

```
# Using google colab - this first step is for loading in the data from my personal Drive

# Login with google credentials

from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)

# Handle errors from too many requests

import logging
logging.getLogger('googleapiclient.discovery_cache').setLevel(logging.ERROR)

# The ID for my personal Drive folder is 1BVUuroPvozFxmJMIYrGOFtI4r6erSBCx
# I am now listing the ID numbers for the files in this folder to find the data files

#file_list = drive.ListFile({'q': "'1BVUuroPvozFxmJMIYrGOFtI4r6erSBCx' in parents and
#for file1 in file_list:
# print('title: %s, id: %s' % (file1['title'], file1['id'])))

# Data ID: 1F2KojI0d-ZnN8ssQFUWSyZA8I0mAgMEf

# Now that I have the ID files, load the files

data_downloaded = drive.CreateFile({'id': '1F2KojI0d-ZnN8ssQFUWSyZA8I0mAgMEf'})
data_downloaded.GetContentFile('Full.csv')

# Load the data into pandas

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. Save and reload the page. ✕

print(data.isnull().values.any())

# No nulls after import

False

data.head(3)
```



| | Year_sold | PARCEL | SequenceNum | SaleDate | SalePrice | PrPerSqFt | p_Cat | Record |
|---|-----------|-----------|-------------|----------|-----------|-----------|-------|--------|
| 0 | 2019 | 401554180 | 20191960498 | 201906 | 49000 | 61.790668 | 6 | |
| 1 | 2019 | 401554100 | 20190300552 | 201901 | 50500 | 63.682219 | 6 | |
| 2 | 2019 | 401553650 | 20191970632 | 201907 | 58000 | 63.112078 | 6 | |

```

import seaborn as sn
corrMat = data.corr()

#plt.figure(figsize=(30,15))
#sn.heatmap(corrMat, annot=True)

# SalesPrice correlates heavily with CLASS, PrPerSqFt, BATHFIXTUR, MAIN, CONTROL, ACTU
# Years_sold, SequenceNum, SalesDate are all the same
# MAIN, CONTROL, ACTUAL are all about the same

# Feature Engineering

## Years_sold, SequenceNum, SalesDate and RecordingDate all contain basically the same
data = data.drop(columns=['Year_sold', 'SequenceNum'], axis=1)

## Main, Control, Actual are all tax assessments. Let's average them and use thart ins
data['TaxEval'] = (data['MAIN'] + data['CONTROL'] + data['ACTUAL'] ) / 3

data['diff'] = data['SalePrice'] - data['TaxEval']

data = data.drop(columns=['MAIN', 'CONTROL', 'ACTUAL'], axis=1)

## PrPerSqFt is just SalesPrice / Sqft. There's no reason to include that along with s
data = data.drop(columns=[ 'PARCEL', 'INSPECTION'], axis=1)

features = list(data.columns.values.tolist())

corrMat = data[features].corr()

plt.figure(figsize=(30,15))
sn.heatmap(corrMat, annot=True)

```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



```
# Now remove features we can't use in the model
```

```
features.remove('diff')  
features.remove('PrPerSqFt')  
features.remove('TaxEval')
```



Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



| | | | | | | | | | | | | | | | | | | | |
|----------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|------|
| SaleDate | 1 | 0.023 | 0.033 | 0.051 | 0.94 | 0.027 | 0.012 | 0.001 | 0.019 | 0.011 | -0.005 | 0.044 | 0.14 | -0.019 | 0.008 | 0.014 | -0.03 | -0.018 | 0.0 |
| SalePrice | 0.023 | 1 | 0.72 | 0.21 | 0.048 | 0.58 | 0.042 | 0.39 | 0.23 | -0.085 | 0.085 | -0.13 | -0.21 | 0.56 | -0.18 | 0.32 | 0.31 | 0.65 | 0.2 |
| PrPerSqFt | 0.033 | 0.72 | 1 | 0.19 | 0.054 | 0.18 | -0.069 | -0.03 | 0.049 | 0.038 | 0.041 | -0.021 | -0.04 | 0.086 | -0.046 | 0.076 | 0.068 | 0.052 | 0.0 |
| p_Cat | 0.051 | 0.21 | 0.19 | 1 | 0.066 | 0.18 | -0.055 | -0.015 | 0.052 | -0.011 | 0.037 | -0.027 | -0.058 | 0.088 | -0.041 | 0.073 | 0.088 | 0.069 | 0.0 |
| RecordingDate | 0.94 | 0.048 | 0.054 | 0.066 | 1 | -0.012 | 0.021 | 0.006 | -0.010 | 0.006 | 0.005 | 0.037 | 0.13 | 0.002 | 0.003 | 0.009 | 0.010 | 0.018 | 0.0 |
| CLASS | -0.027 | 0.58 | 0.18 | 0.18 | -0.012 | 1 | 0.017 | 0.38 | 0.15 | -0.2 | 0.07 | -0.26 | -0.33 | 0.6 | -0.25 | 0.34 | 0.47 | 0.63 | 0.4 |
| STORIES | -0.012 | 0.042 | -0.069 | -0.055 | 0.021 | 0.017 | 1 | 0.31 | 0.23 | -0.27 | 0.064 | -0.065 | -0.16 | 0.29 | -0.054 | 0.079 | 0.27 | 0.25 | 0.3 |
| ROOMS | -0.001 | 0.39 | -0.03 | -0.015 | 0.006 | 0.38 | 0.31 | 1 | 0.28 | -0.18 | 0.062 | -0.2 | -0.25 | 0.63 | -0.22 | 0.29 | 0.38 | 0.74 | 0.3 |
| QUALITY | -0.019 | 0.23 | 0.049 | 0.052 | -0.01 | 0.15 | 0.23 | 0.28 | 1 | -0.21 | 0.11 | -0.2 | -0.31 | 0.35 | -0.19 | 0.19 | 0.55 | 0.33 | 0.4 |
| WALLS | -0.011 | -0.085 | 0.038 | 0.011 | 0.006 | -0.2 | -0.27 | -0.18 | -0.21 | 1 | -0.066 | 0.032 | 0.29 | -0.31 | 0.13 | -0.081 | -0.43 | -0.15 | -0.6 |
| ROOF | -0.005 | 0.085 | 0.041 | 0.037 | 0.005 | 0.07 | 0.064 | 0.062 | 0.11 | -0.066 | 1 | 0.15 | -0.002 | 0.088 | -0.074 | 0.075 | 0.049 | 0.097 | 0.0 |
| HEAT | -0.044 | -0.13 | -0.021 | -0.027 | 0.037 | -0.26 | -0.065 | -0.2 | -0.2 | 0.032 | 0.15 | 1 | 0.38 | -0.25 | 0.11 | -0.085 | -0.29 | -0.2 | -0.1 |
| COOL | -0.14 | -0.21 | -0.04 | -0.058 | 0.13 | -0.33 | -0.16 | -0.25 | -0.31 | 0.29 | -0.002 | 0.38 | 1 | -0.36 | 0.19 | -0.17 | -0.46 | -0.3 | -0.4 |
| BATHFIXTUR | -0.015 | 0.56 | 0.086 | 0.088 | 0.002 | 0.6 | 0.29 | 0.63 | 0.35 | -0.31 | 0.088 | -0.25 | -0.36 | 1 | -0.25 | 0.34 | 0.51 | 0.78 | 0.5 |
| PATIO | -0.008 | -0.18 | -0.046 | -0.041 | 0.003 | -0.25 | -0.054 | -0.22 | -0.19 | 0.13 | -0.074 | 0.11 | 0.19 | -0.25 | 1 | -0.69 | -0.25 | -0.26 | -0.2 |
| PATIONUMBE | -0.014 | 0.32 | 0.076 | 0.073 | 0.009 | 0.34 | 0.079 | 0.29 | 0.19 | -0.081 | 0.075 | -0.085 | -0.17 | 0.34 | -0.69 | 1 | 0.24 | 0.41 | 0.1 |
| CONDITION | -0.03 | 0.31 | 0.068 | 0.088 | -0.013 | 0.47 | 0.27 | 0.38 | 0.55 | -0.43 | 0.049 | -0.29 | -0.46 | 0.51 | -0.25 | 0.24 | 1 | 0.44 | 0.7 |
| SQFT | -0.018 | 0.65 | 0.052 | 0.069 | 0.001 | 0.63 | 0.25 | 0.74 | 0.33 | -0.15 | 0.097 | -0.2 | -0.3 | 0.78 | -0.26 | 0.41 | 0.44 | 1 | 0.3 |
| YEAR | -0.028 | 0.23 | 0.006 | 0.035 | 0.003 | 0.41 | 0.34 | 0.36 | 0.41 | -0.61 | 0.041 | -0.3 | -0.45 | 0.54 | -0.23 | 0.19 | 0.71 | 0.39 | 1 |
| GARAGE | -0.008 | -0.21 | -0.037 | -0.063 | 0.006 | 0.37 | -0.19 | -0.26 | -0.32 | 0.35 | 0.008 | 0.26 | 0.35 | -0.41 | 0.21 | -0.16 | -0.49 | -0.29 | -0.5 |
| GARAGECAPA | -0.011 | 0.38 | 0.071 | 0.095 | 0.007 | 0.49 | 0.17 | 0.43 | 0.34 | -0.33 | 0.037 | -0.24 | -0.35 | 0.56 | -0.25 | 0.27 | 0.51 | 0.53 | 0.5 |
| POOLAREA | -0.014 | 0.31 | 0.069 | 0.078 | 0.01 | 0.27 | -0.023 | 0.3 | 0.099 | 0.12 | 0.005 | 0.081 | -0.094 | 0.28 | -0.11 | 0.18 | 0.095 | 0.39 | -0.0 |
| LASTACTION | -0.16 | 0.14 | 0.038 | 0.041 | -0.14 | 0.12 | 0.13 | 0.12 | 0.13 | -0.18 | 0.067 | -0.12 | -0.44 | 0.18 | -0.15 | 0.18 | 0.22 | 0.19 | 0.1 |
| GISACRES | -0.008 | 0.29 | 0.09 | 0.09 | -0.006 | 0.22 | -0.06 | 0.14 | 0.077 | 0.085 | 0.007 | 0.16 | -0.021 | 0.19 | -0.072 | 0.2 | 0.078 | 0.31 | 0.0 |
| LON | -0.005 | 0.078 | 0.04 | 0.030 | 0.005 | 0.078 | 0.014 | 0.11 | 0.1 | 0.084 | 0.003 | -0.1 | -0.092 | 0.068 | -0.05 | 0.046 | 0.038 | 0.089 | -0.0 |
| LAT | -0.002 | 0.17 | 0.11 | 0.11 | -0.001 | 0.11 | -0.07 | 0.056 | -0.12 | 0.08 | -0.012 | 0.001 | 0.029 | 0.087 | 0.015 | -0.026 | -0.018 | 0.086 | -0.0 |
| ZIP | -0.005 | 0.068 | 0.057 | -0.026 | 0.002 | 0.074 | 0.016 | 0.017 | -0.01 | 0.021 | -0.033 | 0.008 | 0.021 | -0.032 | 0.024 | -0.026 | -0.031 | 0.026 | 0.0 |
| NonPrimary | -0.017 | 0.016 | 0.008 | 0.039 | 0.004 | 0.057 | -0.048 | -0.063 | -0.02 | 0.004 | 0.026 | 0.025 | 0.005 | 0.013 | 0.005 | 0.002 | 0.006 | 0.014 | 0.0 |
| PrimaryRes | -0.014 | 0.072 | 0.008 | 0.037 | 0.003 | 0.092 | 0.095 | 0.15 | 0.12 | -0.096 | 0.014 | -0.13 | -0.15 | 0.14 | -0.067 | 0.07 | 0.16 | 0.14 | 0.0 |
| Rental | -0.000 | 0.11 | 0.001 | 0.081 | -0.012 | -0.16 | -0.075 | -0.13 | -0.13 | 0.11 | -0.006 | 0.13 | 0.18 | -0.17 | 0.086 | -0.089 | -0.19 | -0.16 | -0.2 |
| JointTenancyDeed | -0.018 | -0.02 | -0.014 | 0.011 | -0.027 | 0.025 | 0.003 | 0.016 | -0.026 | 0.021 | 0.005 | 0.001 | 0.008 | -0.02 | 0.017 | -0.012 | -0.03 | -0.017 | 0.0 |
| WarrantyDeed | -0.011 | 0.024 | 0.023 | 0.017 | 0.017 | 0.022 | 0.003 | 0.013 | 0.022 | -0.018 | 0.003 | 0.000 | 0.014 | 0.012 | -0.018 | 0.006 | 0.027 | 0.01 | 0.0 |
| FinancingCash | -0.009 | 0.035 | 0.049 | 0.000 | 0.036 | -0.09 | -0.085 | -0.032 | 0.021 | 0.041 | 0.067 | 0.052 | -0.036 | 0.009 | 0.002 | -0.049 | 0.031 | -0.0 | 0.0 |
| isSolar | -0.009 | 0.007 | 0.001 | 0.006 | 0.013 | 0.009 | 0.007 | 0.02 | 0.001 | -0.01 | 0.000 | 0.007 | 0.009 | 0.016 | 0.004 | 0.000 | 0.005 | 0.011 | 0.0 |
| GoodSale | -0.018 | -0.058 | -0.018 | -0.011 | -0.013 | -0.096 | 0.008 | 0.011 | -0.028 | 0.049 | -0.071 | -0.041 | -0.012 | 0.062 | 0.03 | -0.038 | -0.04 | -0.07 | -0.0 |
| BSOutOfState | -0.039 | 0.14 | 0.07 | 0.09 | 0.032 | 0.19 | 0.015 | 0.074 | 0.091 | -0.12 | 0.064 | -0.053 | -0.12 | 0.14 | -0.093 | 0.097 | 0.14 | 0.12 | 0.1 |
| SaleUnderDuress | -0.015 | -0.19 | -0.17 | -0.13 | -0.031 | -0.14 | -0.082 | -0.1 | -0.13 | 0.14 | -0.003 | 0.16 | 0.2 | -0.16 | 0.099 | -0.086 | -0.2 | -0.11 | -0.2 |
| NoAssessmentRole | -0.082 | 0.079 | 0.029 | 0.05 | -0.024 | 0.079 | 0.11 | 0.037 | 0.11 | -0.14 | 0.068 | -0.033 | -0.084 | 0.11 | -0.035 | 0.009 | 0.15 | 0.095 | 0.2 |
| BSRelatedorCorporate | -0.017 | 0.1 | 0.19 | -0.02 | -0.009 | 0.031 | -0.025 | -0.014 | -0.034 | 0.04 | -0.006 | 0.008 | 0.071 | -0.025 | 0.018 | 0.009 | 0.044 | 0.025 | -0.0 |
| SaleGovt | -0.023 | 0.046 | 0.039 | -0.03 | -0.03 | -0.023 | -0.012 | -0.012 | -0.02 | 0.023 | -0.017 | -0.003 | 0.011 | -0.02 | 0.012 | -0.011 | -0.026 | 0.023 | 0.0 |
| isBuyerSellerRelated | -0.016 | 0.095 | 0.18 | -0.022 | 0.008 | 0.031 | -0.026 | -0.015 | -0.034 | 0.04 | -0.007 | 0.008 | 0.072 | -0.025 | 0.016 | 0.006 | 0.044 | 0.025 | 0.0 |

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

| | | | | | | | | | | | | | | | | | | | |
|---------------|--------|------|------|------|-------|------|-------|-------|-------|--------|------|-------|--------|------|-------|------|-------|------|------|
| diff | -0.063 | 0.75 | 0.89 | 0.12 | 0.087 | 0.14 | 0.007 | 0.075 | 0.045 | -0.014 | 0.03 | -0.03 | -0.026 | 0.14 | -0.04 | 0.07 | 0.072 | 0.14 | 0.05 |
| SaleDate | | | | | | | | | | | | | | | | | | | |
| SalePrice | | | | | | | | | | | | | | | | | | | |
| PrPerSqFt | | | | | | | | | | | | | | | | | | | |
| p_Cat | | | | | | | | | | | | | | | | | | | |
| RecordingDate | | | | | | | | | | | | | | | | | | | |
| CLASS | | | | | | | | | | | | | | | | | | | |
| STORIES | | | | | | | | | | | | | | | | | | | |
| ROOMS | | | | | | | | | | | | | | | | | | | |
| QUALITY | | | | | | | | | | | | | | | | | | | |
| WALLS | | | | | | | | | | | | | | | | | | | |
| ROOF | | | | | | | | | | | | | | | | | | | |
| HEAT | | | | | | | | | | | | | | | | | | | |
| COOL | | | | | | | | | | | | | | | | | | | |
| BATHFIXTUR | | | | | | | | | | | | | | | | | | | |
| PATIO | | | | | | | | | | | | | | | | | | | |
| PATIONUMBE | | | | | | | | | | | | | | | | | | | |
| CONDITION | | | | | | | | | | | | | | | | | | | |
| SQFT | | | | | | | | | | | | | | | | | | | |
| YEAR | | | | | | | | | | | | | | | | | | | |

```

# Normalize all data. That way my weights will be my importances

normalized_df=(data-data.min())/(data.max()-data.min())

print((normalized_df['SalePrice'] - normalized_df['TaxEval']).mean())

# train test split

data_copy = normalized_df.copy()
trainData = data_copy.sample(frac=0.8, random_state=0)
testData = data_copy.drop(trainData.index)

X_train = trainData[features].to_numpy().astype(float)
y_train = (trainData['diff']).to_numpy().reshape(len(trainData),1).astype(float)

X_test = testData[features].to_numpy().astype(float)
y_test = (testData['diff']).to_numpy().reshape(len(testData),1).astype(float)

[-0.04555994695631326

class OrdinaryLeastSquares():
    def __init__(self):
        self.coefficients = []

    def _reshape_x(self,X):
        return X.reshape(-1, 1)

    def _concatenate_ones(self, X):
        ones = np.ones(shape = X.shape[0]).reshape(-1,1)
        return np.concatenate((ones, X), 1)

    def fit(self, X, y):

        if len(X.shape) == 1: X = self._reshape_x(X)

        X = self._concatenate_ones(X)

        self.coefficients = np.linalg.inv(X.transpose().dot(X)).dot(X.transpose()).dot(y)

        w0 = self.coefficients[0]
        other_gammas = self.coefficients[1:]
        prediction = w0

        for xi, wi in zip(y, other_gammas):
            prediction = prediction + (wi * xi)

        return prediction

```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

```

def predict(self, y):

    y_preds = []

    for row in y:
        y_preds.append(self.predict_one(row))

    return y_preds

def get_importance(self):

    return self.coefficients

def rmse(y, y_hat):

    #combined rmse value
    rss=((y-y_hat)**2).sum()
    mse=np.mean((y-y_hat)**2)
    rmse = np.sqrt(mse)

    return rmse

def getImportanceTable(ols, features):

    weights_array = ols.get_importance().astype(float).tolist()
    weights_array = [item for sublist in weights_array for item in sublist]
    weights_array = weights_array[1:]
    weights_sum = sum(list(map(abs, weights_array)))

    weights_array[:] = [x / weights_sum for x in weights_array]

    importances = list(zip(features, weights_array))
    importances = sorted(importances, key=lambda x : abs(x[1]), reverse=True)

    num = np.array(importances)
    reshaped = num.reshape(len(features),2)
    print(pd.DataFrame(reshaped, columns=['Feature', 'Importance']))

ols = OrdinaryLeastSquares()

```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



```

y_hat = [item for sublist in y_hat for item in sublist]

X_plot = y_test

plt.figure()
plt.scatter(X_plot,y_test)
plt.scatter(X_plot,y_hat, color='#FF0000')

```

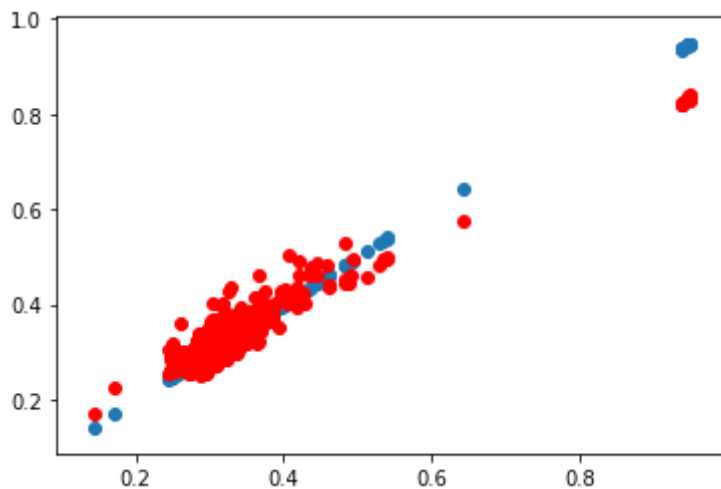
```
from sklearn.metrics import r2_score
```

```
print(r2_score(y_test, y_hat))
```

```
print(rmse(y_test, y_hat))
```

```
print(np.mean(y_test))
```

```
↳ 0.8821828236467271  
0.036954835707755804  
0.3094048175858326
```



```
getImportanceTable(ols, features)
```

```
↳
```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



| | Feature | Importance |
|----|----------------------|-------------------------|
| 0 | SalePrice | 0.5238586940004392 |
| 1 | SQFT | -0.1610867276052656 |
| 2 | GISACRES | -0.05052674619527425 |
| 3 | CLASS | -0.04235772289352165 |
| 4 | ROOMS | 0.034389415493254964 |
| 5 | BATHFIXTUR | -0.017500576640732925 |
| 6 | PATIONUMBE | -0.015694745631574816 |
| 7 | LON | -0.015447626332676292 |
| 8 | LAT | -0.013820349792109683 |
| 9 | isPartialInterest | -0.013443622589129726 |
| 10 | YEAR | 0.010601727068947926 |
| 11 | STORIES | 0.010408606279021399 |
| 12 | QUALITY | -0.010342667567723136 |
| 13 | isBuyerSellerRelated | 0.010105126307962632 |
| 14 | isPersonalProperty | 0.007007546973883275 |
| 15 | p_Cat | -0.006075370807905544 |
| 16 | POOLAREA | -0.005340433535388927 |
| 17 | HEAT | -0.005206694505206239 |
| 18 | GARAGECAPA | -0.0050401740872237895 |
| 19 | NonPrimary | -0.004100824721960035 |
| 20 | SaleGovt | 0.0033379619939088953 |
| 21 | LASTACTION | -0.00316410183909218 |
| 22 | RecordingDate | 0.003102959546016357 |
| 23 | GoodSale | 0.003080464486048154 |
| 24 | Rental | -0.0030583789018807195 |
| 25 | PrimaryRes | -0.002894368182856445 |
| 26 | PATIO | -0.0028219387429721738 |
| 27 | BSOutOfState | 0.002706137857961582 |
| 28 | WALLS | 0.002631767818019001 |
| 29 | BSRelatedorCorporate | -0.0024896089742632104 |
| 30 | COOL | 0.0018649715674562005 |
| 31 | SaleDate | -0.0014912044390243684 |
| 32 | CONDITION | 0.0013012673921590143 |
| 33 | SaleUnderDuress | 0.0010363281544455455 |
| 34 | GARAGE | -0.0006315033091777621 |
| 35 | JointTenancyDeed | 0.0005986506137744087 |
| 36 | ROOF | 0.0004005220253590113 |
| 37 | FinancingCash | -0.00034115991135851166 |
| 38 | NoAssessmentRole | -0.0003024382514219819 |
| 39 | ZIP | 0.00021900536611992087 |
| 40 | isSolar | -0.0001311657845844674 |
| 41 | WarrantyDeed | 3.86958128977186e-05 |

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



```
features = list(data.columns.values.tolist())
```

```
# Weight coefficients say these features are very unimportant
```

```
features.remove('NoAssessmentRole')
```

```
features.remove('WarrantyDeed')
```

```
features.remove('isSolar')
```

```
features.remove('ZIP')
```

```
features.remove('FinancingCash')
```



```
features.remove('JointTenancyDeed')
features.remove('GARAGE')
features.remove('CONDITION')
features.remove('ROOF')
features.remove('SaleUnderDuress')
features.remove('BSOutOfState')
features.remove('COOL')
features.remove('PATIO')
features.remove('SaleGovt')
features.remove('PrimaryRes')
features.remove('RecordingDate')
features.remove('GARAGECAPA')

# Remove very correlated features based on correlation matrix
features.remove('GoodSale')
features.remove('Rental')
features.remove('NonPrimary')
features.remove('PATIONUMBE')
features.remove('isBuyerSellerRelated')
features.remove('WALLS')

# Stays in because it hurts my model to remove
#features.remove('CLASS')

# Now remove features we can't use in the model

features.remove('diff')
features.remove('PrPerSqFt')
features.remove('TaxEval')

# Print correlation matrix

corrMat = data[features].corr()

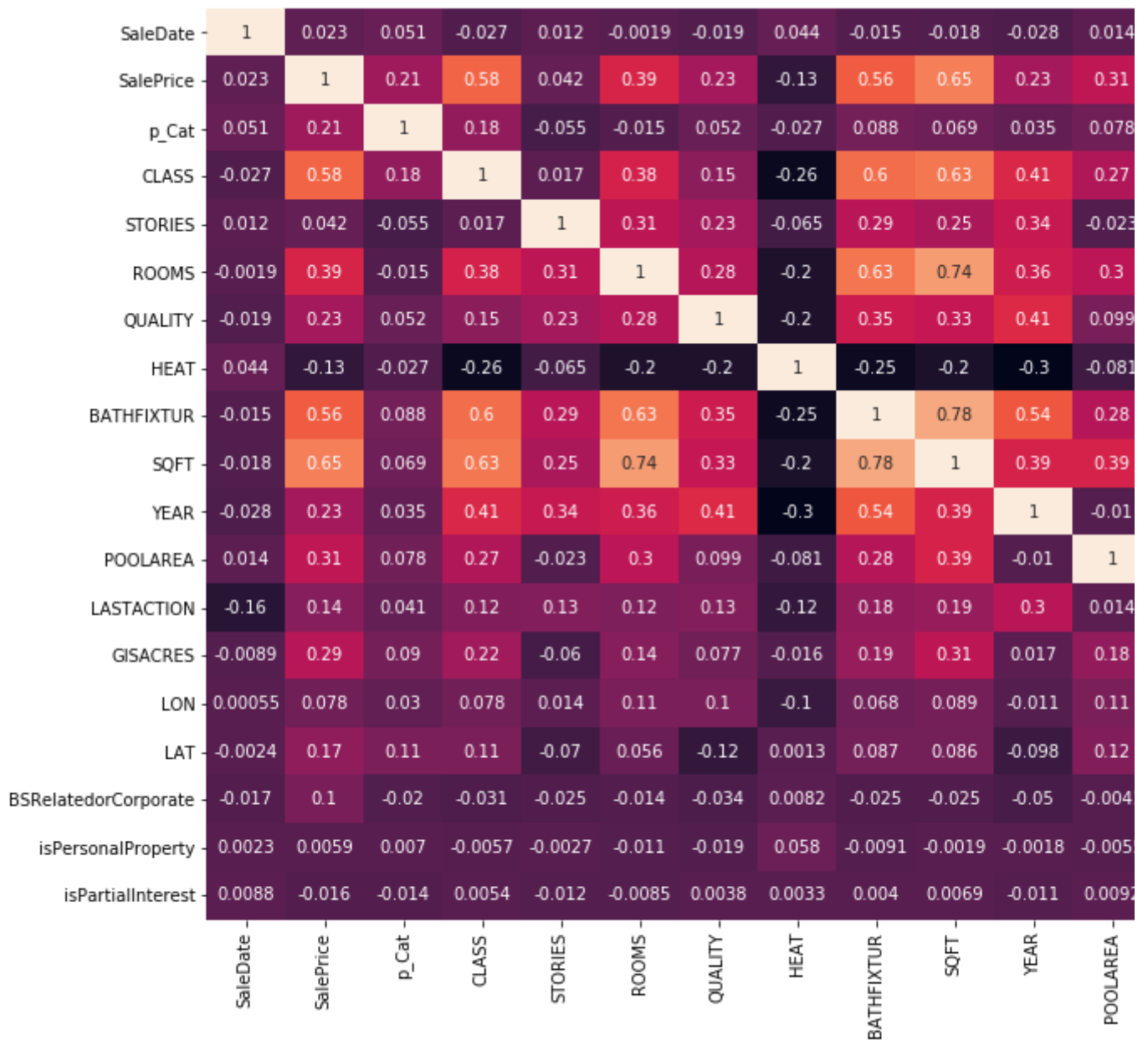
plt.figure(figsize=(20,10))
sn.heatmap(corrMat, annot=True)
```



Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



<matplotlib.axes._subplots.AxesSubplot at 0x7f840ec55080>



```
X_train = trainData[features].to_numpy().astype(float)
```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

```
X_test = testData[features].to_numpy().astype(float)
```

```
y_test = (testData['diff']).to_numpy().reshape(len(testData),1).astype(float)
```

```
ols = OrdinaryLeastSquares()
```

```
ols.fit(X_train,y_train)
```

```
y_hat = ols.predict(X_test)
```

```
y_hat = [item for sublist in y_hat for item in sublist]
```

```

X_plot = (testData['SalePrice']).to_numpy().reshape(len(testData),1).astype(float)

plt.figure()
plt.scatter(X_plot,y_test)
plt.scatter(X_plot,y_hat, color='#FF0000')

from sklearn.metrics import r2_score

print(r2_score(y_test, y_hat))
print(rmse(y_test, y_hat))

getImportanceTable(ols, features)

```

```

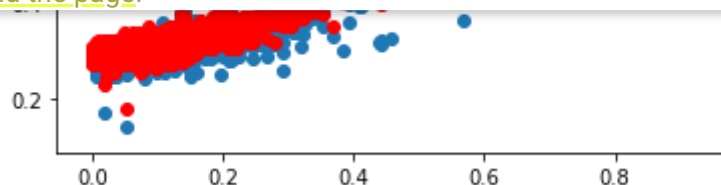
0.8777362989085593
0.03690838036730064

```

| | Feature | Importance |
|----|----------------------|------------------------|
| 0 | SalePrice | 0.5542740419336928 |
| 1 | SQFT | -0.1765944457055261 |
| 2 | GISACRES | -0.05854365244969522 |
| 3 | CLASS | -0.04604063133012461 |
| 4 | ROOMS | 0.038649221036096894 |
| 5 | BATHFIXTUR | -0.01979175525196903 |
| 6 | LON | -0.014576916416537742 |
| 7 | LAT | -0.013937638105724303 |
| 8 | isPartialInterest | -0.012480076120169279 |
| 9 | STORIES | 0.010666380891096677 |
| 10 | QUALITY | -0.010618087609952451 |
| 11 | YEAR | 0.00915728019565712 |
| 12 | HEAT | -0.007591248008502165 |
| 13 | p_Cat | -0.006445846759644615 |
| 14 | BSRelatedorCorporate | 0.0053432230558870305 |
| 15 | isPersonalProperty | 0.005334874975627119 |
| 16 | POOLAREA | -0.005137657654759445 |
| 17 | LASTACTION | -0.0038553440272885756 |
| 18 | SaleDate | 0.0009616784720488255 |



Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



```
print(np.median(y_test))
```

```
0.3073582222790861
```

```
scale_factor = int((max(data['diff']) - min(data['diff'])))

y_test = y_test.reshape(len(y_test),) * scale_factor
y_hat = [x * scale_factor for x in y_hat]
residuals = y_test - y_hat

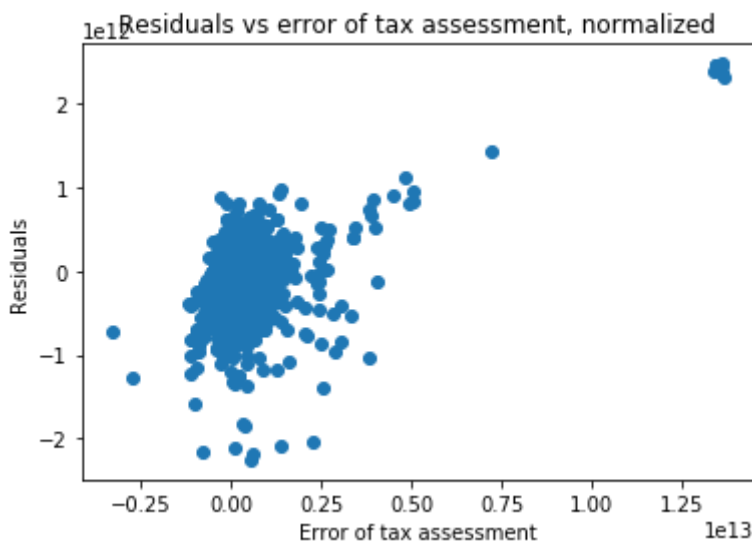
#print(residuals.shape)
#print(y_test.shape)
#print(scale_factor)
plt.figure()

plt.xlabel('Error of tax assessment')
plt.ylabel('Residuals')
plt.title('Residuals vs error of tax assessment, normalized')

plt.scatter(y_test, residuals)

#print(" RMSE of house value is " + str(scale_factor * 0.03691012168704651))
```

```
<matplotlib.collections.PathCollection at 0x7f840effec18>
```



```
# Normalize all data. What you see weights will be my importances
```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

```
# train test split

data_copy = data.copy()
trainData = data_copy.sample(frac=0.8, random_state=0)
testData = data_copy.drop(trainData.index)

X_train = trainData[features].to_numpy().astype(float)
y_train = (trainData['diff']).to_numpy().reshape(len(trainData),1).astype(float)
```

```
X_test = testData[features].to_numpy().astype(float)
y_test = (testData['diff']).to_numpy().reshape(len(testData),1).astype(float)

ols = OrdinaryLeastSquares()

ols.fit(X_train,y_train)
y_hat = ols.predict(X_test)

y_hat = [item for sublist in y_hat for item in sublist]

X_plot = (testData['SalePrice']).to_numpy().reshape(len(testData),1).astype(float)

plt.figure()

plt.xlabel('House Sales Price')
plt.ylabel('Predicted and Actual Assessment Error')
plt.title('Predicted (red) vs Actual (Blue) Assessment Error')

plt.scatter(X_plot,y_test)
plt.scatter(X_plot,y_hat, color='#FF0000')

from sklearn.metrics import r2_score

print(r2_score(y_test, y_hat))
print(rmse(y_test, y_hat))

getImportanceTable(ols, features)
```



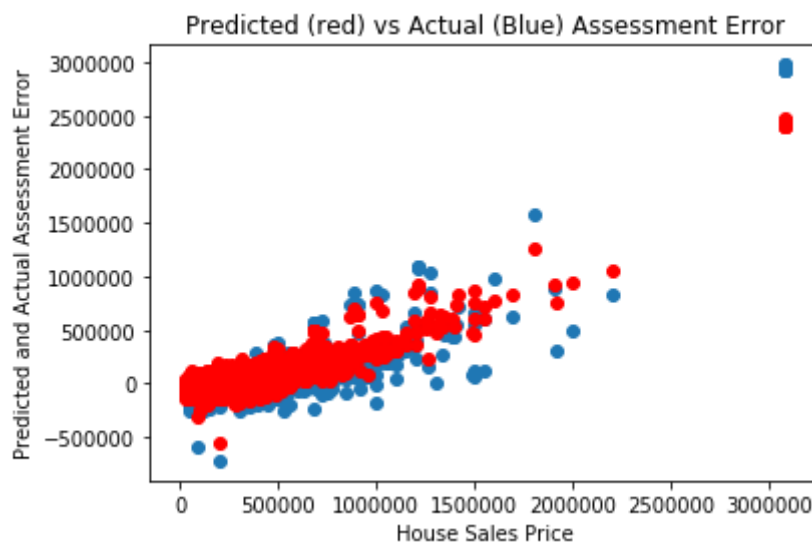
Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



0.8777362989087802

169237.7188891875

| | Feature | Importance |
|----|----------------------|-------------------------|
| 0 | LAT | -0.2538650895008694 |
| 1 | isPartialInterest | -0.17812355698527096 |
| 2 | CLASS | -0.13142421472402277 |
| 3 | LON | -0.08576271178970861 |
| 4 | BSRelatedorCorporate | 0.07626186629749096 |
| 5 | isPersonalProperty | 0.076142717205788 |
| 6 | STORIES | 0.07611867453217096 |
| 7 | QUALITY | -0.05051602558156512 |
| 8 | ROOMS | 0.025073917307846406 |
| 9 | GISACRES | -0.02197862270553631 |
| 10 | HEAT | -0.012038567038466199 |
| 11 | BATHFIXTUR | -0.01046223985680548 |
| 12 | YEAR | 0.0010372897411348257 |
| 13 | p_Cat | -0.0007863180402079631 |
| 14 | SQFT | -0.00027271887741514334 |
| 15 | POOLAREA | -6.666173327413906e-05 |
| 16 | SaleDate | 6.630765533463487e-05 |
| 17 | SalePrice | 2.383125060784364e-06 |
| 18 | LASTACTION | -1.1730203139332165e-07 |



```
y_test = y_test.reshape(len(y_test),)
residuals = y_test - y_hat
```

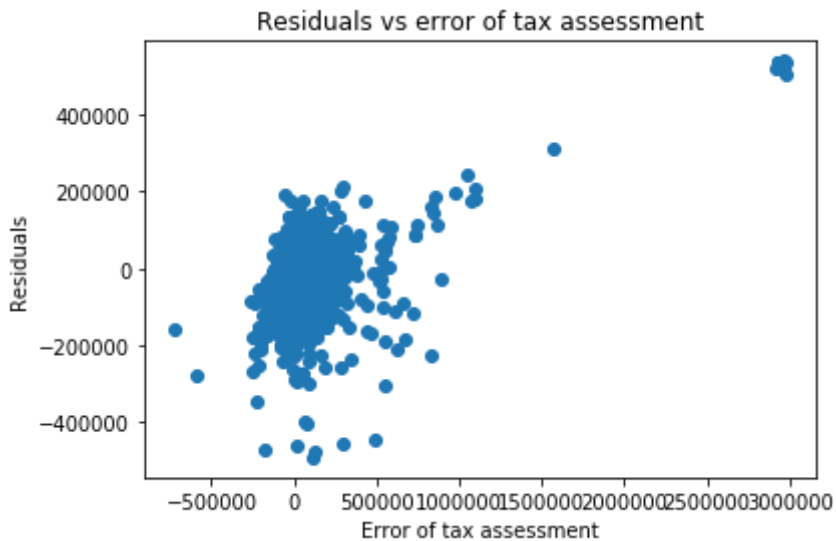
Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

```
#print(scale_factor)
plt.figure()
```

```
plt.xlabel('Error of tax assessment')
plt.ylabel('Residuals')
plt.title('Residuals vs error of tax assessment')
```

```
plt.scatter((y_test), (residuals))
```

↳ <matplotlib.collections.PathCollection at 0x7f840b43ba90>

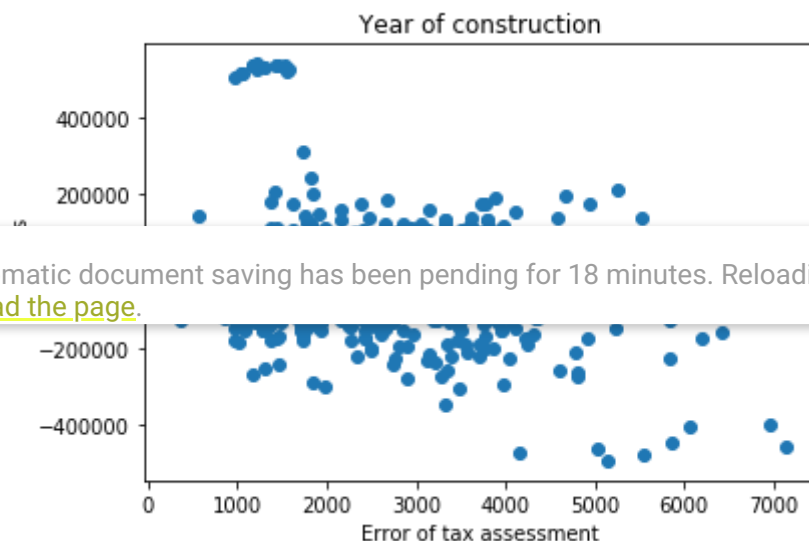


```
y_test = y_test.reshape(len(y_test),)
residuals = y_test - y_hat
```

```
#print(residuals.shape)
#print(y_test.shape)
#print(scale_factor)
plt.figure()
```

```
plt.xlabel('Error of tax assessment')
plt.ylabel('Residuals')
plt.title('Year of construction')
X_axis = X_test[:,9]
y_axis = residuals
plt.scatter(X_axis, y_axis)
```

↳ <matplotlib.collections.PathCollection at 0x7f840a3a5518>



Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



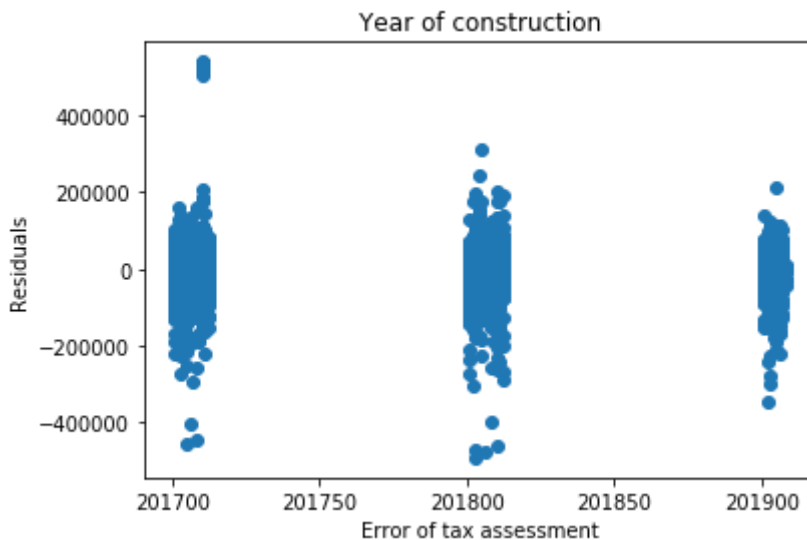
```
y_test = y_test.reshape(len(y_test),)
```

```
residuals = y_test - y_nat
```

```
#print(residuals.shape)
#print(y_test.shape)
#print(scale_factor)
plt.figure()
```

```
plt.xlabel('Error of tax assessment')
plt.ylabel('Residuals')
plt.title('Year of construction')
X_axis = X_test[:,0]
y_axis = residuals
plt.scatter(X_axis, y_axis)
```

↳ <matplotlib.collections.PathCollection at 0x7f840f0c0668>



```
plt.figure()

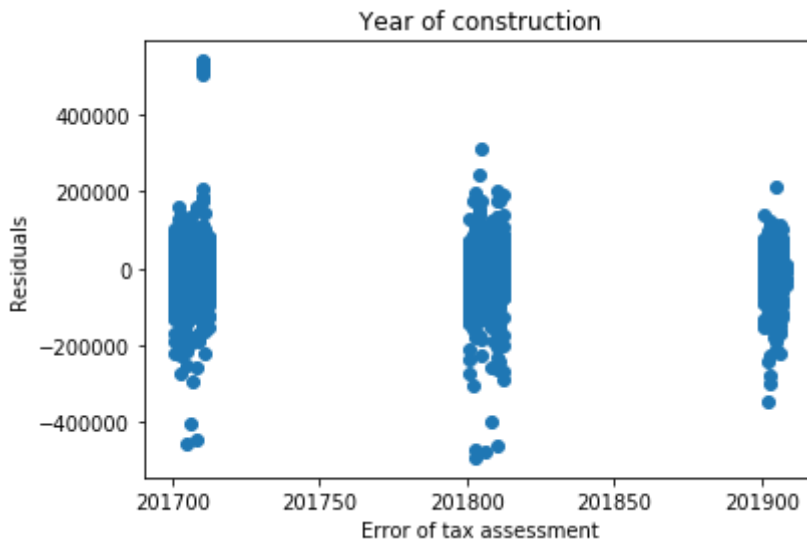
plt.xlabel('Error of tax assessment')
plt.ylabel('Residuals')
plt.title('Year of construction')
X_axis = X_test[:,0]
y_axis = residuals
plt.scatter(X_axis, y_axis)
```

↳

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



<matplotlib.collections.PathCollection at 0x7f840f46edd8>



```
print("Mean diff is:" + str(np.median(y_test)))
print("Mean house is is:" + str(np.median(X_test[:,1])))

print(np.median(y_test) / np.median(X_test[:,1]))
```

```
☞ Mean diff is:152909061510.66672
   Mean house is is:210000.0
   728138.388146032
```

```
print((data['TaxEval']).median())
print((data['SalePrice']).median())
```

```
☞ 176805.66666666666
   210000.0
```

Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)



Automatic document saving has been pending for 18 minutes. Reloading may fix the problem. [Save and reload the page.](#)

