015-assignment

February 9, 2022

Assignment: Housing in Brazil

[13]:
```
import wqet_grader

wqet_grader.init("Project 1 Assessment")
```

```
<IPython.core.display.HTML object>
```

In this assignment, you'll work with a dataset of homes for sale in Brazil. Your goal is to determine if there are regional differences in the real estate market. Also, you will look at southern Brazil to see if there is a relationship between home size and price, similar to what you saw with housing in some states in Mexico.

Before you start: Import the libraries you'll use in this notebook: Matplotlib, pandas, and plotly. Be sure to import them under the aliases we've used in this project.

[14]:
```
# Import Matplotlib, pandas, and plotly
import matplotlib.pyplot as plt
import pandas as pd
import plotly.express as px
```

1 Prepare Data

In this assignment, you'll work with real estate data from Brazil. In the `data` directory for this project there are two CSV that you need to import and clean.

1.1 Import

Task 1.5.1: Import the CSV file `data/brasil-real-estate-1.csv` into the DataFrame `df1`.

[17]:
```
df1 = pd.read_csv('data/brasil-real-estate-1.csv')
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.1", df1)
```

Before you move to the next task, take a moment to inspect `df1` using the `info` and `head` methods. What issues do you see in the data? What cleaning will you need to do before you can conduct your analysis?

[18]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12834 entries, 0 to 12833
Data columns (total 6 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   property_type            12834 non-null  object
 1   place_with_parent_names  12834 non-null  object
 2   region                   12834 non-null  object
 3   lat-lon                  11551 non-null  object
 4   area_m2                  12834 non-null  float64
 5   price_usd                12834 non-null  object
dtypes: float64(1), object(5)
memory usage: 601.7+ KB
```

Task 1.5.2: Drop all rows with `NaN` values from the DataFrame `df1`.

[19]:
```
df1.dropna(inplace=True)
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11551 entries, 0 to 12833
Data columns (total 6 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   property_type            11551 non-null  object
 1   place_with_parent_names  11551 non-null  object
 2   region                   11551 non-null  object
 3   lat-lon                  11551 non-null  object
 4   area_m2                  11551 non-null  float64
 5   price_usd                11551 non-null  object
dtypes: float64(1), object(5)
memory usage: 631.7+ KB
```

[20]:
```
df1.head()
```

[20]:

	property_type	place_with_parent_names	region	lat-lon	area_m2	price_usd
0	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.6443051,-35.7088142	110.0	$187,230.85
1	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.6430934,-35.70484	65.0	$81,133.37
2	house	|Brasil|Alagoas|Maceió|	Northeast	-9.6227033,-35.7297953	211.0	$154,465.45
3	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.622837,-35.719556	99.0	$146,013.20
4	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.654955,-35.700227	55.0	$101,416.71

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.2", df1)
```

Task 1.5.3: Use the `"lat-lon"` column to create two separate columns in `df1`: `"lat"` and `"lon"`. Make sure that the data type for these new columns is `float`.

[21]:
```
df1[['lat','lon']] = df1['lat-lon'].str.split(',', expand=True)
# display the dataframe
df1 = df1.astype({"lat": float, "lon": float})
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11551 entries, 0 to 12833
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   property_type           11551 non-null  object
 1   place_with_parent_names  11551 non-null  object
 2   region                  11551 non-null  object
 3   lat-lon                 11551 non-null  object
 4   area_m2                 11551 non-null  float64
 5   price_usd               11551 non-null  object
 6   lat                     11551 non-null  float64
 7   lon                     11551 non-null  float64
dtypes: float64(3), object(5)
memory usage: 812.2+ KB
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.3", df1)
```

Task 1.5.4: Use the `"place_with_parent_names"` column to create a `"state"` column for `df1`. (Note that the state name always appears after `"|Brasil|"` in each string.)

[22]:
```
df1['state']=df1['place_with_parent_names'].str.split('|', expand=True)[2]
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.4", df1)
```

Task 1.5.5: Transform the `"price_usd"` column of `df1` so that all values are floating-point numbers instead of strings.

[23]:
```
df1['price_usd']=df1['price_usd'].str.replace('$','',regex=False).str.
 ↪replace(',','').astype(float)
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11551 entries, 0 to 12833
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   property_type           11551 non-null  object
```

```
 1   place_with_parent_names  11551 non-null  object
 2   region                   11551 non-null  object
 3   lat-lon                  11551 non-null  object
 4   area_m2                  11551 non-null  float64
 5   price_usd                11551 non-null  float64
 6   lat                      11551 non-null  float64
 7   lon                      11551 non-null  float64
 8   state                    11551 non-null  object
dtypes: float64(4), object(5)
memory usage: 902.4+ KB
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.5", df1)
```

Task 1.5.6: Drop the `"lat-lon"` and `"place_with_parent_names"` columns from `df1`.

[24]:
```
df1.drop(['lat-lon','place_with_parent_names'],axis="columns",inplace = True)
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.6", df1)
```

```
<b>Good work!</b> You're halfway through your data wrangling. Take a break: Get up from your ma
```

Task 1.5.7: Import the CSV file `brasil-real-estate-2.csv` into the DataFrame `df2`.

[25]:
```
df2 = pd.read_csv('data/brasil-real-estate-2.csv')
```

[ ]:
```
wqet_grader.grade(
    "Project 1 Assessment", "Task 15.7", df2.sort_values("price_brl").head()
)
```

Before you jump to the next task, take a look at `df2` using the `info` and `head` methods. What issues do you see in the data? How is it similar or different from `df1`?

[26]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12833 entries, 0 to 12832
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   property_type  12833 non-null  object
 1   state          12833 non-null  object
 2   region         12833 non-null  object
 3   lat            12833 non-null  float64
 4   lon            12833 non-null  float64
 5   area_m2        11293 non-null  float64
 6   price_brl      12833 non-null  float64
dtypes: float64(4), object(3)
memory usage: 701.9+ KB
```

Task 1.5.8: Use the `"price_brl"` column to create a new column named `"price_usd"`. (Keep in mind that, when this data was collected in 2015 and 2016, a US dollar cost 3.19 Brazilian reals.)

[27]:
```
df2["price_usd"] = df2['price_brl'] / 3.19
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.8", df2)
```

Task 1.5.9: Drop the `"price_brl"` column from `df2`, as well as any rows that have `NaN` values.

[28]:
```
df2.drop("price_brl",axis="columns",inplace = True)
df2= df2.dropna()
df2.head()
```

[28]:

	property_type	state	region	lat	lon	area_m2	price_usd
0	apartment	Pernambuco	Northeast	-8.134204	-34.906326	72.0	129850.463950
1	apartment	Pernambuco	Northeast	-8.126664	-34.903924	136.0	265958.786834
2	apartment	Pernambuco	Northeast	-8.125550	-34.907601	75.0	93867.799373
3	apartment	Pernambuco	Northeast	-8.120249	-34.895920	187.0	265958.786834
4	apartment	Pernambuco	Northeast	-8.142666	-34.906906	80.0	145495.097179

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.9", df2)
```

Task 1.5.10: Concatenate `df1` and `df2` to create a new DataFrame named `df`.

[29]:
```
df = pd.concat([df1, df2])
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22844 entries, 0 to 12832
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   property_type  22844 non-null  object
 1   region         22844 non-null  object
 2   area_m2        22844 non-null  float64
 3   price_usd      22844 non-null  float64
 4   lat            22844 non-null  float64
 5   lon            22844 non-null  float64
 6   state          22844 non-null  object
dtypes: float64(4), object(3)
memory usage: 1.4+ MB
```

[ ]:
```
wqet_grader.grade(
    "Project 1 Assessment", "Task 15.10", df.sort_values("price_usd").head()
)
```

Keep it up! Your data is clean. Take another break. You've earned it.

1.2 Explore

It's time to start exploring your data. In this section, you'll use your new data visualization skills to learn more about the regional differences in the Brazilian real estate market.

Complete the code below to create a `scatter_mapbox` showing the location of the properties in `df`.

[30]:
```
fig = px.scatter_mapbox(
    df,
    lat="lat",
    lon="lon",
    center={"lat": -14.2, "lon": -51.9},  # Map will be centered on Brazil
    width=600,
    height=600,
    hover_data=["price_usd"],  # Display price when hovering mouse over house
)

fig.update_layout(mapbox_style="open-street-map")

fig.show()
```

Task 1.5.11: Use the `describe` method to create a DataFrame `summary_stats` with the summary statistics for the `"area_m2"` and `"price_usd"` columns.

[31]:
```
summary_stats = df[['area_m2','price_usd']].describe()
#summary_stats =summary_stats['area_m2','price_usd']
summary_stats
```

[31]:

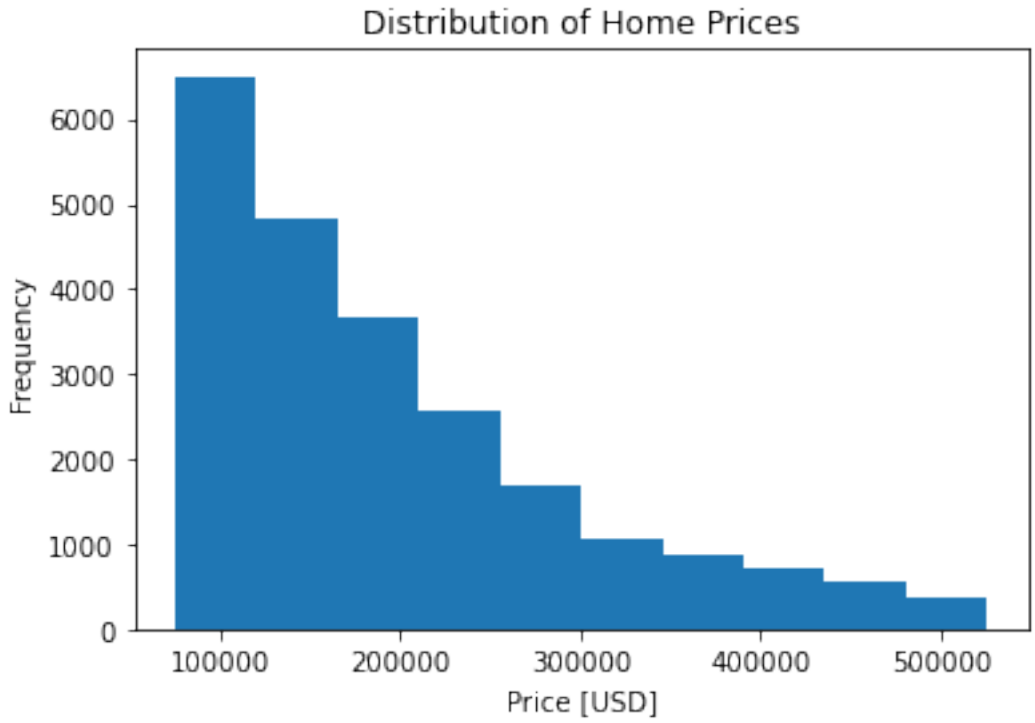
	area_m2	price_usd
count	22844.000000	22844.000000
mean	115.020224	194987.315480
std	47.742932	103617.682978
min	53.000000	74892.340000

```
25%        76.000000   113898.770000
50%       103.000000   165697.555000
75%       142.000000   246900.880878
max       252.000000   525659.717868
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.11", summary_stats)
```

Task 1.5.12: Create a histogram of `"price_usd"`. Make sure that the x-axis has the label `"Price [USD]"`, the y-axis has the label `"Frequency"`, and the plot has the title `"Distribution of Home Prices"`.

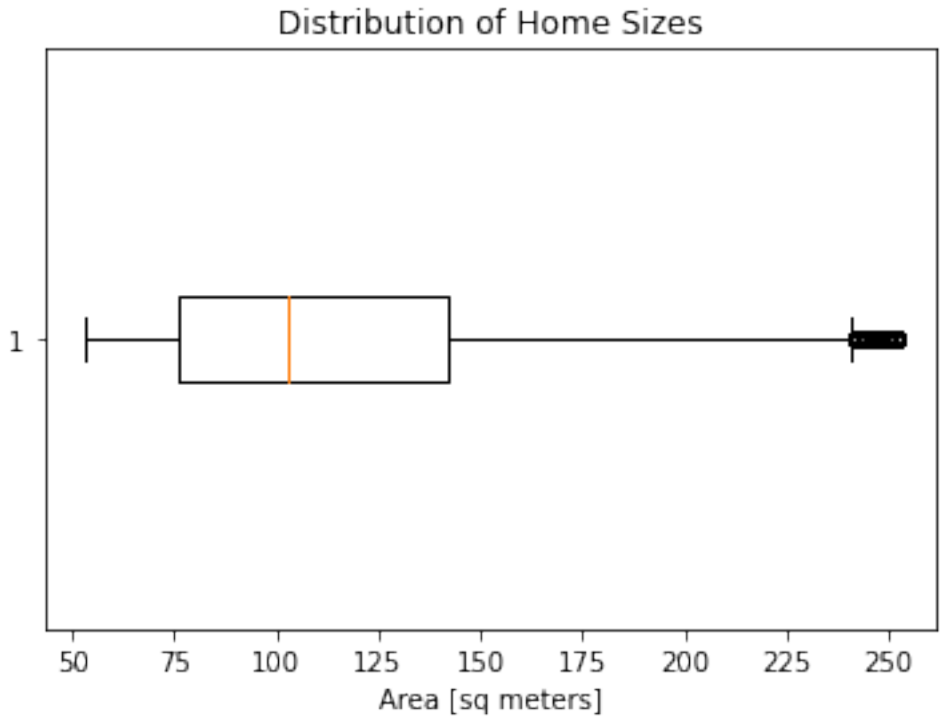
[32]:
```
plt.hist(df["price_usd"])
plt.xlabel("Price [USD]")
plt.ylabel("Frequency")
plt.title("Distribution of Home Prices")

# Don't change the code below
plt.savefig("images/15-12.png", dpi=150)
```



[ ]:
```
with open("images/15-12.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 15.12", file)
```

Task 1.5.13: Create a horizontal boxplot of `"area_m2"`. Make sure that the x-axis has the label `"Area [sq meters]"` and the plot has the title `"Distribution of Home Sizes"`.

[33]:
```
plt.boxplot(df["area_m2"],vert=0)
plt.xlabel("Area [sq meters]")
plt.title("Distribution of Home Sizes")
# Don't change the code below
plt.savefig("images/15-13.png", dpi=150)
```



[ ]:
```
with open("images/15-13.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 15.13", file)
```

Task 1.5.14: Use the `groupby` method to create a Series named `mean_price_by_region` that shows the mean home price in each region in Brazil, sorted from smallest to largest.

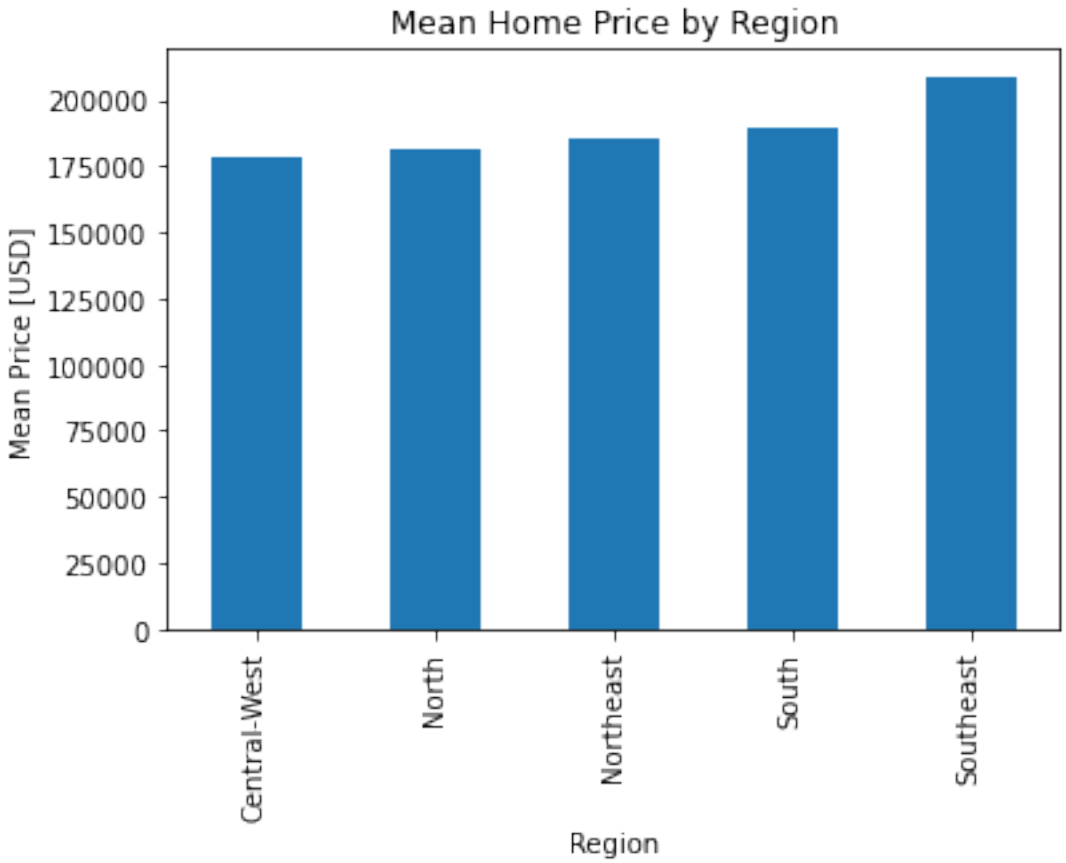
[34]:
```
mean_price_by_region = df.groupby('region')['price_usd'].mean()
mean_price_by_region
```

[34]:
```
region
Central-West    178596.283663
North           181308.958207
Northeast       185422.985441
South           189012.345265
Southeast       208996.762778
Name: price_usd, dtype: float64
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.14", mean_price_by_region)
```

Task 1.5.15: Use `mean_price_by_region` to create a bar chart. Make sure you label the x-axis as `"Region"` and the y-axis as `"Mean Price [USD]"`, and give the chart the title `"Mean Home Price by Region"`.

[35]:
```
mean_price_by_region.plot(

    kind= 'bar',
    xlabel = 'Region',
    ylabel = 'Mean Price [USD]',
    title = 'Mean Home Price by Region'
)

# Don't change the code below
plt.savefig("images/15-15.png", dpi=150)
```



[ ]:
```
with open("images/15-15.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 15.15", file)
```

Keep it up! You're halfway through your data exploration. Take one last break and get re

You're now going to shift your focus to the southern region of Brazil, and look at the relationship

between home size and price.

Task 1.5.16: Create a DataFrame `df_south` that contains all the homes from `df` that are in the `"South"` region.

[36]:
```
df_south = df[df['region'] =='South']
df_south.head()
```

[36]:

	property_type	region	area_m2	price_usd	lat	lon	state
9304	apartment	South	127.0	296448.85	-25.455704	-49.292918	Paraná
9305	apartment	South	104.0	219996.25	-25.455704	-49.292918	Paraná
9306	apartment	South	100.0	194210.50	-25.460236	-49.293812	Paraná
9307	apartment	South	77.0	149252.94	-25.460236	-49.293812	Paraná
9308	apartment	South	73.0	144167.75	-25.460236	-49.293812	Paraná

[ ]:
```
wqet_grader.grade(
    "Project 1 Assessment", "Task 15.16", df_south.sort_values("area_m2").head()
)
```

Task 1.5.17: Use the `value_counts` method to create a Series `homes_by_state` that contains the number of properties in each state in `df_south`.

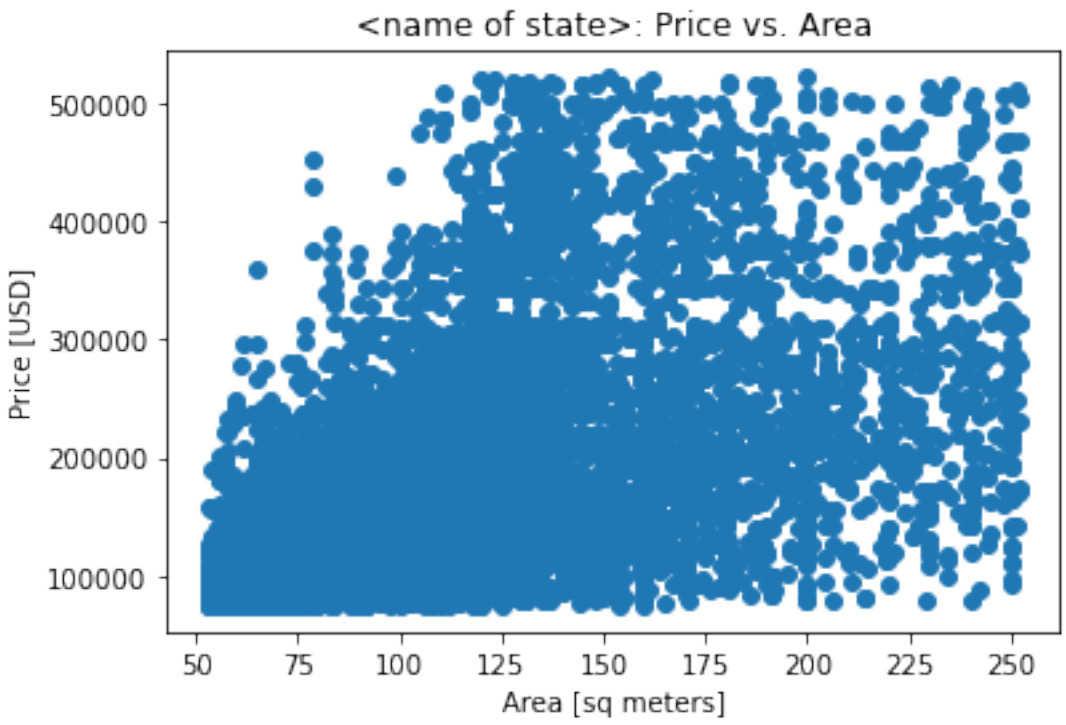
[37]:
```
homes_by_state = df_south.state.value_counts()
homes_by_state
```

[37]:
```
Rio Grande do Sul    2643
Santa Catarina       2634
Paraná               2544
Name: state, dtype: int64
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.17", homes_by_state)
```

Task 1.5.18: Create a scatter plot showing price vs. area for the state in `df_south` that has the largest number of properties. Be sure to label the x-axis `"Area [sq meters]"` and the y-axis `"Price [USD]"`; and use the title `"<name of state>: Price vs. Area"`.

[38]:
```
plt.scatter(df_south['area_m2'], df_south['price_usd'])
plt.xlabel('Area [sq meters]')
plt.ylabel('Price [USD]')
plt.title('<name of state>: Price vs. Area')
# Don't change the code below
plt.savefig("images/15-18.png", dpi=150)
```



[ ]:
```
with open("images/15-18.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 15.18", file)
```

Task 1.5.19: Create a dictionary `south_states_corr`, where the keys are the names of the three states in the `"South"` region of Brazil, and their associated values are the correlation coefficient between `"area_m2"` and `"price_usd"` in that state.

As an example, here's a dictionary with the states and correlation coefficients for the Southeast region. Since you're looking at a different region, the states and coefficients will be different, but the structure of the dictionary will be the same.

```
{'Espírito Santo': 0.6311332554173303,
 'Minas Gerais': 0.5830029036378931,
 'Rio de Janeiro': 0.4554077103515366,
 'São Paulo': 0.45882050624839366}
```

[39]:
```
key=df_south.state.unique()
a=df_south[df_south['state']=='Paraná']
b=df_south[df_south['state']=='Rio Grande do Sul']
c=df_south[df_south['state']=='Santa Catarina']
p=a['area_m2'].corr(a['price_usd'])
r=b['area_m2'].corr(b['price_usd'])
s=c['area_m2'].corr(c['price_usd'])
```

[40]:
```
south_states_corr = {'Paraná':p,'Rio Grande do Sul':r,'Santa Catarina':s}

south_states_corr
```

[40]:
```
{'Paraná': 0.5436659935502659,
 'Rio Grande do Sul': 0.5773267433717683,
 'Santa Catarina': 0.5068121776366781}
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 15.19", south_states_corr)
```
