Lakeridge Elementary School	Washington	Renton School District	98178	47.50
411	2015	530771001229	Olympic View Elementary School	Washington	Seattle Public Schools	98115	47.70
456	2015	530771003361	Rainier View Elementary School	Washington	Seattle Public Schools	98178	47.50
482	2015	530792003445	Head Start	Washington	Shoreline School District	98133	47.75

In [249]:
```
# reviewing duplicates
schools.loc[schools.school_name=='Panther Lake Elementary School
```

Out[249]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude
99	2015	530282001767	Panther Lake Elementary School	Washington	Federal Way School District	98003	47.29
203	2015	530396000628	Panther Lake Elementary School	Washington	Kent School District	98031	47.41

In [250]:
```
schools.loc[schools.school_name=='Cascade Middle School']
```

Out[250]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude
12	2015	530030000033	Cascade Middle School	Washington	Auburn School District	98002	47.33
123	2015	530354000522	Cascade Middle School	Washington	Highline School District	98146	47.50

In [251]:
```
schools.loc[schools.school_name=='Sunrise Elementary']
```

Out[251]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude
5	2015	530000100478	Sunrise Elementary	Washington	Enumclaw School District	98022	47.19
321	2015	530591001993	Sunrise Elementary	Washington	Northshore School District	98052	47.73

When reviewing the 11 duplicates for "school_name", it was apparent that these were
entries, but rather, different institutions with the same name in different school districts

```
In [252]: # checking for null values
          schools.isnull().sum()
```

```
Out[252]: year            0
          ncessch         0
          school_name     0
          state_name      0
          lea_name        0
          zip_location    0
          latitude        0
          longitude       0
          county_code     0
          school_level    0
          school_type     0
          dtype: int64
```

```
In [253]: school_coordinates = []
          x = round(schools.latitude, 2)
          y = round(schools.longitude, 2)
          school_coordinates = list(zip(x,y))
```

```
In [254]: def distance_to(point_of_interest):
              """
              Calculates distance between point of interest and a house.

              Takes in coordinates for point of interest as latitude and lo
              Calculates distance from each point in dataframe (df) to poin
              Uses haversine formula to calculate distance and return as ki
              Can set distances as new column of dataframe by using df['new

              Parameters:
              point_of_interest (float): user input coordinates (latitude,l

              Returns:
              Distances in kilometers, using haversine formula.

              """
              distance = df[['lat','long']].apply(lambda x: hs.haversine(x
              return distance
```

```
In [255]: for i in range(len(school_coordinates)):
              df[f'school_{i}'] = distance_to(school_coordinates[i])

          school_cols = []
          for i in range(len(school_coordinates)):
              school_cols.append(f'school_{i}')
              df['closest_distance_to_school'] = df[school_cols].min(axis=1
```

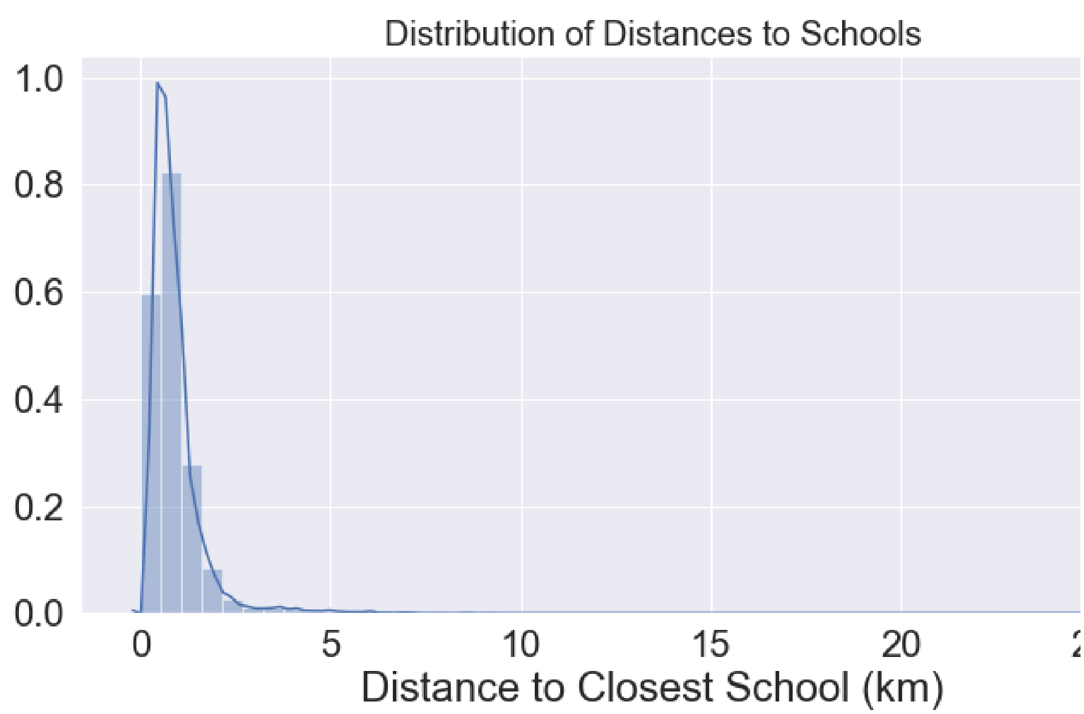
In [256]:
```
df.closest_distance_to_school.describe()
```

Out[256]:
```
count    21597.00
mean         0.88
std          0.77
min          0.00
25%          0.47
50%          0.71
75%          1.06
max         26.95
Name: closest_distance_to_school, dtype: float64
```

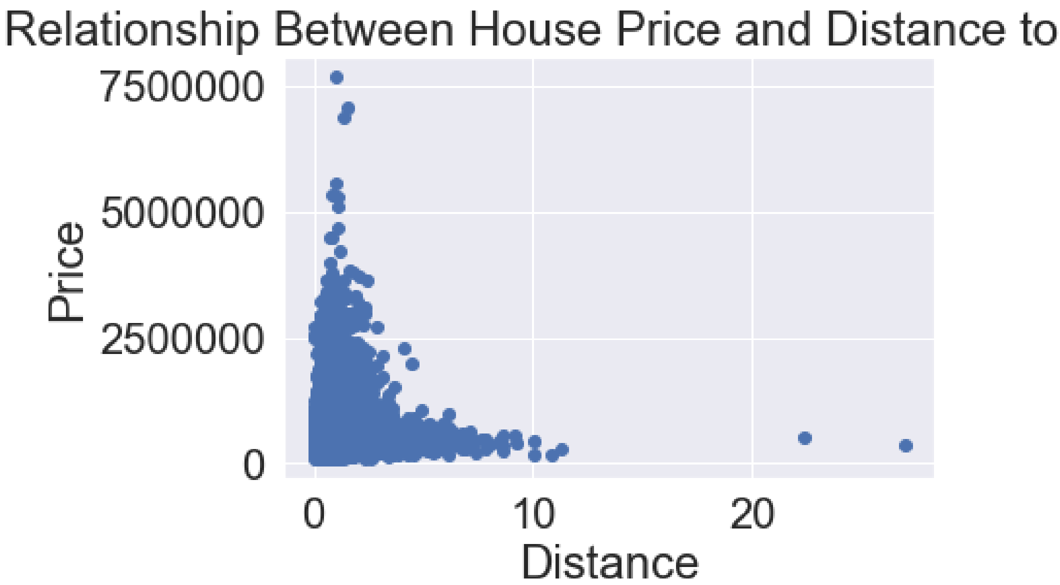
The closest distance to a school is 0.00 km (house located at the exact same latitude a as a school building). The farthest distance is 26.95 km.

In [257]:
```
plt.figure(figsize=(12,6))
sns.distplot(df['closest_distance_to_school'])
plt.title("Distribution of Distances to Schools", fontsize=20)
plt.xlabel('Distance to Closest School (km)');
print("Skewness:", df['closest_distance_to_school'].skew())
print("Kurtosis:", df['closest_distance_to_school'].kurt())
```

```
Skewness: 6.218078338828554
Kurtosis: 108.62323888858803
```



In [258]:
```python
plt.scatter(x=df['closest_distance_to_school'], y=df['price'])
plt.title('Relationship Between House Price and Distance to Schoo
plt.xlabel('Distance')
plt.ylabel('Price');
```



As expected, there seemed to be a negative correlation between distance to a school a
of a house. As the distance between a house and a school decreased, the house price

In [259]:
```python
ropping unnecessary columns
op = ['date','id','yr_built', 'yr_renovated', 'sqft_above', 'sqft
_cleaned = df.drop(columns = drop, axis=1)
```

In [260]:
```python
df_cleaned = df_cleaned.drop(columns = school_cols, axis=1)
```

In [261]:
```python
df_cleaned.head()
```

Out[261]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	con
0	221900.00	3	1.00	1180	5650	1.00	nan	0.00	
1	538000.00	3	2.25	2570	7242	2.00	0.00	0.00	
2	180000.00	2	1.00	770	10000	1.00	0.00	0.00	
3	604000.00	4	3.00	1960	5000	1.00	0.00	0.00	
4	510000.00	3	2.00	1680	8080	1.00	0.00	0.00	

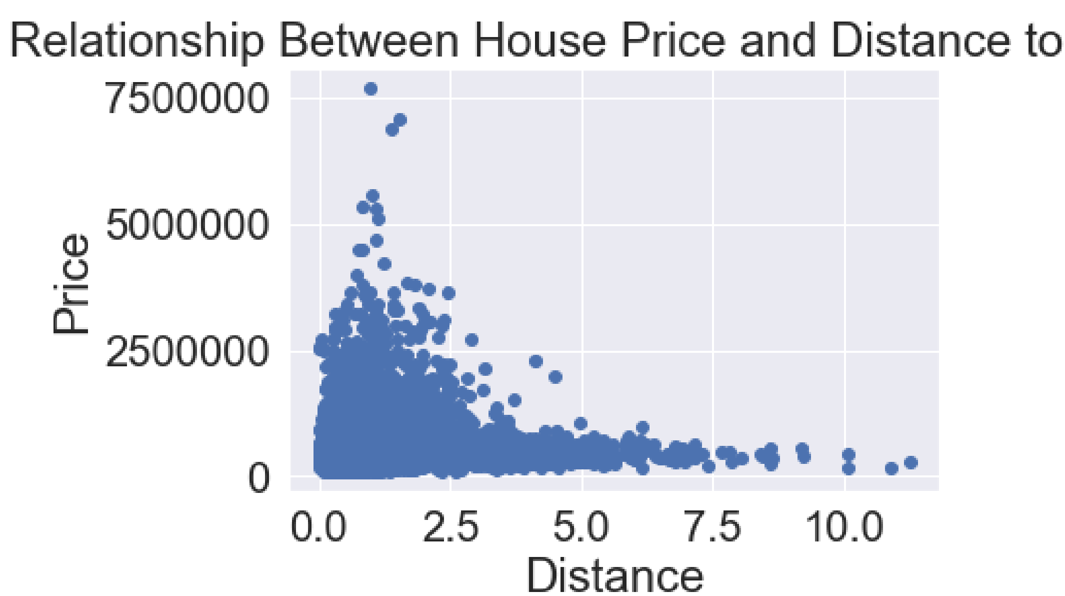
In [262]:
```python
df_cleaned.corr()
```

Out[262]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	w
price	1.00	0.31	0.53	0.70	0.09	0.26	
bedrooms	0.31	1.00	0.51	0.58	0.03	0.18	
bathrooms	0.53	0.51	1.00	0.76	0.09	0.50	
sqft_living	0.70	0.58	0.76	1.00	0.17	0.35	
sqft_lot	0.09	0.03	0.09	0.17	1.00	-0.00	
floors	0.26	0.18	0.50	0.35	-0.00	1.00	
waterfront	0.28	-0.00	0.07	0.11	0.02	0.02	
view	0.40	0.08	0.19	0.28	0.08	0.03	
condition	0.04	0.03	-0.13	-0.06	-0.01	-0.26	
grade	0.67	0.36	0.67	0.76	0.11	0.46	
zipcode	-0.05	-0.15	-0.20	-0.20	-0.13	-0.06	
lat	0.31	-0.01	0.02	0.05	-0.09	0.05	
long	0.02	0.13	0.22	0.24	0.23	0.13	
closest_distance_to_school	0.07	0.00	0.10	0.15	0.35	0.04	

In [263]:
```python
df_cleaned = df_cleaned.loc[df_cleaned.closest_distance_to_schoo
```

In [264]:
```python
plt.scatter(x=df_cleaned['closest_distance_to_school'], y=df_clea
plt.title('Relationship Between House Price and Distance to Schoo
plt.xlabel('Distance')
plt.ylabel('Price');
```



With outliers removed, we are able to more clearly visualize this relationship.

In [265]:
```
df_cleaned.corr()
```

Out[265]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	w
price	1.00	0.31	0.53	0.70	0.09	0.26	
bedrooms	0.31	1.00	0.51	0.58	0.03	0.18	
bathrooms	0.53	0.51	1.00	0.76	0.09	0.50	
sqft_living	0.70	0.58	0.76	1.00	0.17	0.35	
sqft_lot	0.09	0.03	0.09	0.17	1.00	-0.00	
floors	0.26	0.18	0.50	0.35	-0.00	1.00	
waterfront	0.28	-0.00	0.07	0.11	0.02	0.02	
view	0.40	0.08	0.19	0.28	0.08	0.03	
condition	0.04	0.03	-0.13	-0.06	-0.01	-0.26	
grade	0.67	0.36	0.67	0.76	0.11	0.46	
zipcode	-0.05	-0.15	-0.20	-0.20	-0.13	-0.06	

King County Top Schools

There was only a correlation of 0.07 between proximity to a school and house price. Sc narrowed this down to the top 8 school districts in King County, as per rankings on Nic see if there was a stronger correlation between house price and a highly ranked school

In [266]:
```
schools.lea_name.value_counts()
```

Out[266]:
```
Seattle Public Schools                                   107
Lake Washington School District                           53
Federal Way School District                               48
Kent School District                                      43
Highline School District                                  43
Bellevue School District                                  30
Renton School District                                    29
Issaquah School District                                  27
Northshore School District                                22
Auburn School District                                    22
Shoreline School District                                 19
Snoqualmie Valley School District                         12
Tahoma School District                                     9
Enumclaw School District                                   9
Riverview School District                                  9
Tukwila School District                                    7
Mercer Island School District                              5
Vashon Island School District                              5
Mary Walker School District                                4
Lake Washington Institute of Technology                    3
Skykomish School District                                  2
South Seattle Community College (CC Dist #6)               1
Seattle Central Community College                          1
Rainier Prep Charter School District                       1
First Place Scholars Charter School District               1
Green River Community College                              1
Excel Public Charter School LEA                            1
University of Washington (17904)                           1
Renton Technical College                                   1
Summit Public School: Sierra                               1
Monroe School District                                     1
Name: lea_name, dtype: int64
```

In [267]:
```
from bs4 import BeautifulSoup
# url for Niche.com King County school district ranking
url = f"https://www.niche.com/k12/search/best-school-districts/c/
response = requests.get(url)
# creating soup
soup = BeautifulSoup(response.text, 'lxml')
soup.findAll('section')
```

Out[267]:
```
[<section class="container"> <div class="customer-logo-wrapper">
ss="customer-logo"> <img alt="Logo" src="http://a.niche.com/wp-c
emes/niche-about/images/about-home/stacked-green.svg"/> </div> <
v class="page-title-wrapper"> <div class="page-title"> <h1>Pleas
you are a human</h1> </div> </div> <div class="content-wrapper">
ss="content"> <div id="px-captcha"> </div> <p> Access to this pa
en denied because we believe you are using automation tools to b
website.  </p> <p> This may happen as a result of the following:
> <li> Javascript is disabled or blocked by an extension (ad blo
example) </li> <li> Your browser does not support cookies </li>
Please make sure that Javascript and cookies are enabled on your
and that you are not blocking them from loading.  </p> <p> Refer
#5ff0f150-398d-11eb-9e5b-e9d0542fad5f </p> </div> </div> <div cl
-footer-wrapper"> <div class="page-footer"> <p> Powered by <a hr
s://www.perimeterx.com/whywasiblocked">PerimeterX</a> , Inc.  </
</div> </section>]
```

I attempted to web-scrape the data for the highest-ranked school districts in King Cou
Niche.com, but I was unable to do so due to being blocked by their server. So instead,
entered the eight school districts that were ranked in the A range (A+, A, A-) into a list.

In [268]:
```
top_schools = ['Mercer Island School District', 'Bellevue School
               'Lake Washington School District', 'Issaquah Schoo
               'Tahoma School District', 'Shoreline School Distri
               'Vashon Island School District', 'Snoqualmie Valle
               'Seattle Public Schools']
```

In [269]:
```
top_schools_df = schools.loc[schools['lea_name'].isin(top_school
top_schools_df.head()
```

Out[269]:

	year	ncessch	school_name	state_name	lea_name	zip_location	latitude	l
43	2015	530039000058	Ardmore Elementary School	Washington	Bellevue School District	98008	47.64	
44	2015	530039000060	Bellevue High School	Washington	Bellevue School District	98004	47.60	
45	2015	530039000062	Bennett Elementary School	Washington	Bellevue School District	98008	47.62	
46	2015	530039000063	Cherry Crest Elementary School	Washington	Bellevue School District	98005	47.64	
47	2015	530039000064	Chinook Middle School	Washington	Bellevue School District	98004	47.63	

In [270]:
```
# saving copy of DataFrame as csv file
#top_schools_df.to_csv('./data/top_schools.csv')
```

In [271]:
```
top_school_coordinates = []
x = round(top_schools_df.latitude, 2)
y = round(top_schools_df.longitude, 2)
top_school_coordinates = list(zip(x,y))
```

In [272]:
```
for i in range(len(top_school_coordinates)):
    df[f'top_school_{i}'] = distance_to(top_school_coordinates[i

top_school_cols = []
for i in range(len(top_school_coordinates)):
    top_school_cols.append(f'top_school_{i}')
    df['closest_distance_to_top_school'] = df[top_school_cols].m
```

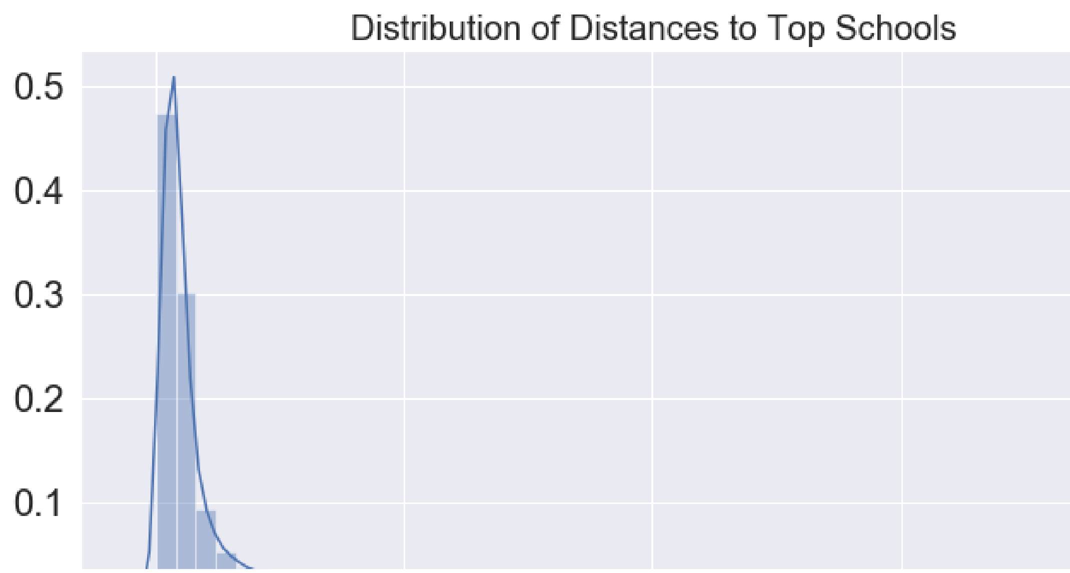
In [273]:
```
df.closest_distance_to_top_school.describe()
```

Out[273]:
```
count    21597.00
mean         3.09
std          4.41
min          0.00
25%          0.60
50%          1.05
75%          3.43
max         40.09
Name: closest_distance_to_top_school, dtype: float64
```

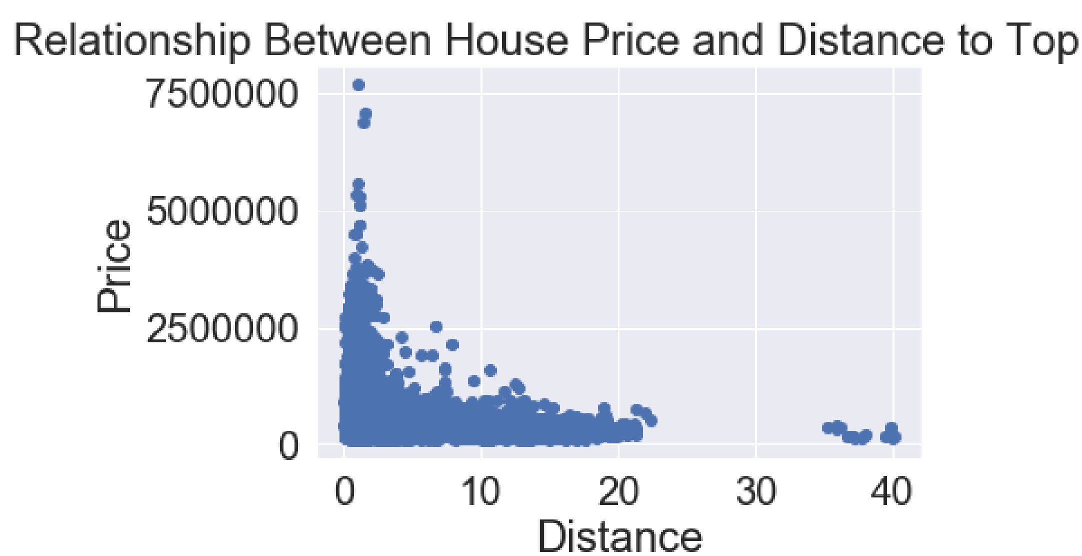
The closest distance from a house to a top school is 0.00 km (house located at the exa
latitude and longitude as a school building). The farthest distance is 40.09 km

In [274]:
```
plt.figure(figsize=(12,6))
sns.distplot(df['closest_distance_to_top_school'])
plt.title("Distribution of Distances to Top Schools", fontsize=2(
plt.xlabel('Distance to Closest Top School (km)');
print("Skewness:", df['closest_distance_to_top_school'].skew())
print("Kurtosis:", df['closest_distance_to_top_school'].kurt())
```

```
Skewness: 2.2762581074960346
Kurtosis: 5.809128777092479
```



In [275]:
```
plt.scatter(x=df['closest_distance_to_top_school'], y=df['price'
plt.title('Relationship Between House Price and Distance to Top S
plt.xlabel('Distance')
plt.ylabel('Price');
```



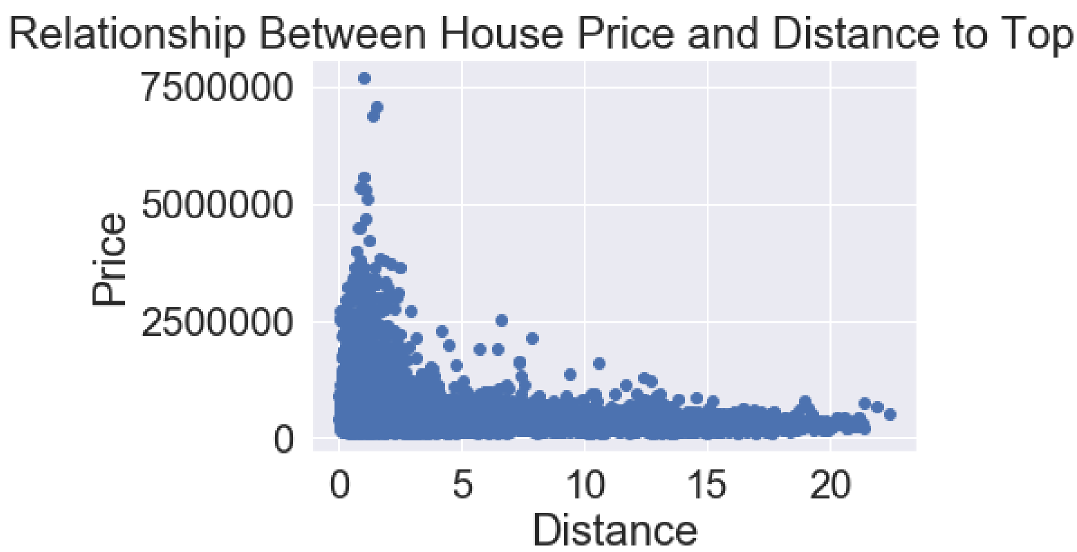
In [276]:
```
#dropping unnecessary columns
drop = ['date','id','yr_built', 'yr_renovated', 'sqft_above', 'so
df_cleaned = df.drop(columns = drop, axis=1)
```

In [277]:
```python
df_cleaned = df_cleaned.drop(columns = school_cols, axis=1)
```

In [278]:
```python
df_cleaned = df_cleaned.drop(columns = top_school_cols, axis=1)
```

In [279]:
```python
df_cleaned = df_cleaned.loc[df_cleaned.closest_distance_to_top_s
```

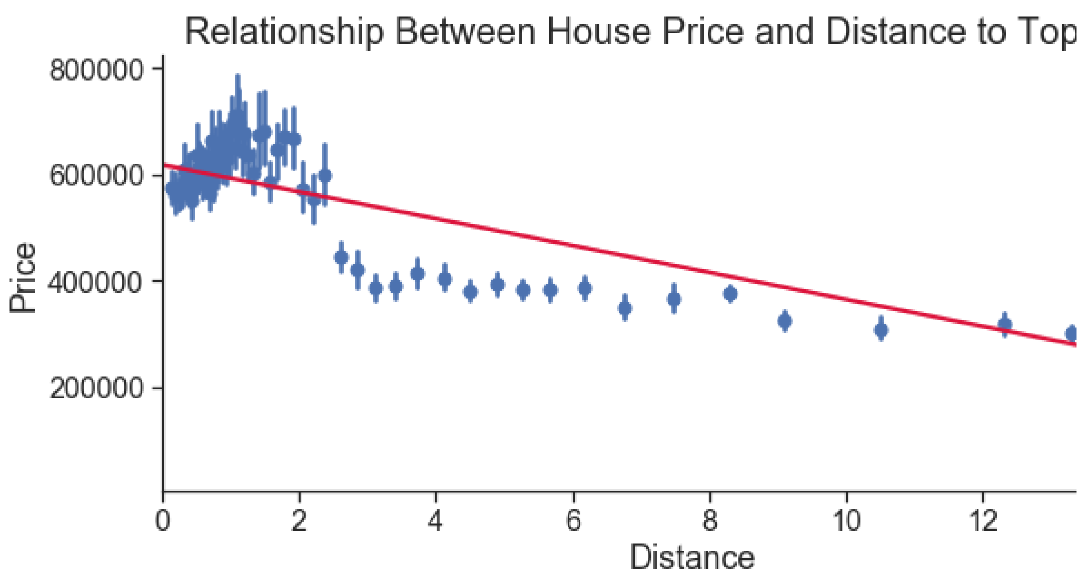
In [280]:
```python
plt.scatter(x=df_cleaned['closest_distance_to_top_school'], y=df_
plt.title('Relationship Between House Price and Distance to Top S
plt.xlabel('Distance')
plt.ylabel('Price')
plt.savefig('./visualizations/school_price.png');
```



In [281]:
```python
df_cleaned.corr()
```

sqft_living	0.70	0.58	0.76	1.00	0.17	0.3
sqft_lot	0.09	0.03	0.09	0.17	1.00	-0.0
floors	0.26	0.18	0.50	0.35	-0.00	1.0
waterfront	0.28	-0.00	0.07	0.11	0.02	0.0
view	0.40	0.08	0.19	0.28	0.08	0.0
condition	0.04	0.03	-0.13	-0.06	-0.01	-0.2
grade	0.67	0.36	0.67	0.76	0.11	0.4
zipcode	-0.05	-0.16	-0.21	-0.20	-0.13	-0.0
lat	0.31	-0.01	0.03	0.05	-0.09	0.0
long	0.03	0.14	0.23	0.25	0.23	0.1
closest_distance_to_school	0.07	0.01	0.11	0.16	0.36	0.0

In [333]:
```
sns.set_style('ticks')
sns.lmplot(x='closest_distance_to_top_school', y='price', data=d
plt.title('Relationship Between House Price and Distance to Top S
plt.xlabel('Distance', fontsize=18)
plt.ylabel('Price', fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16);
#plt.ylim(100000,750000)
plt.xlim(0, 15.5);
plt.tight_layout()
plt.savefig('./visualizations/price_school_2.png')
```



When we look at the distance to a school and price, there is not much of a correlation
However, once we narrow it down to the top schools, we start to see a stronger negati
correlation. So as the distance to a top school decreases, the house price increases.

Proximity to Coffee Shops via Yelp API

We speculated that there may be a relationship between good coffee shops and highe
prices. We used the Yelp API to obtain the data for the top 50 highest-rated coffee sho
the provided latitudes and longitudes to calculate their distances from each home.

In [283]:
```
import requests
import json
```

In [284]:
```
def get_keys(path):
    """Retrieves API key from files as api_key."""
    with open(path) as f:
        return json.load(f)
```

In [285]:
```
keys = get_keys("/Users/dtunnicliffe/.secret/yelp_api.json")
api_key = keys['api_key']
```

In [286]:
```
term = 'coffee'
location = 'King County, WA'
SEARCH_LIMIT = 50
mochas = pd.DataFrame([])
def yelp(term, location, SEARCH_LIMIT):
    """
    Creates a new dataframe of information retrieved from yelp AI

    Searches businesses and returns top results based on criteri
    Makes API call as if searching on yelp.
    Returns relevant information for businesses such as name, loc

    Parameters:
    term (str): user input term to search for.
    location (str): user input city, state, or zip code to searcl
    SEARCH_LIMIT (int): user input number of results to return.

    Returns:
    New dataframe populated with requested information.

    """
    global mochas
    url = 'https://api.yelp.com/v3/businesses/search'
    headers = {
    'Authorization': f'Bearer {api_key}',
    }
    url_params = {
    'term': term.replace(' ', '+'),
    'location': location.replace(' ', '+'),
    'limit': SEARCH_LIMIT,
    'sort_by': 'rating'
    }
    response = requests.get(url, headers=headers, params=url_para
    df_temp = pd.DataFrame.from_dict(response.json()['businesses
    mochas = mochas.append(df_temp)
    return mochas
```

In [287]:
```
mochas = yelp(term, location, SEARCH_LIMIT)
```

In [288]:
```
mochas.shape
```

Out[288]:
```
(50, 16)
```

In [289]:
```
mochas.head()
```

1	PJakGoM3gkStlwG5AvPadw	mighty-mugs-coffee-kent	Mighty Mugs Coffee	https media1.fl.yelpcdn.com/bphoto/xKB
2	S6CXIQ5KrMpTPZf1eNMa2w	five-stones-coffee-company-redmond	Five Stones Coffee Company	https media3.fl.yelpcdn.com/bphoto/Omz
3	0ms-mWSw4ywRDM4Yn11r7g	lamppost-coffee-roasters-bonney-lake	Lamppost Coffee Roasters	https media2.fl.yelpcdn.com/bphoto/d4p
4	rl43r90cPQJ6qCo-eEsXpA	burien-press-burien	Burien Press	https media1.fl.yelpcdn.com/bphoto/m-

In [290]:
```
coffee_coordinates = []
x = [round(coordinate['latitude'], 2) for coordinate in mochas['c
y = [round(coordinate['longitude'], 2) for coordinate in mochas[
coffee_coordinates = list(zip(x,y))
```

In [291]:
```
in range(len(coffee_coordinates)):
_cleaned[f'coffee_{i}'] = distance_to(coffee_coordinates[i])

_cols = []
in range(len(coffee_coordinates)):
ffee_cols.append(f'coffee_{i}')
_cleaned['closest_distance_to_good_coffee'] = df_cleaned[coffee_c
```

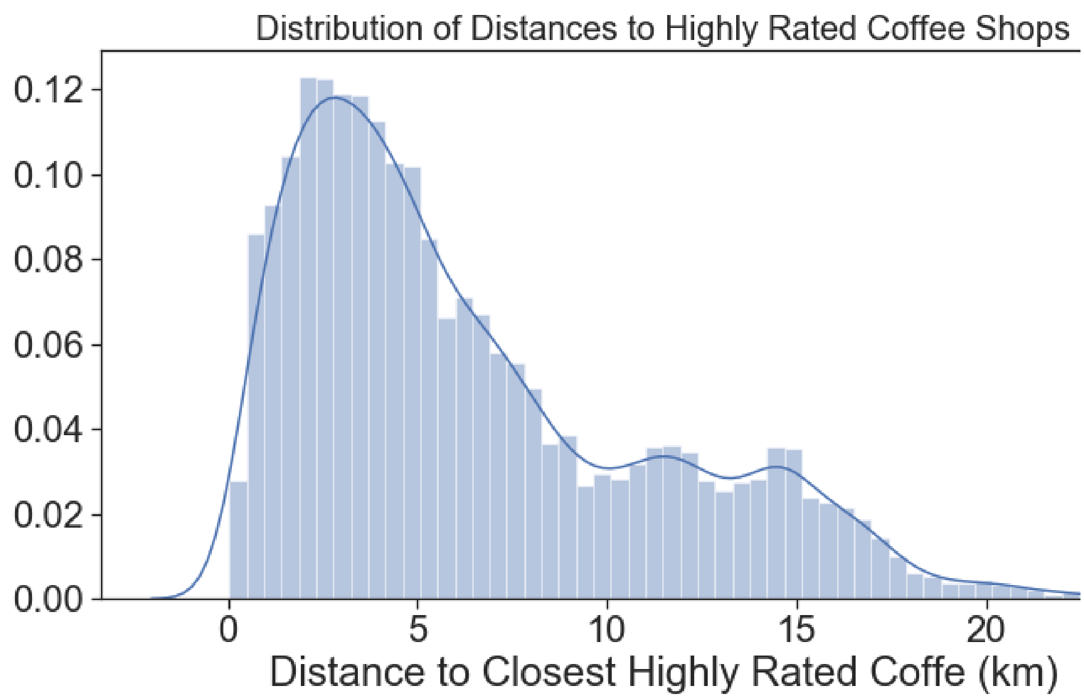
In [292]:
```
df_cleaned.closest_distance_to_good_coffee.describe()
```

Out[292]:
```
count    21580.00
mean         6.45
std          4.75
min          0.03
25%          2.74
50%          5.01
75%          9.22
max         22.89
Name: closest_distance_to_good_coffee, dtype: float64
```

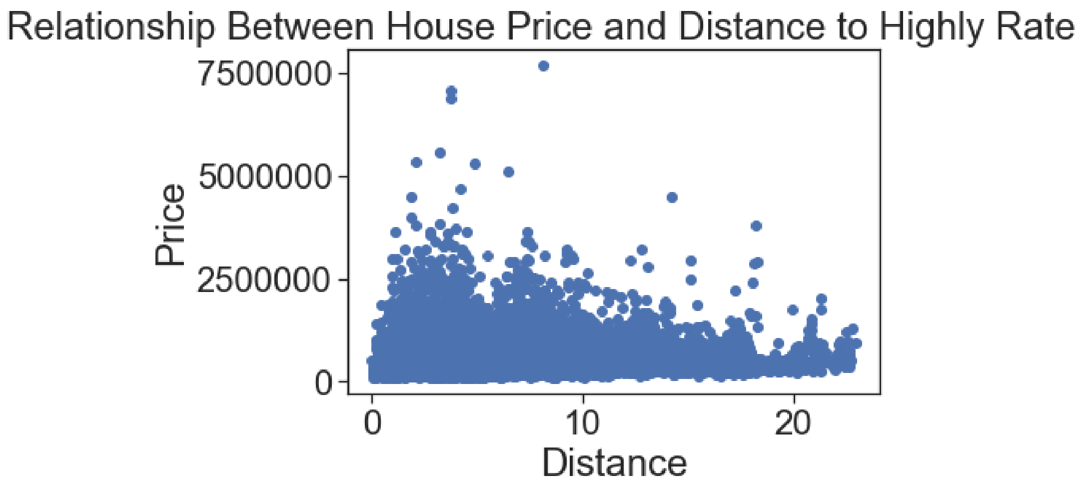
The closest distance to a highly rated coffee shop is 0.03 km. The farthest distance is 2

In [293]:
```
igure(figsize=(12,6))
istplot(df_cleaned['closest_distance_to_good_coffee'])
itle("Distribution of Distances to Highly Rated Coffee Shops", fc
label('Distance to Closest Highly Rated Coffe (km)');
("Skewness:", df_cleaned['closest_distance_to_good_coffee'].skew(
("Kurtosis:", df_cleaned['closest_distance_to_good_coffee'].kurt(
```

```
Skewness: 0.9194557509542861
Kurtosis: -0.03028616916931748
```



In [294]:
```
plt.scatter(x=df_cleaned['closest_distance_to_good_coffee'], y=df
plt.title('Relationship Between House Price and Distance to Highl
plt.xlabel('Distance')
plt.ylabel('Price');
```



In [295]:
```
#dropping unnecessary columns
df_cleaned = df_cleaned.drop(columns = coffee_cols, axis=1)
df_cleaned.head()
```

Out[295]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	con
0	221900.00	3	1.00	1180	5650	1.00	nan	0.00	
1	538000.00	3	2.25	2570	7242	2.00	0.00	0.00	
2	180000.00	2	1.00	770	10000	1.00	0.00	0.00	
3	604000.00	4	3.00	1960	5000	1.00	0.00	0.00	
4	510000.00	3	2.00	1680	8080	1.00	0.00	0.00	

In [296]:
```
optimal = df_cleaned.loc[(df_cleaned['price']>180000) & (df_clea
optimal.corr()
```

Out[296]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floo
price	1.00	0.19	0.32	0.44	0.07	0.
bedrooms	0.19	1.00	0.46	0.59	0.02	0.
bathrooms	0.32	0.46	1.00	0.67	0.03	0.
sqft_living	0.44	0.59	0.67	1.00	0.14	0.
sqft_lot	0.07	0.02	0.03	0.14	1.00	-0.
floors	0.21	0.11	0.49	0.28	-0.05	1.
waterfront	0.03	-0.04	-0.04	-0.02	0.02	-0.
view	0.14	0.01	0.04	0.10	0.10	-0.
condition	0.01	0.02	-0.16	-0.08	0.01	-0.
grade	0.47	0.26	0.56	0.60	0.04	0.
zipcode	0.03	-0.16	-0.23	-0.23	-0.14	-0.
lat	0.47	-0.10	-0.10	-0.13	-0.11	-0.
long	0.07	0.14	0.24	0.28	0.22	0.
closest_distance_to_school	0.06	-0.00	0.08	0.14	0.39	0.
closest_distance_to_top_school	-0.42	0.09	0.07	0.12	0.13	-0.
closest_distance_to_good_coffee	0.17	-0.12	-0.14	-0.16	-0.06	-0.

Unfortunately, there was no observable relationship between house price and distance
rated coffee shop.

Top 10 Highest-Rated Coffee Shops from Yelp API

We then gathered data for the top 10 highest-rated coffee shops in King County, as pe
API, and tried to find a connection between house price and distance from a very highl
coffee shop.

In [297]:
```
term = 'coffee'
location = 'King County, WA'
SEARCH_LIMIT = 10
espresso = pd.DataFrame([])
def yelp(term, location, SEARCH_LIMIT):
    """
    Creates a new dataframe of information retrieved from yelp AI

    Searches businesses and returns top results based on criteri
    Makes API call as if searching on yelp.
    Returns relevant information for businesses such as name, loc

    Parameters:
    term (str): user input term to search for.
    location (str): user input city, state, or zip code to searcl
    SEARCH_LIMIT (int): user input number of results to return.

    Returns:
    New dataframe populated with requested information.

    """
    global espresso
    url = 'https://api.yelp.com/v3/businesses/search'
    headers = {
    'Authorization': f'Bearer {api_key}',
    }
    url_params = {
    'term': term.replace(' ', '+'),
    'location': location.replace(' ', '+'),
    'limit': SEARCH_LIMIT,
    'sort_by': 'rating'
    }
    response = requests.get(url, headers=headers, params=url_para
    df_temp = pd.DataFrame.from_dict(response.json()['businesses
    espresso = espresso.append(df_temp)
    return espresso
```

In [298]:
```
espresso = yelp(term, location, SEARCH_LIMIT)
```

In [299]:
```
espresso.shape
```

Out[299]:
```
(10, 16)
```

In [300]:
```
espresso.head(10)
```

Out[300]:

	id	alias	name	
0	S6CXIQ5KrMpTPZf1eNMa2w	five-stones-coffee-company-redmond	Five Stones Coffee Company	media3.fl.yelpcdn.com/bphoto/
1	EWqgeiGor-aVJlMLc8iSKw	boon-boona-coffee-renton	Boon Boona Coffee	media3.fl.yelpcdn.com/bphotc
2	v7xfqk9f7N8A98AQ2kddWg	anchorhead-coffee-bellevue-3	Anchorhead Coffee	media3.fl.yelpcdn.com/bphotc
3	t2DOOFh-oJLddtpxbVlDrQ	huxdotter-coffee-north-bend	Huxdotter Coffee	media3.fl.yelpcdn.com/bphoto
4	-MzbuOLr2kAoqlQY8w7ECA	pioneer-coffee-north-bend-north-bend	Pioneer Coffee - North Bend	media3.fl.yelpcdn.com/bphot
5	kybVpzGFcYov1d0X00vDjQ	candor-coffee-renton	Candor Coffee	media4.fl.yelpcdn.com/bphotc
6	oUk6IZAFQ37R5OK0etWocg	the-north-bend-bakery-north-bend	The North Bend Bakery	media1.fl.yelpcdn.com/bphoto/
7	9DJY3ndAM0E6T7qGtrq0kg	issaquah-coffee-company-issaquah	Issaquah Coffee Company	media4.fl.yelpcdn.com/bphoto/
8	9yDshpKSd3mjYs2JUY5JbQ	espresso-chalet-index	Espresso Chalet	media1.fl.yelpcdn.com/bphotc
9	RNPQ65ZXmRdtH7dDGOLYMQ	bobs-espresso-snoqualmie-pass-3	Bobs Espresso	media3.fl.yelpcdn.com/bphot

In [301]:
```
great_coffee_coordinates = []
x = [round(coordinate['latitude'], 2) for coordinate in espresso
y = [round(coordinate['longitude'], 2) for coordinate in espress
great_coffee_coordinates = list(zip(x,y))
```

In [302]:
```
 i in range(len(great_coffee_coordinates)):
 df_cleaned[f'great_coffee_{i}'] = distance_to(great_coffee_coord

at_coffee_cols = []
 i in range(len(great_coffee_coordinates)):
 great_coffee_cols.append(f'great_coffee_{i}')
 df_cleaned['closest_distance_to_great_coffee'] = df_cleaned[grea
```

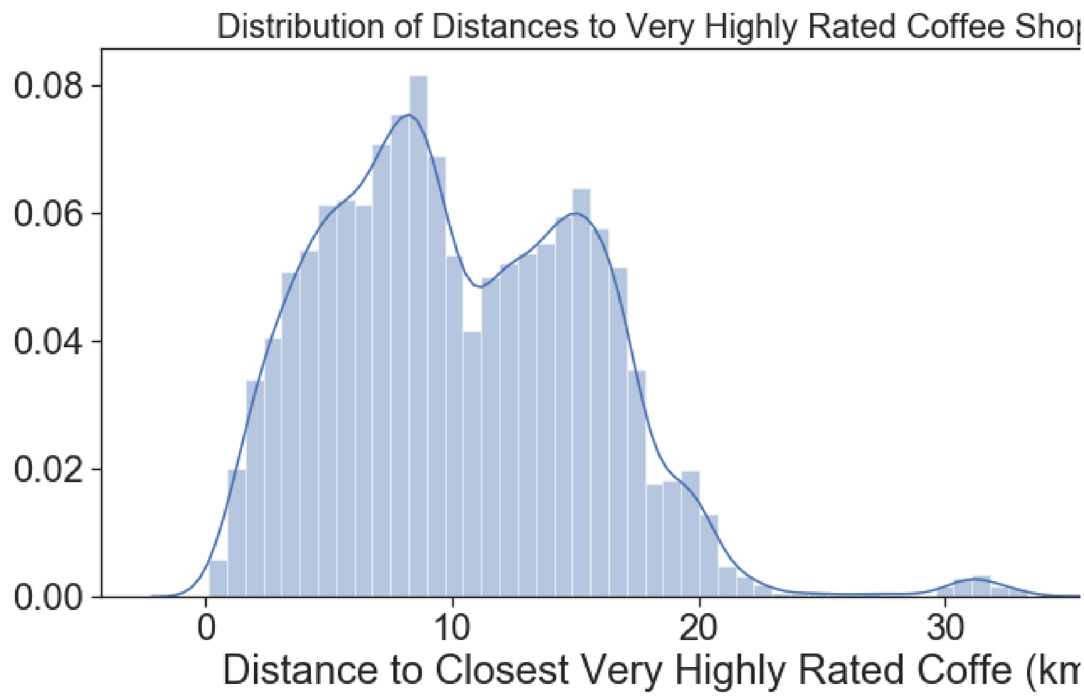
In [303]:
```
df_cleaned.closest_distance_to_great_coffee.describe()
```

Out[303]:
```
count   21580.00
mean       10.33
std         5.39
min         0.12
25%         6.16
50%         9.60
75%        14.39
max        36.98
Name: closest_distance_to_great_coffee, dtype: float64
```

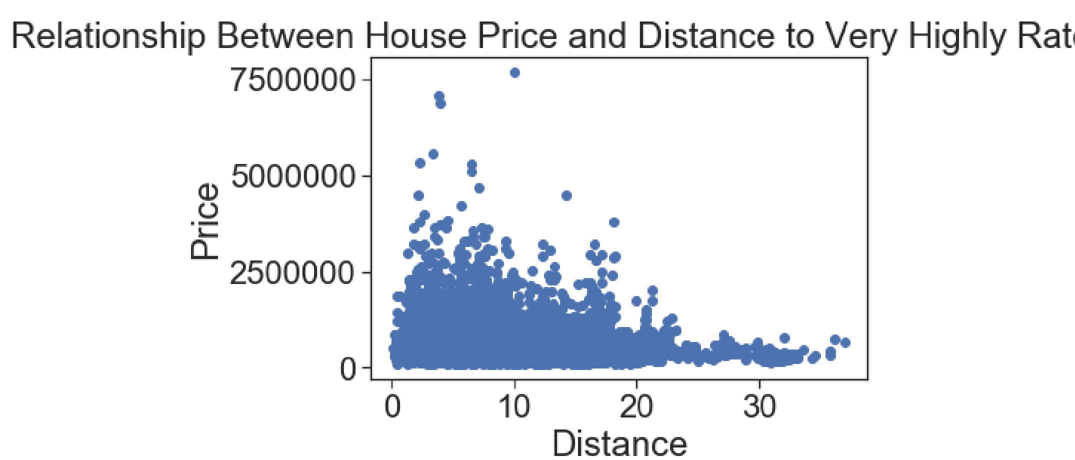
The closest distance to a very highly rated coffee shop is 0.09 km. The farthest distanc
km.

In [304]:
```
plt.figure(figsize=(12,6))
sns.distplot(df_cleaned['closest_distance_to_great_coffee'])
plt.title("Distribution of Distances to Very Highly Rated Coffee
plt.xlabel('Distance to Closest Very Highly Rated Coffe (km)');
print("Skewness:", df_cleaned['closest_distance_to_great_coffee'
print("Kurtosis:", df_cleaned['closest_distance_to_great_coffee'
```

```
Skewness: 0.6130716695116233
Kurtosis: 0.7558328850401486
```



In [305]:
```
plt.scatter(x=df_cleaned['closest_distance_to_great_coffee'], y=
plt.title('Relationship Between House Price and Distance to Very
plt.xlabel('Distance')
plt.ylabel('Price');
```



In [306]:
```
ing house price by distance to highly rated coffee
_style('darkgrid')
.ot(x='closest_distance_to_great_coffee', y='price', data=df_clear
.e('Relationship Between House Price and Distance to Very Highly F
oel('Distance', fontsize=15)
oel('Price', fontsize=15)
n(0, 25)
.ine(y=600000, ls='--', c='green');
nt_layout()
efig('./visualizations/price_coffee_2.png')
```

In [307]:
```
#dropping unnecessary columns
df_cleaned = df_cleaned.drop(columns = great_coffee_cols, axis=1
df_cleaned.head()
```

Out[307]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	con
0	221900.00	3	1.00	1180	5650	1.00	nan	0.00	
1	538000.00	3	2.25	2570	7242	2.00	0.00	0.00	
2	180000.00	2	1.00	770	10000	1.00	0.00	0.00	
3	604000.00	4	3.00	1960	5000	1.00	0.00	0.00	
4	510000.00	3	2.00	1680	8080	1.00	0.00	0.00	

In [308]:
```
df_cleaned.corr()
```

floors	0.26	0.18	0.50	0.35	-0.00	1.
waterfront	0.28	-0.00	0.07	0.11	0.02	0.
view	0.40	0.08	0.19	0.28	0.08	0.
condition	0.04	0.03	-0.13	-0.06	-0.01	-0.
grade	0.67	0.36	0.67	0.76	0.11	0.
zipcode	-0.05	-0.16	-0.21	-0.20	-0.13	-0.
lat	0.31	-0.01	0.03	0.05	-0.09	0.
long	0.03	0.14	0.23	0.25	0.23	0.
closest_distance_to_school	0.07	0.01	0.11	0.16	0.36	0.
closest_distance_to_top_school	-0.30	-0.00	-0.05	-0.06	0.11	-0.
closest_distance_to_good_coffee	0.03	-0.10	-0.12	-0.12	-0.06	-0.
closest_distance_to_great_coffee	-0.20	-0.14	-0.15	-0.18	0.07	-0.

We found that, similar to good schools, there was a negative correlation between hous
proximity to a very highly-rated coffee shop. As distance to a great coffee shop decrea
price increases.

Proximity to Scientology Churches

We had heard a theory that homes located near scientology churches tend to be highe
due to the fact that scientologists are known for investing funds in their surrounding co
While certainly unique, we wanted to explore this feature and see if there was any conr
between house price and proximity to a church of scientology.

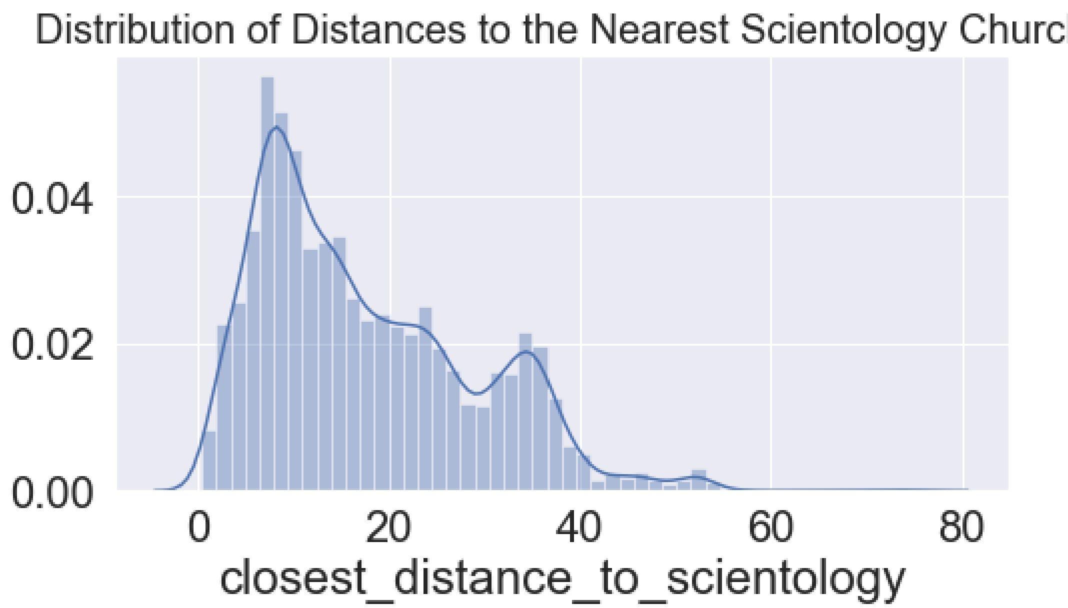
In [309]:
```
#locations pulled from scientology-seattle.org
church_of_scientology_mission = (47.818100, -122.315430)
church_of_scientology_washington = (47.622380, -122.361020)
church_of_scientology_life_improvement_center = (47.615060, -122
```

In [310]:
```
reating a dataframe to investigate scientology proximity
entology = pd.read_csv('./data/kc_house_data.csv')

reating new columns of distances from houses to scientology churc
unning our haversine calculator function on these points
entology['distance_to_scientology_m'] = distance_to(church_of_sci
entology['distance_to_scientology_w'] = distance_to(church_of_sci
entology['distance_to_scientology_l'] = distance_to(church_of_sci
entology['closest_distance_to_scientology'] = scientology[['dista
                                                           'distance_tc
                                                           'distance_tc
```

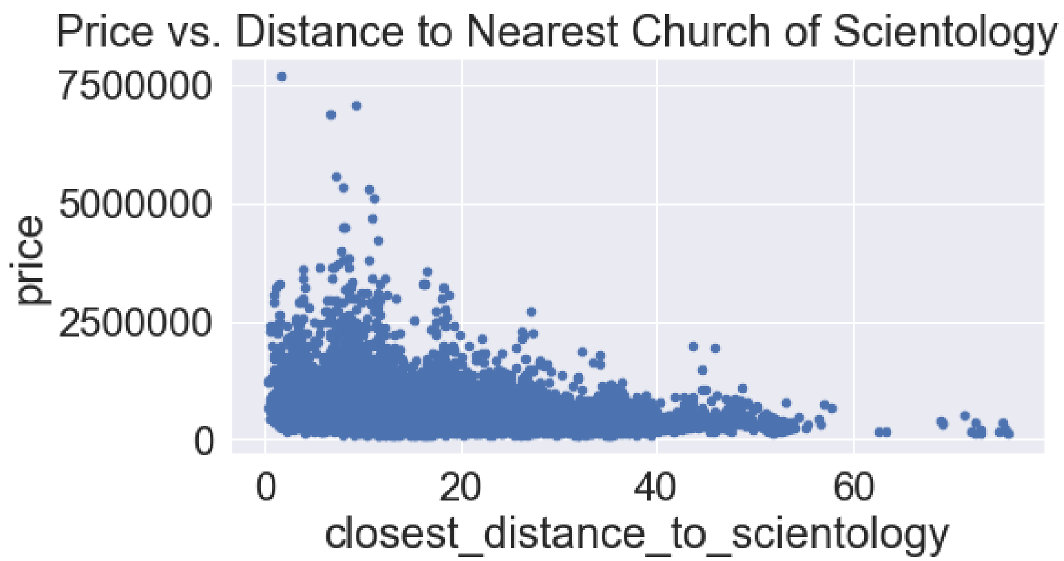
In [311]:
```
plt.figure(figsize=(8,4))
sns.distplot(scientology['closest_distance_to_scientology'])
plt.title("Distribution of Distances to the Nearest Scientology C
print("Distribution appears to deviate slightly from a normal dis
print("Displays a positive skewness.")
print("Skewness:", scientology['closest_distance_to_scientology'
print("Kurtosis:", scientology['closest_distance_to_scientology'
```

```
Distribution appears to deviate slightly from a normal distribut
Displays a positive skewness.
Skewness: 0.8119816020278896
Kurtosis: 0.1550669496730026
```



In [312]:
```
#church of scientology vs price plot
plot1 = pd.concat([scientology['price'], scientology['closest_di
plot1.plot.scatter(x='closest_distance_to_scientology', y='price
plt.title("Price vs. Distance to Nearest Church of Scientology M
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, wh
ld be avoided as value-mapping will have precedence in case its 
tches with 'x' & 'y'.  Please use a 2-D array with a single row 
ally want to specify the same RGB or RGBA value for all points.
```



In [313]:
```
scientology.corr()
```

yr_built	0.02	0.05	0.16	0.51	0.32	0.0
yr_renovated	-0.01	0.13	0.02	0.05	0.06	0.0
zipcode	-0.01	-0.05	-0.15	-0.20	-0.20	-0.1
lat	-0.00	0.31	-0.01	0.02	0.05	-0.0
long	0.02	0.02	0.13	0.22	0.24	0.2
sqft_living15	-0.00	0.59	0.39	0.57	0.76	0.1
sqft_lot15	-0.14	0.08	0.03	0.09	0.18	0.7
distance_to_scientology_m	0.01	-0.29	0.02	0.03	0.00	0.1
distance_to_scientology_w	0.01	-0.28	0.07	0.09	0.09	0.2
distance_to_scientology_l	0.00	-0.30	0.05	0.07	0.07	0.2
closest_distance_to_scientology	0.01	-0.28	0.05	0.08	0.07	0.2

Like schools and coffee shops, there appears to be a negative correlation between pro
scientology church and the price of a house. As distance from a home to a scientology
decreases, house price tends to increase.

Web-scraped Data for Proximity to Parks

We hypthesized that being close to a park may have a correlation with house price as \
web-scraped data to investigate this possibility.

In [314]:
```
# web-scraping park data from kingcounty.gov
url_parks = 'https://www.kingcounty.gov/services/parks-recreatio
html_parks = requests.get(url_parks)
soup_parks = BeautifulSoup(html_parks.content, 'html.parser')
addresses = soup_parks.findAll('strong')
```

In [315]:
```
ddresses = []
em in addresses:
rk_addresses.append(item.text.strip())

ed = ['Access','Use','Useful Links','Acreage:','Usage:','','Acces
      'Length:','Use:','Access:','Useful links','.','Trail length
ddresses = [x for x in park_addresses if x not in unwanted]
```

In [316]:
```
names = soup_parks.findAll('a', class_ = 'collapsed')
```

In [317]:
```
park_names = []
for item in names:
    park_names.append(item.text.strip())
```

In [318]:
```
# removing inconsistent data
# no addresses listed for these particular parks
park_names.pop(0)
park_names.pop(27)
park_names.pop(7)
park_names.pop(41)
park_names.pop(62)
park_names.pop(-39)
```

Out[318]:
```
'Rattlesnake Mountain Scenic Area'
```

In [319]:
```
print(len(park_names))
print(len(park_addresses))
```

```
158
158
```

In [320]:
```
parks = dict(zip(park_names, park_addresses))
```

In [321]:
```
parks_df = pd.DataFrame.from_dict(parks, orient = 'index')
# saving to csv file
# parks_df.to_csv('./data/ParkAddresses_wLatLong.csv')
```

In [322]:

```
# importing park data
# reading the csv file
king_parks = pd.read_csv('data/ParkAddresses_wLatLong.csv', inde:
# previewing the DataFrame
king_parks.head()
```

Out[322]:

	Address	Combined	
ID			
0.00	Auburn Black Diamond Rd and SE Green Valley Rd...	47.301182311345315, -122.17491469179195	47
1.00	NE 165th St and 179th Pl NE Redmond WA 98072	47.74702351303733, -122.09810603412113	47
2.00	NaN	NaN	ı
3.00	NE 138th and Juanita Drive NE Kirkland WA 98028	47.72417796430824, -122.2384511052857	47
4.00	S 284th Pl and 37th Ave S Federal Way WA 98003	47.34814028865613, -122.2811067550002	47

In [323]:

```
king_parks.dropna(inplace=True)
```

In [324]:
```
#create function to find distances between all points in DF and :
def find_distance(dataframe):
    """
    Calculates distance between points of interest and houses.

    Generates a distance matrix for distances between houses and
    Calculates distance from each point in dataframe (df) to poi
    Converts latitude and longitude to radians in order to calcu
    Returns values as kilometers.

    Parameters:
    dataframe (Pandas DataFrame object): user input name of Panda

    Returns:
    Matrix of distances.

    """
    dist = sklearn.neighbors.DistanceMetric.get_metric('haversine

    #convert lat and long to radians
    dataframe[['lat_radians','long_radians']] = (np.radians(data

    #create list matrix (results in miles)
    dist_matrix = (dist.pairwise
    (df[['lat_radians_A','long_radians_A']],
     dataframe[['lat_radians','long_radians']])*3959)

    #return a matrix DataFrame
    return pd.DataFrame(dist_matrix)
```

In [325]:
```
#convert lat and long to radians in housing data
df[['lat_radians_A','long_radians_A']] = (np.radians(df.loc[:,['
```

In [326]:
```
park_matrix = find_distance(king_parks)
```

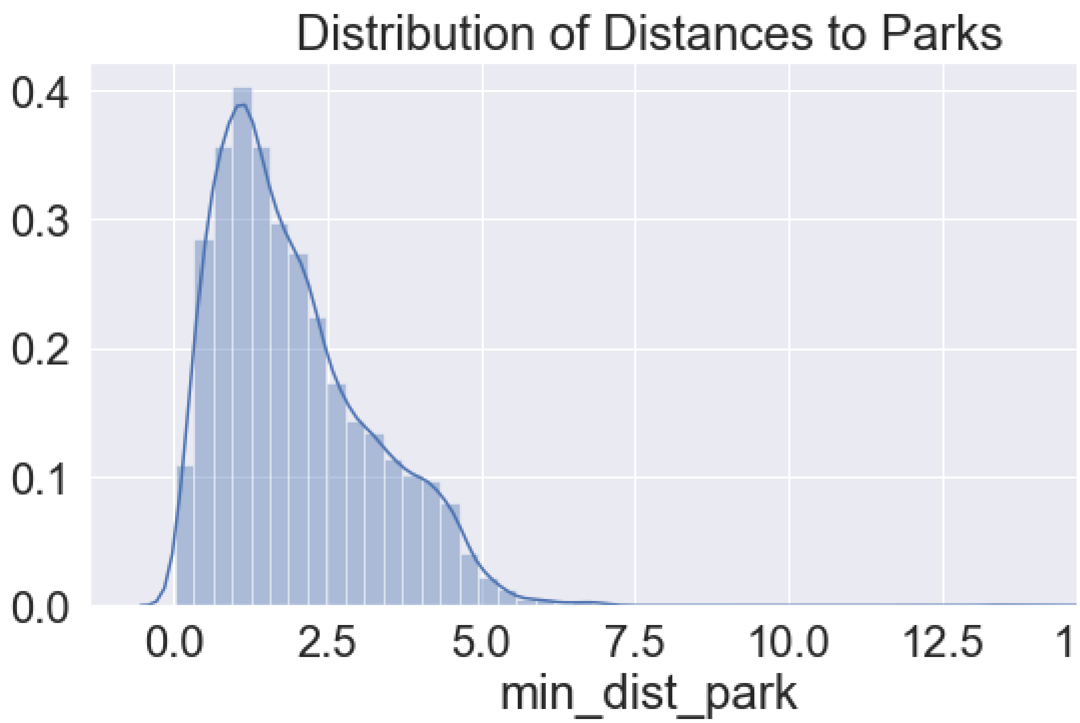
In [327]:
```
#find min distance in each row
park_min_matrix = park_matrix.where(park_matrix.values == park_ma
    axis=1)[:,None]).drop_duplicates()
```

In [328]:
```
#create a new column with only min distance and remove the rest
park_min_matrix['min_dist_park'] = park_min_matrix[park_min_matr
    lambda x: ','.join(x.dropna().astype(str)),
    axis=1)
nearest_park = park_min_matrix['min_dist_park']
```

In [329]:
```
data2 = df.join(nearest_park)
data2['min_dist_park']= data2['min_dist_park'].astype('float64')
```

In [330]:
```
# data2[['min_dist_park']].to_csv('data/park_distance.csv')
```

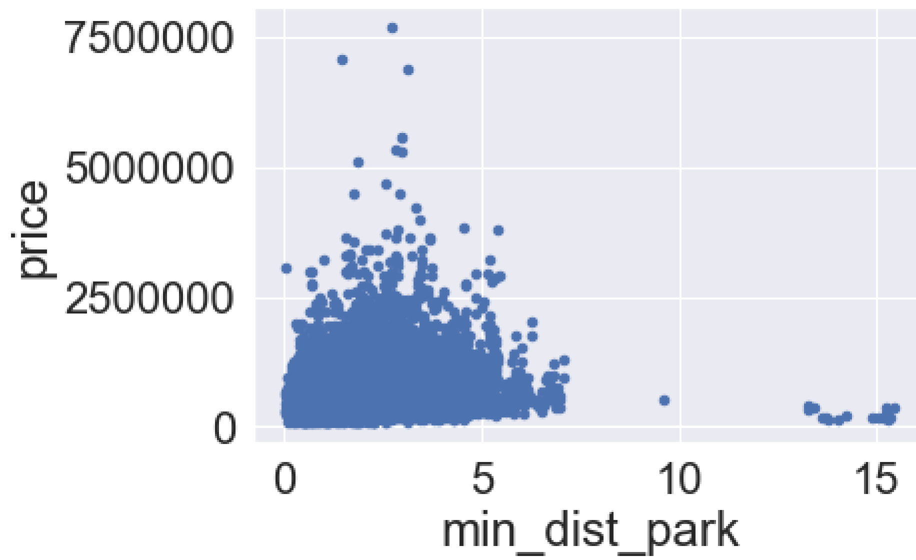
In [331]:
```
plt.figure(figsize=(10,5))
sns.distplot(data2['min_dist_park'])
plt.title('Distribution of Distances to Parks');
```



In [332]:
```
plot.scatter(x='min_dist_park', y='price');
'corr. price and parks: ' + str(data2['price'].corr(data2['min_di
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, wh
ld be avoided as value-mapping will have precedence in case its l
tches with 'x' & 'y'.  Please use a 2-D array with a single row i
ally want to specify the same RGB or RGBA value for all points.
```

```
corr. price and parks: 0.1640772485169492
```



It was not yet clear whether there was a relationship between proximity to a park and tl
home. As we continued our exploration, removed outliers, narrowed down our data, ar
our park list to eliminate forests and trail heads, we began to see more of a connection

Please see our next notebook, 'data_preparation', for the cleaning, compiling, and tran
of our data.

In [ ]: