Iowa House Prices

●●●

By Nobel Gautam

Data

- Given features of a house listing, predict the sale price.
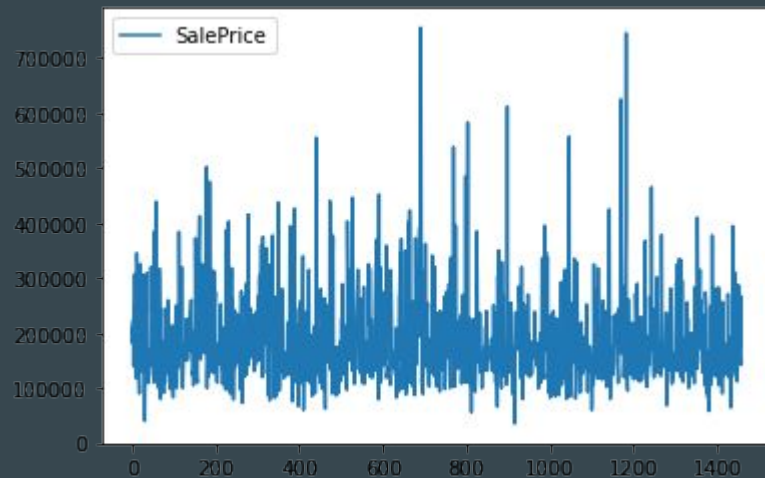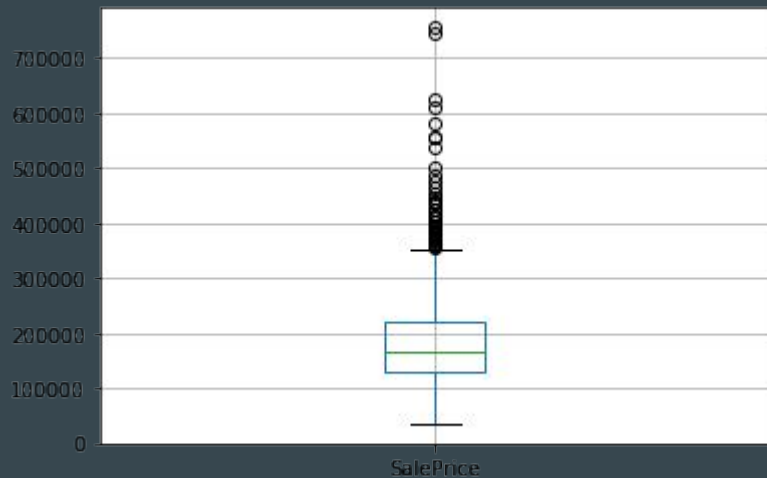
Data

- 80 columns. However, a lot of them are low-info.
- Lots of categorical columns.
- For simplicity, only use numerical columns (and transform categorical to numerical where applicable).

Example subsection of data.describe()

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	BsmtFinSF1	BsmtUnfSF	TotalBsmtSF	1stFlrSF	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath
count	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000	1127.000000
mean	728.352263	56.113576	70.683230	10120.070098	6.220053	5.559006	1972.440106	1985.788820	439.729370	594.275067	1078.356699	1174.341615	1531.812777	0.415262	0.055013	1.582076	0.383319
std	420.432826	41.746693	24.261598	8110.632788	1.383558	1.066120	30.978187	21.020073	469.047346	450.144102	445.692845	386.305987	522.979768	0.512419	0.231968	0.549631	0.499028
min	1.000000	20.000000	21.000000	1300.000000	2.000000	2.000000	1880.000000	1950.000000	0.000000	0.000000	0.000000	438.000000	438.000000	0.000000	0.000000	0.000000	0.000000
25%	367.000000	20.000000	60.000000	7593.000000	5.000000	5.000000	1953.000000	1967.000000	0.000000	250.500000	804.000000	894.000000	1158.000000	0.000000	0.000000	1.000000	0.000000
50%	728.000000	50.000000	70.000000	9430.000000	6.000000	5.000000	1975.000000	1995.000000	375.000000	506.000000	1008.000000	1098.000000	1479.000000	0.000000	0.000000	2.000000	0.000000
75%	1091.500000	70.000000	80.000000	11361.500000	7.000000	6.000000	2003.000000	2005.000000	705.500000	840.000000	1330.500000	1411.500000	1776.000000	1.000000	0.000000	2.000000	1.000000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	5644.000000	2336.000000	6110.000000	4692.000000	5642.000000	2.000000	2.000000	3.000000	2.000000

Data

- House sale price average ~185k, outliers exceeding 700k

Logistic Regression

- Run logistic regression and ridge regression on dataset, testing StandardScaler() and MinMaxScaler() as ways to transform values.
- Test ridge regression with alpha 0.01 and 0.1.
- Measured in Root Mean Squared Error.

	Algo	Transformation	Alpha	Train RMSE	Test RMSE
0	LogReg	StandardScaler	N/A	13453.41	52697.34
1	LogReg	MinMaxScaler	N/A	59848.81	69646.98
2	Ridge	StandardScaler	0.01	32348.58	53090.67
3	Ridge	MinMaxScaler	0.01	32348.70	53054.91
4	Ridge	StandardScaler	0.1	32348.84	53047.04
5	Ridge	MinMaxScaler	0.1	32757.51	51135.92

SVM

- Use grid search to find optimal parameters.
- Tested C=1,5,10
  kernel=linear,poly,rbf
- Test score measured in RMSE

	param_C	param_kernel	mean_test_score
0	1	linear	39840.070462
1	1	poly	64095.892419
2	1	rbf	63357.569956
3	5	linear	39847.560374
4	5	poly	45680.937634
5	5	rbf	44780.844998
6	10	linear	39847.560374
7	10	poly	39898.903398
8	10	rbf	44916.430972

Neural Network

- Multi-Layer Perceptron with (6,6) hidden layers, with relu activation.
- Tested alphas 0.0001, 0.001, 0.01.
- Issues: more complex model overfit drastically, with incredibly low train error but high test error. Settled at (6,6) as a compromise.

	Algo	Alpha	Train RMSE	Test RMSE
0	MLP	0.0001	46452.13	72805.58
1	MLP	0.001	30972.09	64610.52
2	MLP	0.01	32739.34	59847.55

Results

Logistic Regression:

- StandardScaler and MinMaxScaler performed similarly. StandardScale better for Logistic regression, MinMaxScaler better for Ridge regression.
- Changing alpha did not change error much.
- Best result with Ridge, alpha=0.1, MinMaxScaler.

	Algo	Transformation	Alpha	Train RMSE	Test RMSE
0	LogReg	StandardScaler	N/A	13453.41	52697.34
1	LogReg	MinMaxScaler	N/A	59848.81	69646.98
2	Ridge	StandardScaler	0.01	32348.58	53090.67
3	Ridge	MinMaxScaler	0.01	32348.70	53054.91
4	Ridge	StandardScaler	0.1	32348.84	53047.04
5	Ridge	MinMaxScaler	0.1	32757.51	51135.92

Results

SVM:

- Linear kernel appears to work the best.
- Higher C better for poly,rbf. Did not matter for linear.
- Best results with C=1 and linear kernel.

	param_C	param_kernel	mean_test_score
0	1	linear	39840.070462
1	1	poly	64095.892419
2	1	rbf	63357.569956
3	5	linear	39847.560374
4	5	poly	45680.937634
5	5	rbf	44780.844998
6	10	linear	39847.560374
7	10	poly	39898.903398
8	10	rbf	44916.430972

Results

Neural Network:

- Model overfits training data. Drastically lower error for training data.
- Higher complex models had incredible amount of overfitting (train RMSE in the hundreds), but lower complexity models had high error.
- Best result appears to be MLP with alpha=0.01.

	Algo	Alpha	Train RMSE	Test RMSE
0	MLP	0.0001	46452.13	72805.58
1	MLP	0.001	30972.09	64610.52
2	MLP	0.01	32739.34	59847.55

Conclusions

- On an average sale price of 185k and standard deviation of 83k, the best model (SVM) had MRSE of 39k. Not too bad.

- Trained on 845 rows, 26 columns (down from 80). Did not analyze most columns.

- Models would work better if all categorical data was used, unfortunately didn't have the time.