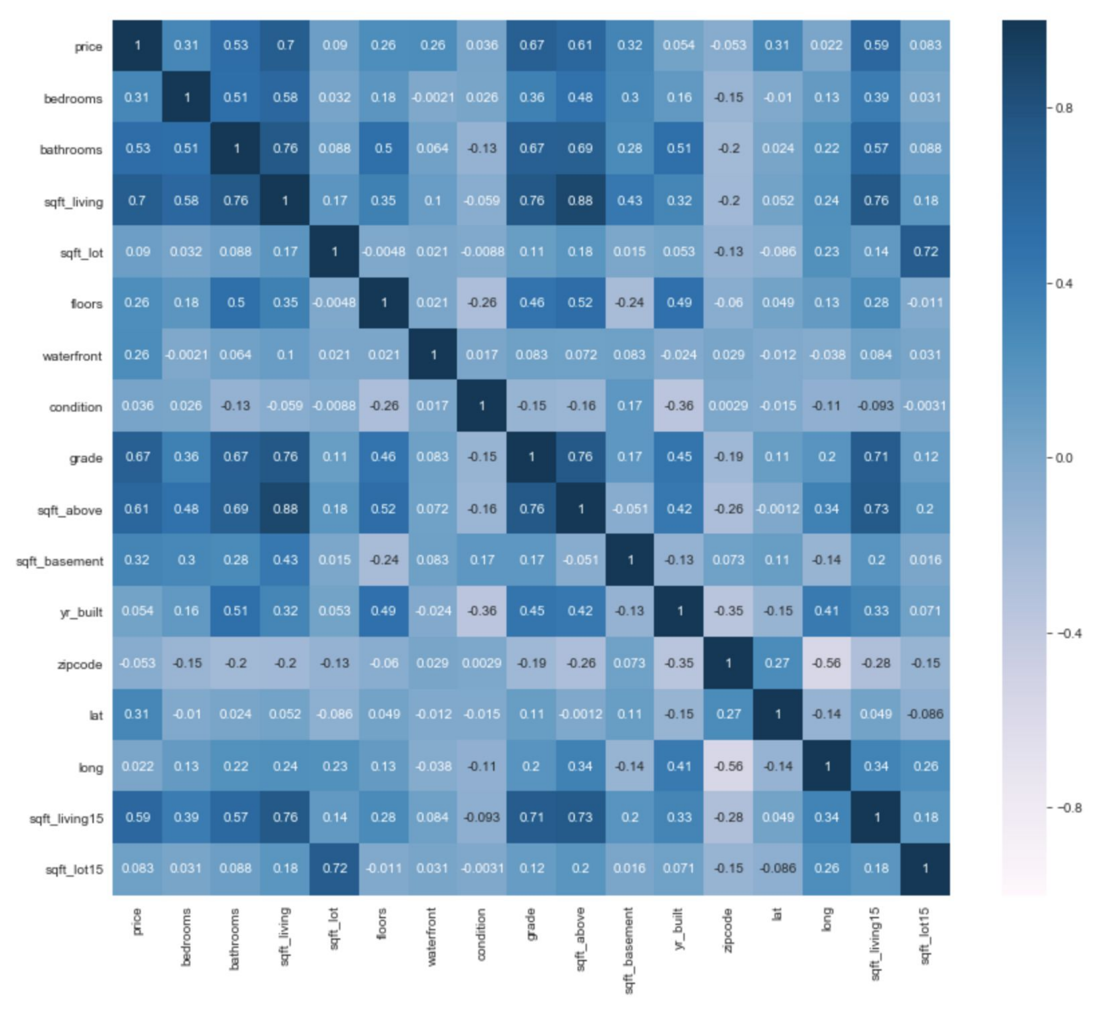
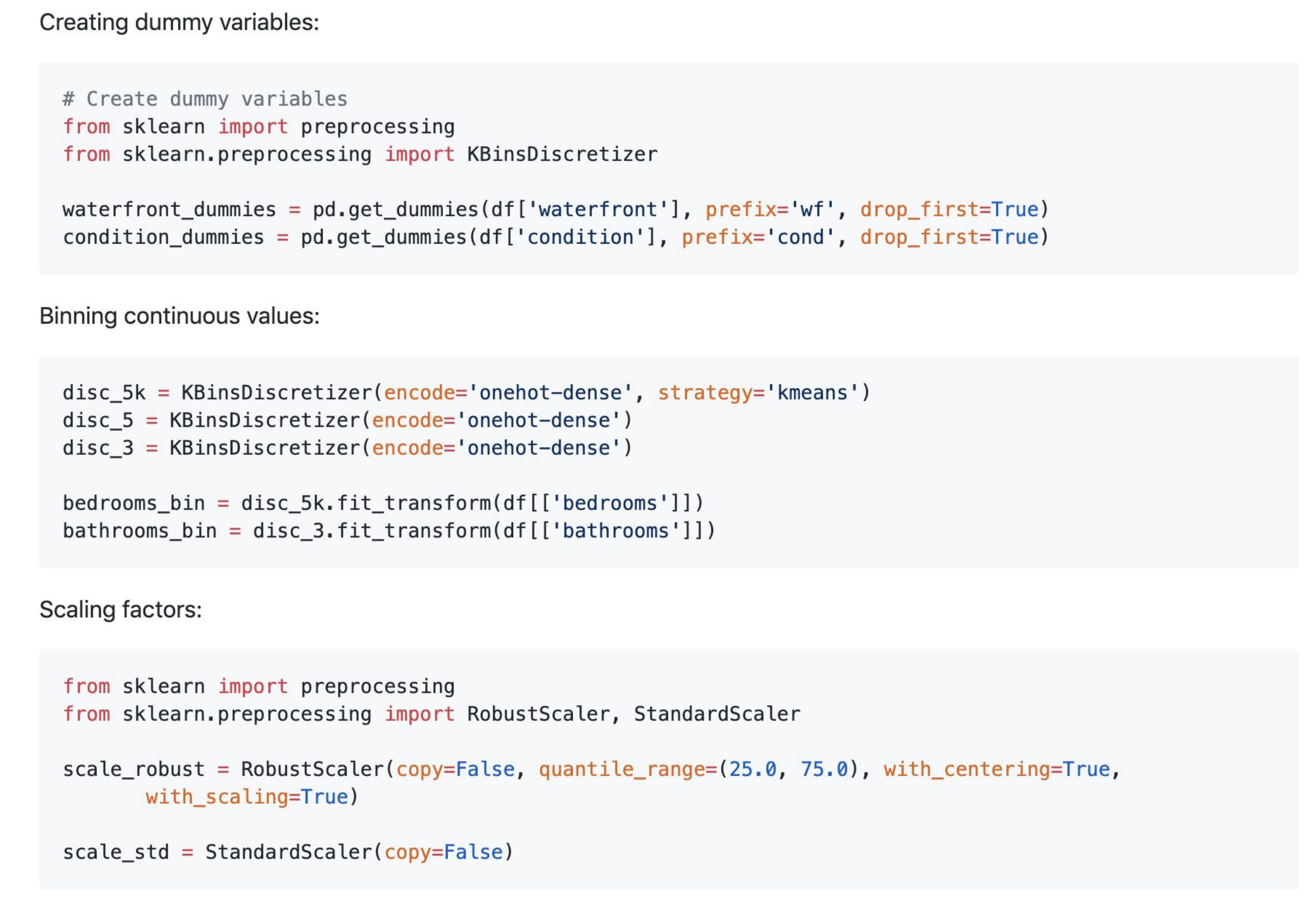
Predicting Home Values with Linear Regression

Contents of the Jupyter Notebook:

1. Methodology
2. Exploring the Data
3. Splitting, Selecting, and Scaling Data
4. Linear Regression
5. Takeaways



	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	condition	grade	sqft_above	sqft_basement	yr_built	zipcode	lat	long	sqft_living15	sqft_lot15
price	1	0.31	0.53	0.7	0.09	0.26	0.26	0.036	0.67	0.61	0.32	0.054	-0.053	0.31	0.022	0.59	0.083
bedrooms	0.31	1	0.51	0.58	0.032	0.18	-0.0021	0.026	0.36	0.48	0.3	0.16	-0.15	-0.01	0.13	0.39	0.031
bathrooms	0.53	0.51	1	0.76	0.088	0.5	0.064	-0.13	0.67	0.69	0.28	0.51	-0.2	0.024	0.22	0.57	0.088
sqft_living	0.7	0.58	0.76	1	0.17	0.35	0.1	-0.059	0.76	0.88	0.43	0.32	-0.2	0.052	0.24	0.76	0.18
sqft_lot	0.09	0.032	0.088	0.17	1	-0.0048	0.021	-0.0088	0.11	0.18	0.015	0.053	-0.13	-0.086	0.23	0.14	0.72
floors	0.26	0.18	0.5	0.35	-0.0048	1	0.021	-0.26	0.46	0.52	-0.24	0.49	-0.06	0.049	0.13	0.28	-0.011
waterfront	0.26	-0.0021	0.064	0.1	0.021	0.021	1	0.017	0.083	0.072	0.083	-0.024	0.029	-0.012	-0.038	0.084	0.031
condition	0.036	0.026	-0.13	-0.059	-0.0088	-0.26	0.017	1	-0.15	-0.16	0.17	-0.36	0.0029	-0.015	-0.11	-0.093	-0.0031
grade	0.67	0.36	0.67	0.76	0.11	0.46	0.083	-0.15	1	0.76	0.17	0.45	-0.19	0.11	0.2	0.71	0.12
sqft_above	0.61	0.48	0.69	0.88	0.18	0.52	0.072	-0.16	0.76	1	-0.051	0.42	-0.26	-0.0012	0.34	0.73	0.2
sqft_basement	0.32	0.3	0.28	0.43	0.015	-0.24	0.083	0.17	0.17	-0.051	1	-0.13	0.073	0.11	-0.14	0.2	0.016
yr_built	0.054	0.16	0.51	0.32	0.053	0.49	-0.024	-0.36	0.45	0.42	-0.13	1	-0.35	-0.15	0.41	0.33	0.071
zipcode	-0.053	-0.15	-0.2	-0.2	-0.13	-0.06	0.029	0.0029	-0.19	-0.26	0.073	-0.35	1	0.27	-0.56	-0.28	-0.15
lat	0.31	-0.01	0.024	0.052	-0.086	0.049	-0.012	-0.015	0.11	-0.0012	0.11	-0.15	0.27	1	-0.14	0.049	-0.086
long	0.022	0.13	0.22	0.24	0.23	0.13	-0.038	-0.11	0.2	0.34	-0.14	0.41	-0.56	-0.14	1	0.34	0.26
sqft_living15	0.59	0.39	0.57	0.76	0.14	0.28	0.084	-0.093	0.71	0.73	0.2	0.33	-0.28	0.049	0.34	1	0.18
sqft_lot15	0.083	0.031	0.088	0.18	0.72	-0.011	0.031	-0.0031	0.12	0.2	0.016	0.071	-0.15	-0.086	0.26	0.18	1



Creating dummy variables:

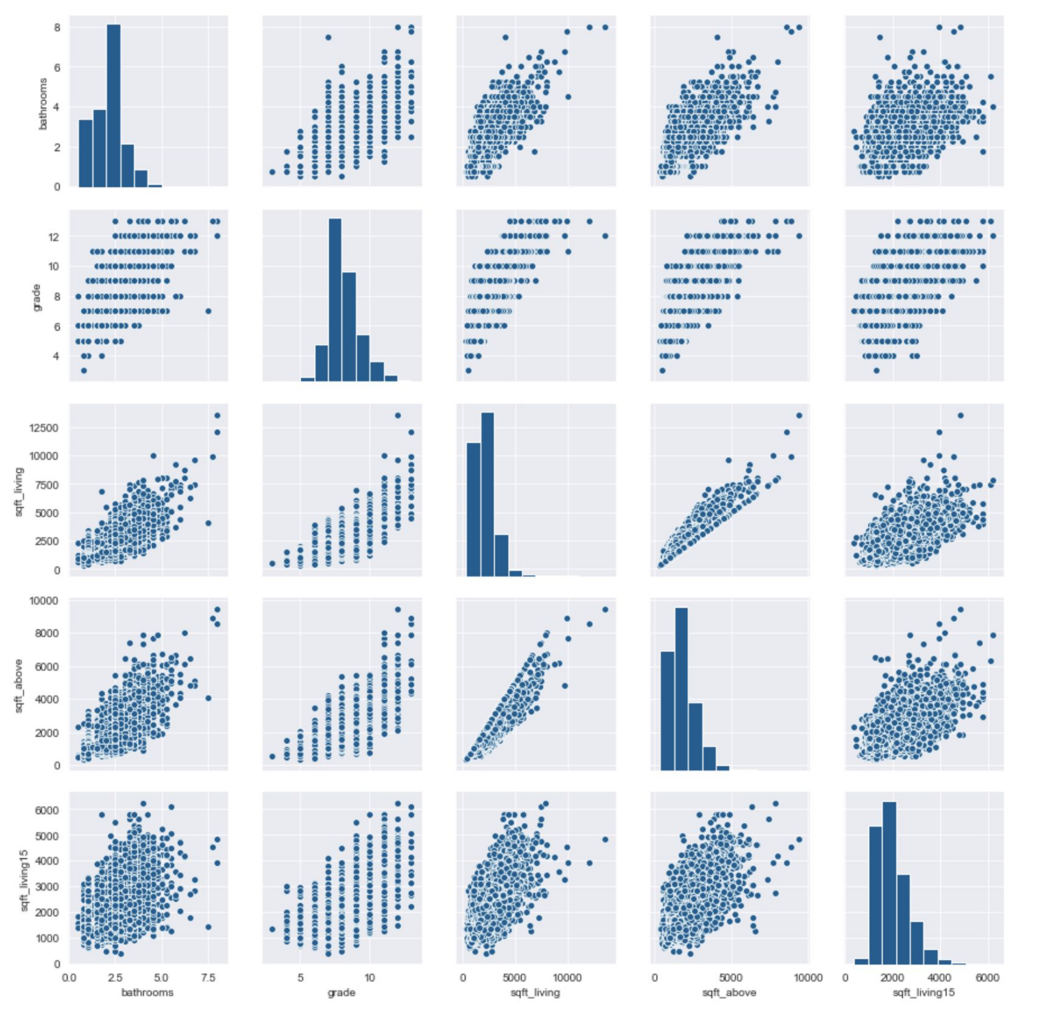
```
# Create dummy variables
from sklearn import preprocessing
from sklearn.preprocessing import KBinsDiscretizer

waterfront_dummies = pd.get_dummies(df['waterfront'], prefix='wf', drop_first=True)
condition_dummies = pd.get_dummies(df['condition'], prefix='cond', drop_first=True)
```

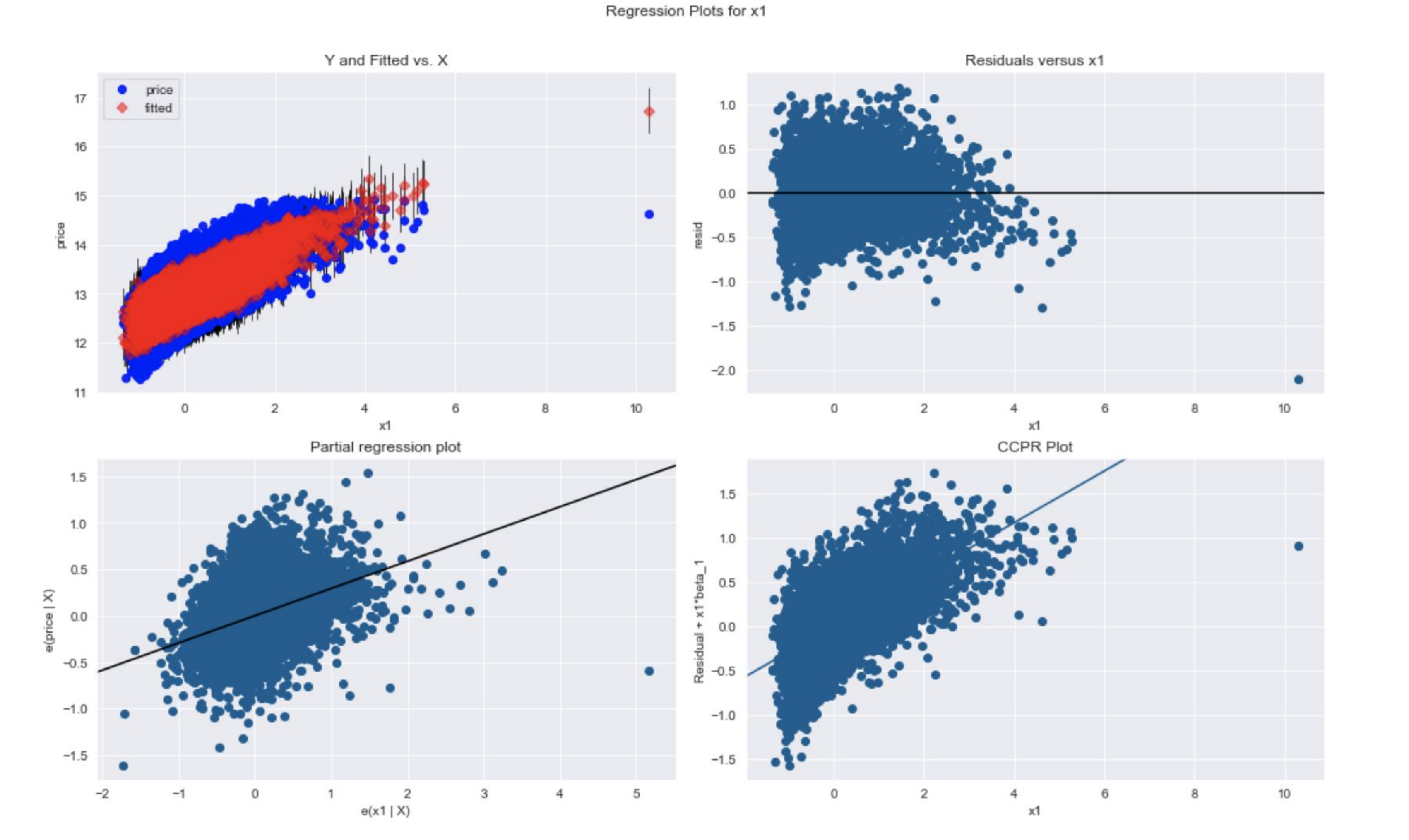
Binning continuous values:

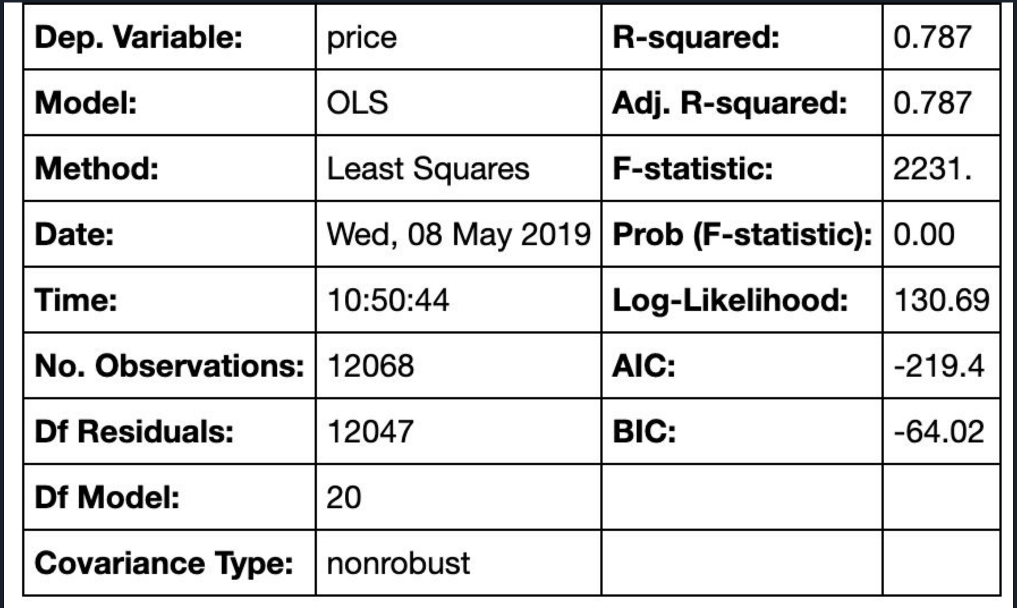
```
disc_5k = KBinsDiscretizer(encode='onehot-dense', strategy='kmeans')
disc_5 = KBinsDiscretizer(encode='onehot-dense')
disc_3 = KBinsDiscretizer(encode='onehot-dense')

bedrooms_bin = disc_5k.fit_transform(df[['bedrooms']])
bathrooms_bin = disc_3.fit_transform(df[['bathrooms']])
```

Scaling factors:

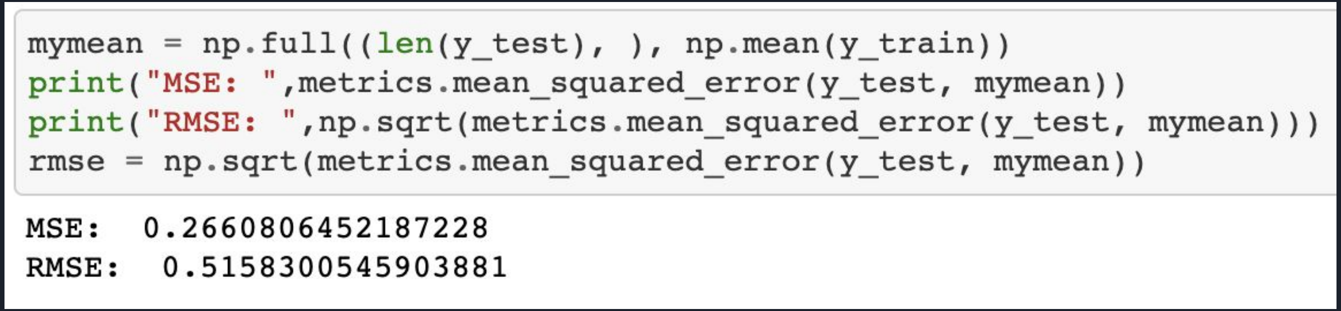
```
from sklearn import preprocessing
from sklearn.preprocessing import RobustScaler, StandardScaler

scale_robust = RobustScaler(copy=False, quantile_range=(25.0, 75.0), with_centering=True,
       with_scaling=True)

scale_std = StandardScaler(copy=False)
```


bathrooms
grade
sqft_living
sqft_above
sqft_living15
0
2
4
6
8
10
12
2500
5000
7500
10000
12500
2000
4000
6000
8000
1000
3000
0.0
2.5
5.0
7.5


Regression Plots for x1
Y and Fitted vs. X
price
fitted
17
16
15
14
13
12
11
0
2
4
6
8
10
x1
Residuals versus x1
resid
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
2
4
6
8
10
x1
Partial regression plot
e(price | X)
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
4
5
e(x1 | X)
CCPR Plot
Residual + x1*beta_1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
2
4
6
8
10
x1

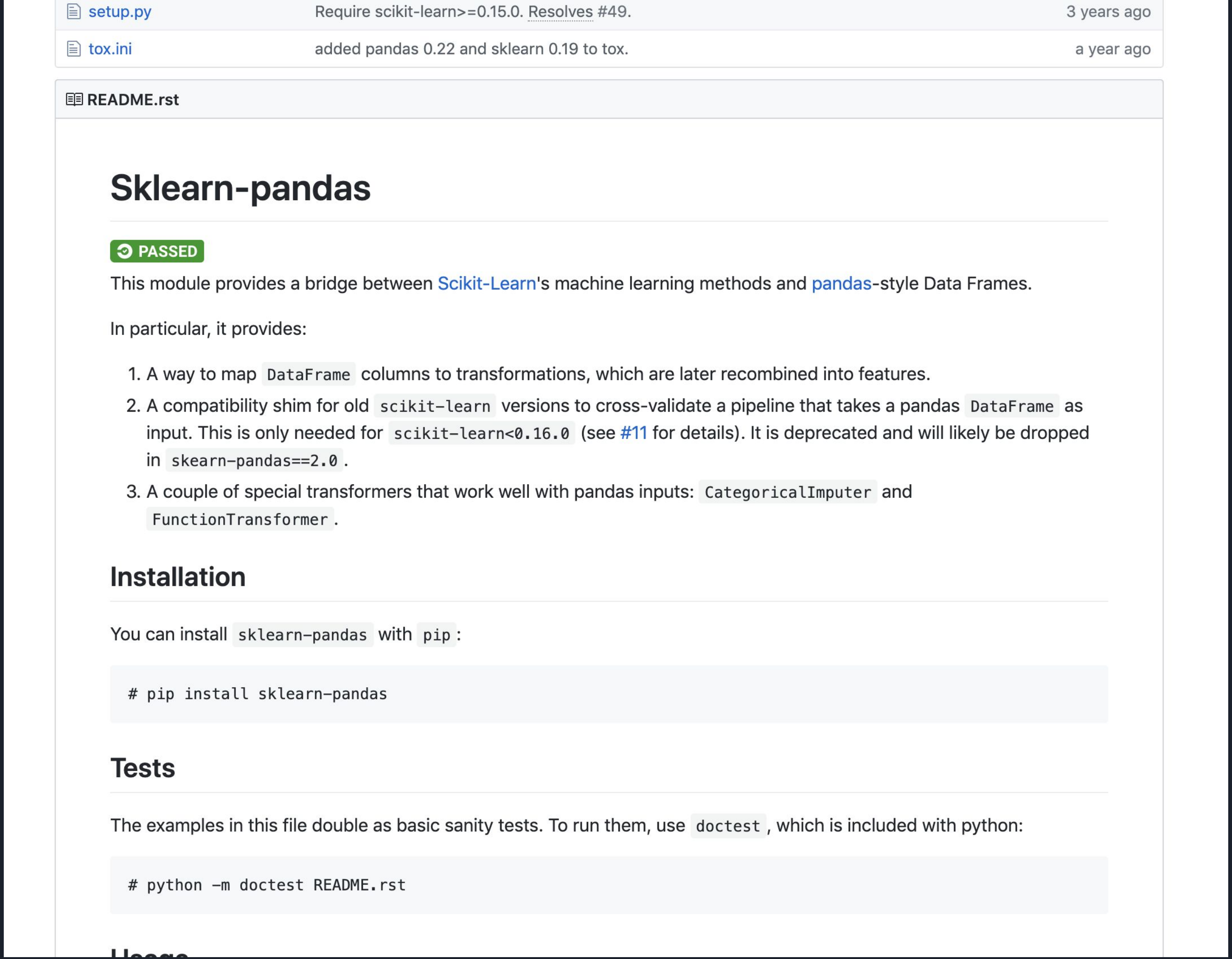


Dep. Variable:	price	**R-squared:**	0.787
Model:	OLS	**Adj. R-squared:**	0.787
Method:	Least Squares	**F-statistic:**	2231.
Date:	Wed, 08 May 2019	**Prob (F-statistic):**	0.00
Time:	10:50:44	**Log-Likelihood:**	130.69
No. Observations:	12068	**AIC:**	-219.4
Df Residuals:	12047	**BIC:**	-64.02
Df Model:	20		
Covariance Type:	nonrobust		



```
mymean = np.full((len(y_test), ), np.mean(y_train))
print("MSE: ",metrics.mean_squared_error(y_test, mymean))
print("RMSE: ",np.sqrt(metrics.mean_squared_error(y_test, mymean)))
rmse = np.sqrt(metrics.mean_squared_error(y_test, mymean))
```

```
MSE:  0.2660806452187228
RMSE:  0.5158300545903881
```



setup.py	Require scikit-learn>=0.15.0. Resolves #49.	3 years ago
tox.ini	added pandas 0.22 and sklearn 0.19 to tox.	a year ago

README.rst

Sklearn-pandas

PASSED

This module provides a bridge between Scikit-Learn's machine learning methods and pandas-style Data Frames.

In particular, it provides:

1. A way to map `DataFrame` columns to transformations, which are later recombined into features.
2. A compatibility shim for old `scikit-learn` versions to cross-validate a pipeline that takes a pandas `DataFrame` as input. This is only needed for `scikit-learn<0.16.0` (see #11 for details). It is deprecated and will likely be dropped in `skearn-pandas==2.0`.
3. A couple of special transformers that work well with pandas inputs: `CategoricalImputer` and `FunctionTransformer`.

Installation

You can install `sklearn-pandas` with `pip`:

```
# pip install sklearn-pandas
```

Tests

The examples in this file double as basic sanity tests. To run them, use `doctest`, which is included with python:

```
# python -m doctest README.rst
```

Usage

Takeaways

- + My models were able to predict the price of a house with an adjusted R-squared of 0.79 and a MSE of 0.27.
- In this dataset, sqft_living and grade are the variables most highly correlated with price.
- R-squared and MSE/RMSE can diverge sharply, and multiple metrics of accuracy must be taken into account.
- The choice of which variables to keep and which to eliminate for multicollinearity has significant impacts on the accuracy of the final model.
- Data exploration and cleaning take up more time than running the actual regressions, but are crucial precursors to manipulating and processing the information.
- Pandas and scikitlearn contain many powerful tools with extensive documentation to aid in statistical analysis of data, but these libraries present a surprising number of incompatibilities that must be resolved via ad-hoc methods (or discovery of additional libraries like sklearn-pandas).