learn-co-curriculum / dsc-multicollinearity-of-features-lab Public

View license

1 star 171 forks

Star Watch

Code Issues Pull requests Actions Projects Security Insights

solution ...

This branch is 7 commits ahead, 12 commits behind master.

hoffm386 fix typo and adjust spacing in objectives ... on Jul 15 12

View code

README.md

Multicollinearity of Features - Lab

Introduction

In this lab, you'll identify multicollinearity in the Ames Housing dataset.

Objectives

You will be able to:

- Create a scatter matrix and correlation matrix
- Assess and interpret the output of a correlation matrix
- Identify if variables are exhibiting collinearity
- Decide how to address the collinearity in the data set

Correlation matrix for the Ames Housing data

Import data

Let's reimport the Ames Housing data assign the numeric variables we want to keep to `numeric_vars`.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

pd.options.display.max_columns = 999

ames = pd.read_csv('ames.csv')

numeric_vars = ['LotFrontage', 'LotArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF',
                'TotalBsmtSF', '1stFlrSF', '2ndFlrSF','LowQualFinSF', 'GrLivArea',
                'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvG
                'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'Garage
                'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPor
                'ScreenPorch', 'PoolArea']
```

Create processed

Create a new dataframe named `ames_preprocessed` that contains only the features in `numeric_vars`.

```
ames_preprocessed = ames.loc[:,numeric_vars]
ames_preprocessed.head()
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

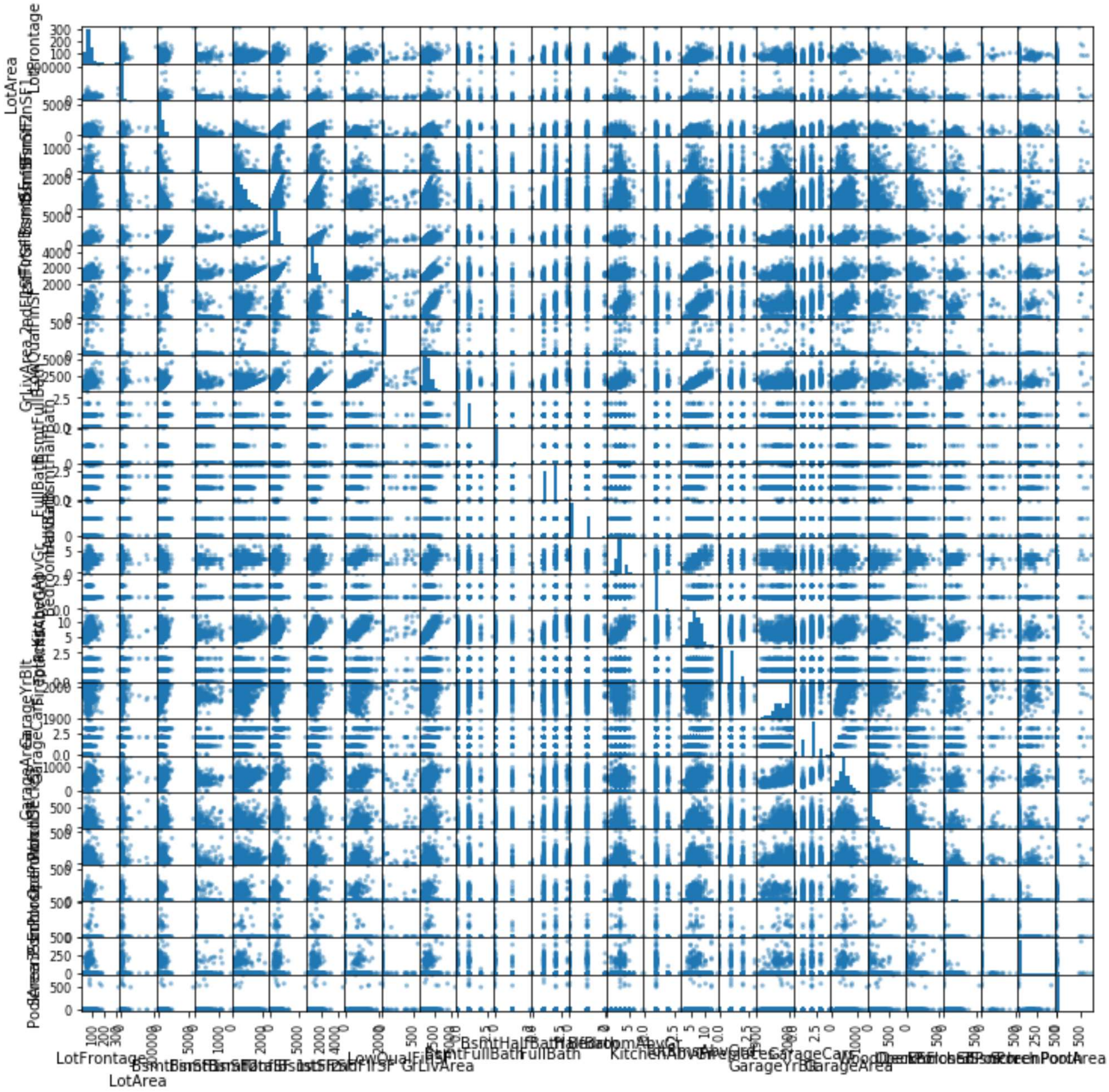
</style>

	LotFrontage	LotArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsm
0	65.0	8450	706	0	150	856
1	80.0	9600	978	0	284	1262
2	68.0	11250	486	0	434	920
3	60.0	9550	216	0	540	756
4	84.0	14260	655	0	490	1145

Scatter matrix

Create the scatter matrix for the Ames Housing data. This takes a few minutes to load!

```
pd.plotting.scatter_matrix(ames_preprocessed, figsize=[12, 12]);
```



The scatter matrix took a while to load and is hard to read. Run the code below to see if adjusting some of the visualization settings helps.

The enhanced plot demonstrates that with larger datasets, scatter matricies become less useful. Through careful examination of the matrix it's clear that `TotRmsAbvGrd` seems correlated with `GrLivArea` , but how easy to use would this matrix if a dataset has hundreds or thousands of variables? Also visual approach to finding correlation cannot be automated, so a numeric approach is a good next step.

Correlation matrix

Next, create and look at the correlation matrix:

```
ames_preprocessed.corr()
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	LotFrontage	LotArea	BsmtFinSF1	BsmtFinSF2	BsmtUn
LotFrontage	1.000000	0.426095	0.233633	0.049900	0.13264
LotArea	0.426095	1.000000	0.214103	0.111170	-0.0026
BsmtFinSF1	0.233633	0.214103	1.000000	-0.050117	-0.4952
BsmtFinSF2	0.049900	0.111170	-0.050117	1.000000	-0.2092
BsmtUnfSF	0.132644	-0.002618	-0.495251	-0.209294	1.00000
TotalBsmtSF	0.392075	0.260833	0.522396	0.104810	0.41536
1stFlrSF	0.457181	0.299475	0.445863	0.097117	0.31798
2ndFlrSF	0.080177	0.050986	-0.137079	-0.099260	0.00446
LowQualFinSF	0.038469	0.004779	-0.064503	0.014807	0.02816
GrLivArea	0.402797	0.263116	0.208171	-0.009640	0.24025
BsmtFullBath	0.100949	0.158155	0.649212	0.158678	-0.4229
BsmtHalfBath	-0.007234	0.048046	0.067418	0.070948	-0.0958
FullBath	0.198769	0.126031	0.058543	-0.076444	0.28888
HalfBath	0.053532	0.014259	0.004262	-0.032148	-0.0411
BedroomAbvGr	0.263170	0.119690	-0.107355	-0.015728	0.16664
KitchenAbvGr	-0.006069	-0.017784	-0.081007	-0.040751	0.03008
TotRmsAbvGrd	0.352096	0.190015	0.044316	-0.035227	0.25064

	LotFrontage	LotArea	BsmtFinSF1	BsmtFinSF2	BsmtUn
Fireplaces	0.266639	0.271364	0.260011	0.046921	0.05157
GarageYrBlt	0.070250	-0.024947	0.153484	-0.088011	0.19070
GarageCars	0.285691	0.154871	0.224054	-0.038264	0.21417
GarageArea	0.344997	0.180403	0.296970	-0.018227	0.18330
WoodDeckSF	0.088521	0.171698	0.204306	0.067898	-0.0053
OpenPorchSF	0.151972	0.084774	0.111761	0.003093	0.12900
EnclosedPorch	0.010700	-0.018340	-0.102303	0.036543	-0.0025
3SsnPorch	0.070029	0.020423	0.026451	-0.029993	0.02076
ScreenPorch	0.041383	0.043160	0.062021	0.088871	-0.0125
PoolArea	0.206167	0.077672	0.140491	0.041709	-0.0350

Return `True` for positive or negative correlations that are bigger than 0.75 in the correlation matrix:

```
abs(ames_preprocessed.corr()) > 0.75
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	LotFrontage	LotArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfS
LotFrontage	True	False	False	False	False
LotArea	False	True	False	False	False
BsmtFinSF1	False	False	True	False	False
BsmtFinSF2	False	False	False	True	False

	LotFrontage	LotArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfS
BsmtUnfSF	False	False	False	False	True
TotalBsmtSF	False	False	False	False	False
1stFlrSF	False	False	False	False	False
2ndFlrSF	False	False	False	False	False
LowQualFinSF	False	False	False	False	False
GrLivArea	False	False	False	False	False
BsmtFullBath	False	False	False	False	False
BsmtHalfBath	False	False	False	False	False
FullBath	False	False	False	False	False
HalfBath	False	False	False	False	False
BedroomAbvGr	False	False	False	False	False
KitchenAbvGr	False	False	False	False	False
TotRmsAbvGrd	False	False	False	False	False
Fireplaces	False	False	False	False	False
GarageYrBlt	False	False	False	False	False
GarageCars	False	False	False	False	False
GarageArea	False	False	False	False	False
WoodDeckSF	False	False	False	False	False
OpenPorchSF	False	False	False	False	False
EnclosedPorch	False	False	False	False	False
3SsnPorch	False	False	False	False	False
ScreenPorch	False	False	False	False	False
PoolArea	False	False	False	False	False

Now, include `stack` and `zip` to create a more robust solution that will return the variable pairs from the correlation matrix that have correlations over .75, but less than 1.

```
df = ames_preprocessed.corr().abs().stack().reset_index().sort_values(0, ascending=F

df['pairs'] = list(zip(df.level_0, df.level_1))

df.set_index(['pairs'], inplace = True)

df.drop(columns=['level_1', 'level_0'], inplace = True)

# cc for correlation coefficient
df.columns = ['cc']

df.drop_duplicates(inplace=True)

df[(df.cc>.75) & (df.cc<1)]
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	cc
pairs	
(GarageArea, GarageCars)	0.882475
(TotRmsAbvGrd, GrLivArea)	0.825489
(1stFlrSF, TotalBsmtSF)	0.819530

Which varibles are highly correlated in the Ames Housing data set?

```
"""
There are three sets of variales that are highly correlated.

Garage Area with Garage Cars, Total Rooms Above Ground with Total Square Feet of Liv
```

```
  """
```

Make a data decision

Now that you know which variables are correlated with eachother, which would you drop from the dataset?

Address the colinearity

Remove the chosen variables from `ames_preprocessed` .

```
ames_preprocessed.drop(columns=['GarageArea','TotRmsAbvGrd','TotalBsmtSF'], inplace=
```

Summary

Good job! You got some hands-on practice creating and interpreting a scatter matrix and correlation matrix to identify if variables are collinear in the Ames Housing data set. You also edited the Ames Housing data set so highly correlated variables are removed.

Releases

No releases published

Packages

No packages published

Contributors 6

Languages

- ● **Jupyter Notebook** 99.9%
- ● **Python** 0.1%