

Tim Elvira

CS540

Applying Linear Regression to the Sales Analysis Table

Connecting to PGAdmin and selecting an area to analyze:

```
In [149]: import psycopg2
import psycopg2.extras

try:
    conn = psycopg2.connect("dbname='spatial' user='postgres' host='localhost' password='arcsEkkonds22@'")
except Exception as E:
    print(E)
    exit(1)
curr = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
sql = "select s.parid, s.aprland, s.aprbldg, s.aprtot, s. price, o.addr1 from volusia.sales_analysis s join volusia.owner o on s.parid=o.parid"
df2 = pd.read_sql_query(sql, conn)

corr2 = df2.corr()
#print(df2.head())
print(corr2)

print(df2.corr().abs().nlargest(3, 'price').index)

get_ipython().run_line_magic('matplotlib', 'inline')

##Total area gave me a NAN error so I had to go to the next one and that was aprland

parid    aprland    aprbldg    aprtot    price
parid    1.000000  -0.100189    0.279986    0.141362    0.130514
aprland  -0.100189    1.000000    0.493688    0.809428    0.794610
aprbldg   0.279986    0.493688    1.000000    0.910274    0.863590
aprtot    0.141362    0.809428    0.910274    1.000000    0.961423
price     0.130514    0.794610    0.863590    0.961423    1.000000
Index(['price', 'aprtot', 'aprbldg'], dtype='object')
```

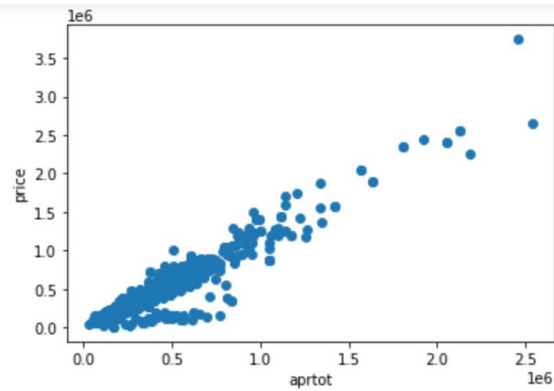
```
In [150]: plt.scatter(df2['aprtot'],df2['price'],marker='o')
plt.xlabel('aprtot')
plt.ylabel('price')
```

```
Out[150]: Text(0, 0.5, 'price')
```

Connecting to PGAdmin in order to query the Sales Analysis table. This is done right into a Pandas dataframe and is shown below the initial block. This block is showing the correlation between the different categories with the most important ones being Price, Aprtot, and apr bldg.

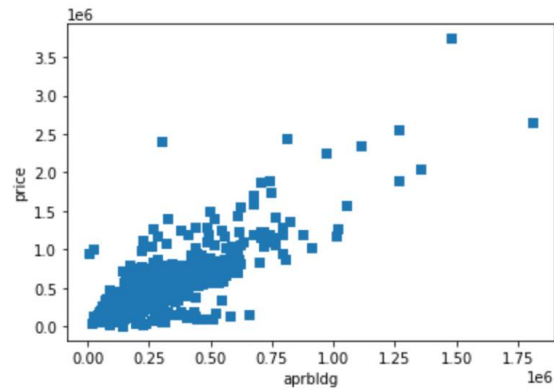
The query selects only valid parcels (joined with the volusia.owners table) from the port orange area that have the zip codes: '32119', '32123', '32127', '32128', and '32129' and luc = '0100' and has_pool = 1. This is to isolate single-family homes with a pool in port orange. More attributes (5) will be added to show the changing of price prediction and analyzing how the predicted price is changed.

Viewing the Multiple Regression



```
In [151]: ▶ plt.scatter(df2['aprbldg'],df2['price'],marker='s')
           plt.xlabel('aprbldg')
           plt.ylabel('price')
```

```
Out[151]: Text(0, 0.5, 'price')
```



The Price regression graphs above show the correlation of X to Y. The X correspond to the APRBLDG, or the total value of the buildings at that parcel, and the aprtot, or the total value of the land and building of that parcel. APRTOT is shown to be more correlated than APRBLDG.

Initial Sales Analysis Model Training

```
In [152]: x = pd.DataFrame(np.c_[df2['aprtot'], df2['aprblgd']], columns = ['aprtot', 'aprblgd'])  
Y = pd.DataFrame(df2['price'], columns = ['price'])
```

```
In [153]: from sklearn.model_selection import train_test_split  
x_train, x_test, Y_train, Y_test = train_test_split(x, Y, test_size = 0.3,  
                                                    random_state=5)
```

```
In [154]: print(x_train.shape)  
print(Y_train.shape)
```

```
(6544, 2)  
(6544, 1)
```

```
In [155]: print(x_test.shape)  
print(Y_test.shape)
```

```
(2805, 2)  
(2805, 1)
```

```
In [156]: from sklearn.linear_model import LinearRegression  
  
model = LinearRegression()  
model.fit(x_train, Y_train)
```

```
Out[156]: LinearRegression()
```

```
In [157]: price_pred = model.predict(x_test)
```

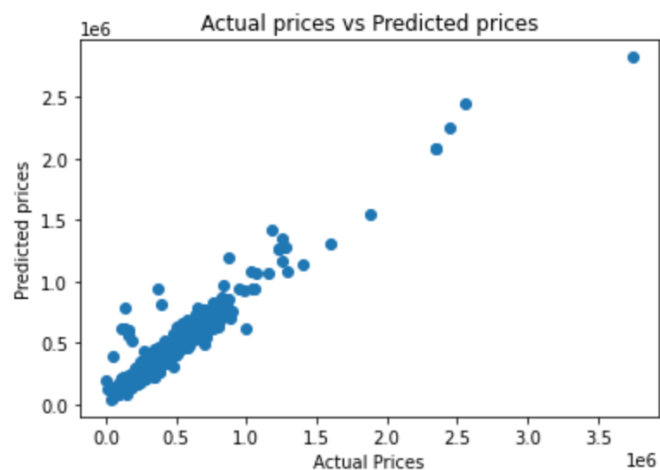
```
In [158]: print('R-squared: %.4f' % model.score(x_test,  
                                                Y_test))
```

```
R-squared: 0.9295
```

Shown is the creation of the X and Y splits. The model is trained by associating all X attributes to result into some Y attribute. The X split is the aprtot and aprblgd with the Y being the price. The test size is to get the R-Squared value which is the performance of the model which sits at around 0.92 or 92% prediction accuracy.

```
22700743 / 2.070004
```

```
Out[159]: Text(0.5, 1.0, 'Actual prices vs Predicted prices')
```



Comparing prices to 5 Zillow properties:

```
In [160]: ▶ print(model.intercept_)
          print(model.coef_)

[5786.14379812]
[[ 1.2057604 -0.09503547]]

In [161]: ▶ ##- 1757 Savannah Ln pt Or
          zillowprice1 = 300739

          ## 52 Woofield Dr
          zillowprice2 = 228138

          ## - 1139 E Willow Run Dr
          zillowprice3 = 279470

          ##- 5292 Bear Corn Run
          zillowprice4 = 325000

          ##1883 Cody Ct
          zillowprice5 = 389900

          ###Verifying this with pgAdmin queries to get the x values
          print("Model difference")
          print((model.predict([[232569,196069]])) - zillowprice1)
          print((model.predict([[172605,148548]])) - zillowprice2)
          print((model.predict([[172211,143411]])) - zillowprice3)
          print((model.predict([[211516,179016]])) - zillowprice4)
          print((model.predict([[243008,214008]])) - zillowprice5)

Model difference
[[-33163.87690778]]
[[-28348.91255956]]
[[-79667.78492897]]
[[-81189.11059601]]
[[-111442.78549119]]
```

The model under predicts the price but this could be because there is an attribute that is raising the price and could change the model performance.

Adding new attributes

Nearest Elementary, Middle, and Highschool for the Port Orange Zip Codes

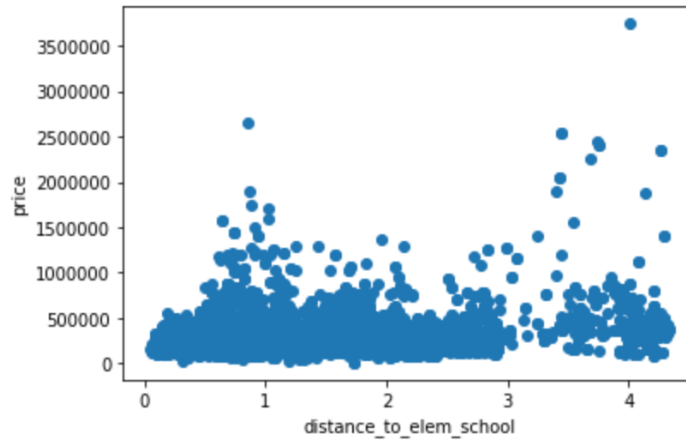
```
parid      parid  aprland  aprbldg  apttot   price  \
parid      1.000000 -0.100189  0.279986  0.141362  0.130514
aprland    -0.100189  1.000000  0.493688  0.809428  0.794610
aprbldg     0.279986  0.493688  1.000000  0.910274  0.863590
aprtot      0.141362  0.809428  0.910274  1.000000  0.961423
price       0.130514  0.794610  0.863590  0.961423  1.000000
distance_to_elem_school -0.061806  0.272655  0.194698  0.261271  0.237843
distance_to_middle_school -0.052991  0.248171  0.057758  0.157149  0.133200
distance_to_high_school  0.212645  0.385565  0.369217  0.432866  0.405332

distance_to_elem_school  distance_to_middle_school  \
parid      -0.061806      -0.052991
aprland      0.272655      0.248171
aprbldg      0.194698      0.057758
aprtot      0.261271      0.157149
price       0.237843      0.133200
distance_to_elem_school  1.000000      0.588282
distance_to_middle_school 0.588282      1.000000
distance_to_high_school  0.628385      0.396697

distance_to_high_school
parid      0.212645
aprland      0.385565
aprbldg      0.369217
aprtot      0.432866
price       0.405332
distance_to_elem_school  0.628385
distance_to_middle_school 0.396697
distance_to_high_school  1.000000
Index(['price', 'aprtot', 'aprbldg'], dtype='object')
```

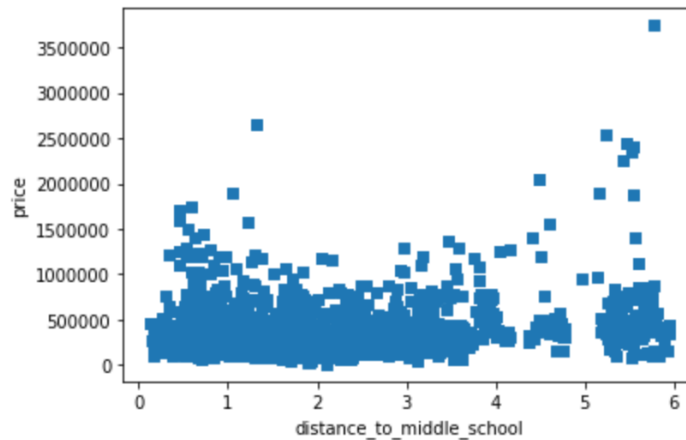
Adding the distance to elementary, middle, and high school. Interestingly, High school has a higher correlation to price than elementary and middle. Elementary schools are more diverse and could have a higher property price around middle schools.

```
Out[8]: Text(0, 0.5, 'price')
```



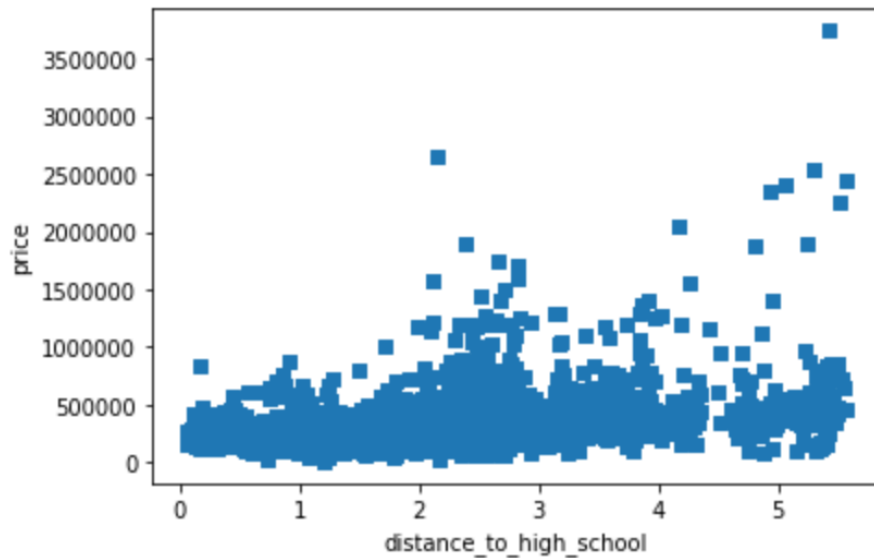
```
In [9]: ▶ plt.scatter(df2['distance_to_middle_school'],df2['price'],marker='s')
plt.xlabel('distance_to_middle_school')
plt.ylabel('price')
```

```
Out[9]: Text(0, 0.5, 'price')
```



The distance to middle school and elementary school do not seem to be correlated and have varying prices at all distances. Compared to high school which shows a better correlation and a slight curve towards a positive slope regression line.

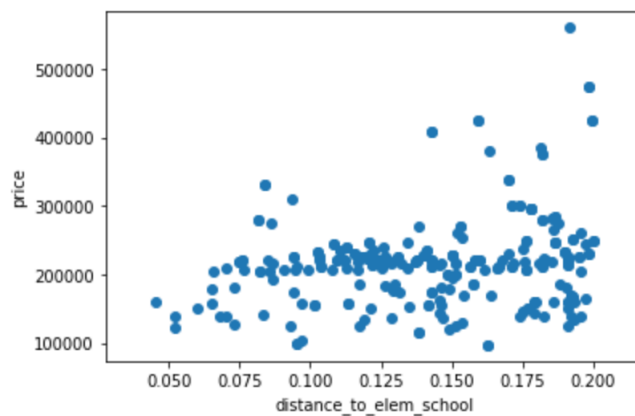
```
Out[11]: Text(0, 0.5, 'price')
```



Since this data is particularly for Single family houses with a pool, there could be a distinct change where prices of a house change within 0.4 and 0.2 of a mile, particularly with middle schools.

```
In [13]: plt.scatter(df2['distance_to_elem_school'],df2['price'],marker='o')
plt.xlabel('distance_to_elem_school')
plt.ylabel('price')
```

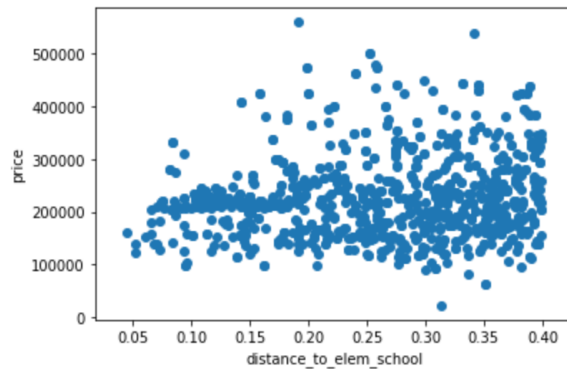
```
Out[13]: Text(0, 0.5, 'price')
```



Here is the data of all the houses that are within 0.2 of a mile of an elementary school. There is a distinct trend line that sits roughly at the 200000+ price of a house.

```
In [16]: ▶ plt.scatter(df2['distance_to_elem_school'],df2['price'],marker='o')
plt.xlabel('distance_to_elem_school')
plt.ylabel('price')
```

Out[16]: Text(0, 0.5, 'price')



Within 0.4 of a mile, that trendline becomes more obfuscated and the data seems to be more spread out in terms of price. This is an interesting find, as you move closer to elementary schools, the price of a house plays more of a role.

Training model on distance to ES, MS, and HS

```
77]: ▶ print(x_train.shape)
print(Y_train.shape)
```

```
(6544, 5)
(6544, 1)
```

```
78]: ▶ print(x_test.shape)
print(Y_test.shape)
```

```
(2805, 5)
(2805, 1)
```

```
79]: ▶ from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train, Y_train)
```

Out[79]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

```
80]: ▶ price_pred = model.predict(x_test)
```

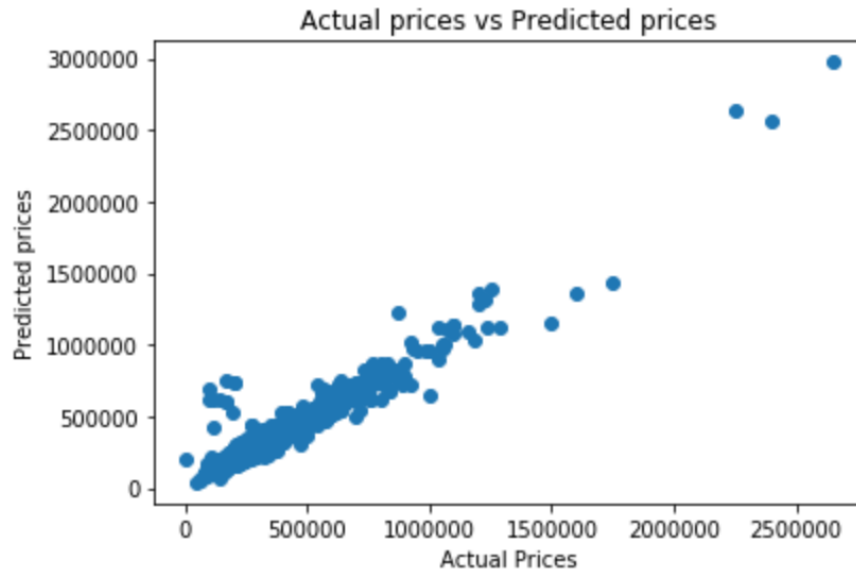
```
81]: ▶ print('R-squared: %.4f' % model.score(x_test,
Y_test))
```

R-squared: 0.9221

After training on the 3 new attributes, the R squared changes very slightly from 92.8% to 92.6%. This could represent the little correlation the school distances have on the price of the property.

```
2179041866.7463923
```

```
Out[82]: Text(0.5, 1.0, 'Actual prices vs Predicted prices')
```



Model view of the performance between actual and predicted prices. Very little change occurred from adding these 3 new attributes. This could be due to the prices not necessarily having much correlation on prices or not enough data to show that the prices can be affected by distance from schools. The prices of the Zillow houses changed very slightly, plus or minus a few hundred dollars.

Adding 5 new attributes (Classmates)

Finally, 5 new attributes are added which include: distance to a marina or boat ramp, distance to a crime, distance to a flood zone, distance to a railroad, and distance to a fire station. In order to see how the model changes based on these additions, the model must be retrained to retest it on the Zillow houses.

Training the Model

```
In [109]: x = pd.DataFrame(np.c_[df2['aprtot'], df2['aprbldg'], df2['distance_to_elem_school'], df2['distance_to_middle_school'], df2['distance_to_marina'], df2['distance_to_boat_ramp'], df2['distance_to_flood_zone'], df2['distance_to_crime'], df2['distance_to_railroad_track'], df2['distance_to_fire_station']], columns = ['price'])
Y = pd.DataFrame(df2['price'], columns = ['price'])
df2.round(decimals = 2)
df2.fillna(value = 0,
            inplace = True)

In [110]: from sklearn.model_selection import train_test_split
x_train, x_test, Y_train, Y_test = train_test_split(x, Y, test_size = 0.3,
                                                    random_state=5)

In [111]: print(x_train.shape)
print(Y_train.shape)

(6544, 11)
(6544, 1)

In [112]: print(x_test.shape)
print(Y_test.shape)

(2805, 11)
(2805, 1)
```

The model is further trained on 11 attributes which are the 2 prevalent models, 3 distance to schools, Distance to Marina, Distance to boat ramp, distance to flood zone, distance to a crime, distance to a railroad track, and distance to a fire station.

2197634913.606927

Out[116]: Text(0.5, 1.0, 'Actual prices vs Predicted prices')



Getting the Intercept and Coefficients

```
In [117]: print(model.intercept_)
print(model.coef_)

[57969.03155118]
[[ 1.28793737e+00 -1.83265388e-01 -1.24183736e+03 -1.04446924e+03
  -2.01291618e+03  5.22979379e+03  2.54428721e+03  5.69123502e+03
  -8.78053549e+05 -5.56547682e+03  1.19822663e+03]]
```

The model is roughly the same in terms of R-Squared but the coefficients are interesting. The most heavily weighted coefficients are marina distance, crime distance, flood zone distance (negatively

affecting) and railroad distance (negatively affect). This is fascinating because this shows that the closer you are to crime and railroads, the lower your property price is affected. Furthermore, the closer you are to a marina and boat ramp, they more property price you have. This could because your property is closer to water.

```
##- 1757 Savannah Ln pt Or
zillowprice1 = 300739

## 52 Woofield Dr
zillowprice2 = 228138

## - 1139 E Willow Run Dr
zillowprice3 = 279470

##- 5292 Bear Corn Run
zillowprice4 = 325000

##1883 Cody Ct
zillowprice5 = 389900

###Verifying this with pgAdmin queries to get the x values
print("Model difference")
print((model.predict([[232569,196069,1.05,2.55,2.44,4.63,2.38,0.016,0.066,4.15,0.17]])) - zillowprice1)
print((model.predict([[172605,148548,0.96,0.75,1.14,2.77,2.70,0.21,0.066,2.36,0.48]])) - zillowprice2)
print((model.predict([[172211,143411,0.81,0.69,1.45,2.76,2.35,0.059,0.066,4.22,0.14]])) - zillowprice3)
print((model.predict([[211516,179016,1.01,2.50,2.44,4.58, 2.38,0.016,0.066,4.12,0.24]])) - zillowprice4)
print((model.predict([[243008,214008,1.82,2.09,2.54,5.36,1.70,0.14,0.066,4.23,1.02]])) - zillowprice5)

Model difference
[[-38531.32117034]]
[[-27318.13319457]]
[[-91152.56536386]]
[[-86690.79449455]]
[[-114845.58951318]]
```

Predicting the Zillow prices shows little change, primarily due to the R-Squared value having a small change. In the case of adding 5 more attributes, the R-squared went up by 0.2%. The change in prices are not drastic but change in $\pm 6k$.

Lessons learned:

This project taught me many advanced ways to query and manipulate a database to fit my data needs. The ability to query, loop, and update helped me tremendously while trying to populate the school data and incorporate other peers' data into my own tables. While I had much experience with linear regression, this project also helped me understand the invariability and non-deterministic nature of machine learning. By adding multiple attributes, I saw how much (or little) the R-Squared value and the predicted Zillow prices changed. If I could redo the project, I might try to attempt a different area of Volusia county or change the property to be more open of a wide arrange of characteristics. Limiting my dataset to only Port Orange and single-family homes with a pool might have limited my dataset and could have overfitted it when trying to predict the Zillow homes. However, I think the size of my dataset was just right (6k/3k train/test split) and would aim for that dataset region for the next trial. Along side the SQL and ML skills, I learned about Jupyter and its easy usability which I plan on incorporating my other projects. Furthermore, the ability to visualize our data through QGIS gave a visual meaning to the data we were processing and almost a tangible purpose to the numerics we were seeing in pgAdmin. Overall, this class was a great learning experience that gave me a wide breadth of knowledge from all the programs and languages I was exposed to.