

House Prices

Ivan Alexeev

3 сентября 2020 г.

1. Introduction

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

2. Data acquisition and cleaning

The following dataset contains 79 columns:

1. MSSubClass: Identifies the type of dwelling involved in the sale.
2. MSZoning: Identifies the general zoning classification of the sale.
3. LotFrontage: Linear feet of street connected to property
4. LotArea: Lot size in square feet
5. Street: Type of road access to property
6. Alley: Type of alley access to property
7. LotShape: General shape of property
8. LandContour: Flatness of the property
9. Utilities: Type of utilities available
10. LotConfig: Lot configuration
11. LandSlope: Slope of property
12. Neighborhood: Physical locations within Ames city limits
13. Condition1: Proximity to various conditions
14. Condition2: Proximity to various conditions (if more than one is present)
15. BldgType: Type of dwelling
16. HouseStyle: Style of dwelling

17. OverallQual: Rates the overall material and finish of the house
18. OverallCond: Rates the overall condition of the house
19. YearBuilt: Original construction date
20. YearRemodAdd: Remodel date (same as construction date if no remodeling or additions)
21. RoofStyle: Type of roof
22. RoofMatl: Roof material
23. Exterior1st: Exterior covering on house
24. Exterior2nd: Exterior covering on house (if more than one material)
25. MasVnrType: Masonry veneer type
26. MasVnrArea: Masonry veneer area in square feet
27. ExterQual: Evaluates the quality of the material on the exterior
28. ExterCond: Evaluates the present condition of the material on the exterior
29. Foundation: Type of foundation
30. BsmtQual: Evaluates the height of the basement
31. BsmtCond: Evaluates the general condition of the basement
32. BsmtExposure: Refers to walkout or garden level walls
33. BsmtFinType1: Rating of basement finished area
34. BsmtFinSF1: Type 1 finished square feet
35. BsmtFinType2: Rating of basement finished area (if multiple types)
36. BsmtFinSF2: Type 2 finished square feet
37. BsmtUnfSF: Unfinished square feet of basement area
38. TotalBsmtSF: Total square feet of basement area
39. Heating: Type of heating
40. HeatingQC: Heating quality and condition
41. CentralAir: Central air condition
42. Electrical: Electrical system
43. 1stFlrSF: First Floor square feet
44. 2ndFlrSF: Second floor square feet
45. LowQualFinSF: Low quality finished square feet (all floors)
46. GrLivArea: Above grade (ground) living area square feet
47. BsmtFullBath: Basement full bathrooms

48. BsmtHalfBath: Basement half bathrooms
49. FullBath: Full bathrooms above grade
50. HalfBath: Half baths above grade
51. Bedroom: Bedrooms above grade (does NOT include basement bedrooms)
52. Kitchen: Kitchens above grade
53. KitchenQual: Kitchen quality
54. TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
55. Functional: Home functionality (Assume typical unless deductions are warranted)
56. Fireplaces: Number of fireplaces
57. FireplaceQu: Fireplace quality
58. GarageType: Garage location
59. GarageYrBlt: Year garage was built
60. GarageFinish: Interior finish of the garage
61. GarageCars: Size of garage in car capacity
62. GarageArea: Size of garage in square feet
63. GarageQual: Garage quality
64. GarageCond: Garage condition
65. PavedDrive: Paved driveway
66. WoodDeckSF: Wood deck area in square feet
67. OpenPorchSF: Open porch area in square feet
68. EnclosedPorch: Enclosed porch area in square feet
69. 3SsnPorch: Three season porch area in square feet
70. ScreenPorch: Screen porch area in square feet
71. PoolArea: Pool area in square feet
72. PoolQC: Pool quality
73. Fence: Fence quality
74. MiscFeature: Miscellaneous feature not covered in other categories
75. MiscVal: Value of miscellaneous feature
76. MoSold: Month Sold (MM)
77. YrSold: Year Sold (YYYY)
78. SaleType: Type of sale
79. SaleCondition: Condition of sale

Out[8]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	Mo
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	

5 rows x 81 columns

In [9]:

```
test = pd.read_csv('test.csv')
test.head()
```

Out[9]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeat
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	h
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	G
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	h
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	h
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN	NaN	h

5 rows x 80 columns

Рис. 1. Data example

Out[8]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	Mo
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	

5 rows x 81 columns

In [9]:

```
test = pd.read_csv('test.csv')
test.head()
```

Out[9]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeat
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	h
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	G
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	h
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	h
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN	NaN	h

5 rows x 80 columns

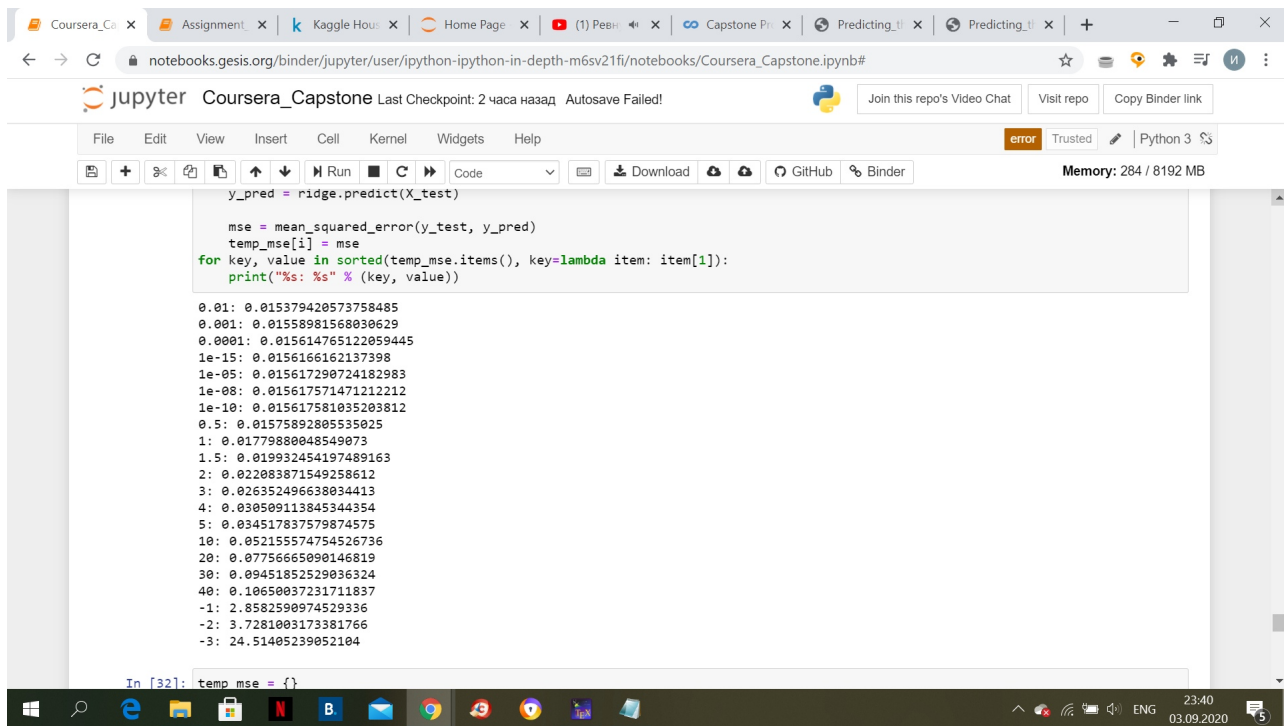
Рис. 2. Types of data

2.1. Data cleaning

First, remove all columns with more than 50% of the missing values. The next step was to get rid of other missing values. To do this, I used the attribute fillna(method='pad', inplace=True). Then, using the get dummies attribute, I got rid of the non-numeric values.

3. Predictive modeling

In my work I used two modeling methods: Lasso regression and Ridge regression. I considered several different parameters and chose the best one. Of all the options, the best was Lasso regression with parameter 0.0001. As a quality assessment, I used mean squared error.



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook is titled 'Coursera_Capstone' and shows the last checkpoint as '2 часа назад' (2 hours ago). The code cell contains the following Python code:

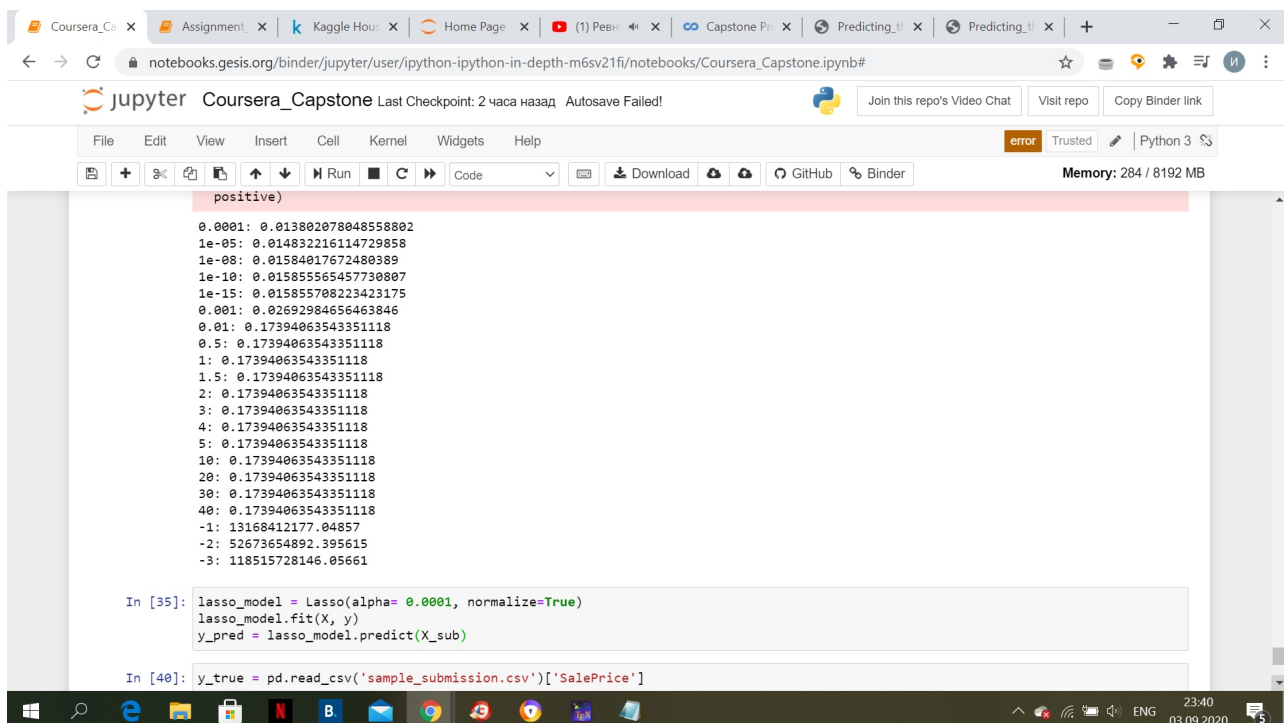
```
y_pred = ridge.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
temp_mse[i] = mse
for key, value in sorted(temp_mse.items(), key=lambda item: item[1]):
    print("%s: %s" % (key, value))
```

The output of the code is a list of values for different lambda values, sorted by their corresponding mean squared error (MSE). The values are:

```
0.01: 0.015379420573758485
0.001: 0.01558981568030629
0.0001: 0.015614765122059445
1e-15: 0.0156166162137398
1e-05: 0.015617290724182983
1e-08: 0.015617571471212212
1e-10: 0.015617581035203812
0.5: 0.01575892805535025
1: 0.01779880048549073
1.5: 0.019932454197489163
2: 0.022083871549258612
3: 0.026352496638034413
4: 0.030509113845344354
5: 0.034517837579874575
10: 0.052155574754526736
20: 0.0775665090146819
30: 0.09451852529036324
40: 0.10650037231711837
-1: 2.8582590974529336
-2: 3.7281003173381766
-3: 24.51405239052104
```

Рис. 3. Ridge regression



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook is titled 'Coursera_Capstone' and shows the last checkpoint as '2 часа назад' (2 hours ago). The code cell contains the following Python code:

```
lasso_model = Lasso(alpha=0.0001, normalize=True)
lasso_model.fit(X, y)
y_pred = lasso_model.predict(X_sub)
```

The output of the code is a list of values for different alpha values, sorted by their corresponding mean squared error (MSE). The values are:

```
0.0001: 0.013802078048558802
1e-05: 0.014832216114729858
1e-08: 0.01584017672480389
1e-10: 0.015855565457730807
1e-15: 0.01585708223423175
0.001: 0.02692984656463846
0.01: 0.17394063543351118
0.5: 0.17394063543351118
1: 0.17394063543351118
1.5: 0.17394063543351118
2: 0.17394063543351118
3: 0.17394063543351118
4: 0.17394063543351118
5: 0.17394063543351118
10: 0.17394063543351118
20: 0.17394063543351118
30: 0.17394063543351118
40: 0.17394063543351118
-1: 13168412177.04857
-2: 52673654892.395615
-3: 118515728146.05661
```

The next code cell contains the following Python code:

```
In [35]: lasso_model = Lasso(alpha=0.0001, normalize=True)
lasso_model.fit(X, y)
y_pred = lasso_model.predict(X_sub)
```

The output of the code is a list of values for different alpha values, sorted by their corresponding mean squared error (MSE). The values are:

```
In [40]: y_true = pd.read_csv('sample_submission.csv')['SalePrice']
```

Рис. 4. Lasso regression

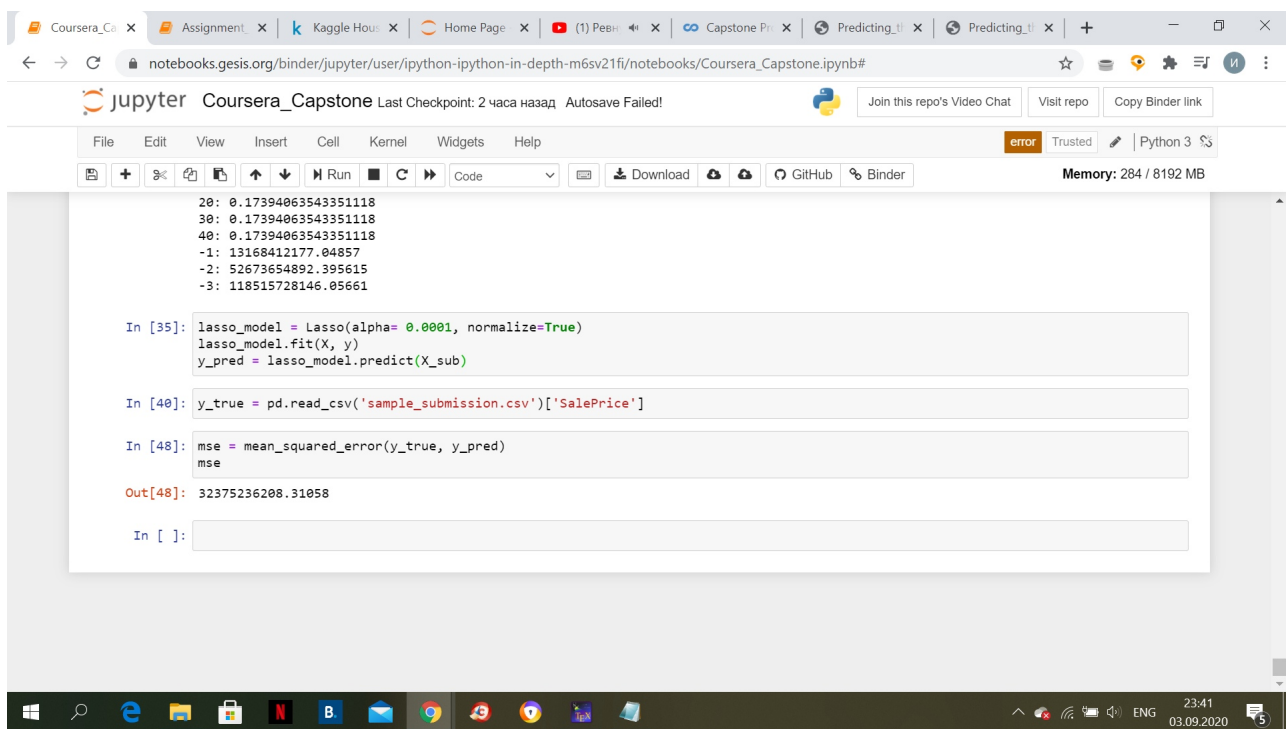


Рис. 5. Result