House Prices dataset: Feature Selection

In the following cells, we will select a group of variables, the most predictive ones, to build our machine learning model.

Why do we select variables?

- For production: Fewer variables mean smaller client input requirements (e.g. customers filling out a form on a website or mobile app), and hence less code for error handling. This reduces the chances of introducing bugs.

- For model performance: Fewer variables mean simpler, more interpretable, better generalizing models

We will select variables using the Lasso regression: Lasso has the property of setting the coefficient of non-informative variables to zero. This way we can identify those variables and remove them from our final model.

In [1]:

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import Lasso
from sklearn.feature_selection import SelectFromModel

pd.pandas.set_option('display.max_columns', None)
```

In [2]:

```
X_train = pd.read_csv('xtrain.csv')
X_test = pd.read_csv('xtest.csv')

X_train.head()
```

Out[2]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilitie
0	931	0.000000	0.75	0.461171	0.377048	1.0	1.0	0.333333	1.000000	1.(
1	657	0.000000	0.75	0.456066	0.399443	1.0	1.0	0.333333	0.333333	1.(
2	46	0.588235	0.75	0.394699	0.347082	1.0	1.0	0.000000	0.333333	1.(
3	1349	0.000000	0.75	0.388581	0.493677	1.0	1.0	0.666667	0.666667	1.(
4	56	0.000000	0.75	0.577658	0.402702	1.0	1.0	0.333333	0.333333	1.(

In [4]:

```
# capture the target (remember that the target is log transformed)
y_train = X_train['SalePrice']
y_test = X_test['SalePrice']

# drop unnecessary variables from our training and testing sets
X_train.drop(['Id', 'SalePrice'], axis=1, inplace=True)
X_test.drop(['Id', 'SalePrice'], axis=1, inplace=True)
```

In [5]:
```
# We will do the model fitting and feature selection
# altogether in a few lines of code

# first, we specify the Lasso Regression model, and we
# select a suitable alpha (equivalent of penalty).
# The bigger the alpha the less features that will be selected.

# Then we use the selectFromModel object from sklearn, which
# will select automatically the features which coefficients are non-zero

# remember to set the seed, the random state in this function
selection = SelectFromModel(Lasso(alpha=0.005, random_state=0))

# train Lasso model and select features
selection.fit(X_train, y_train)
```

Out[5]:
```
SelectFromModel(estimator=Lasso(alpha=0.005, copy_X=True, fit_intercept=True,
                                max_iter=1000, normalize=False, positive=False,
                                precompute=False, random_state=0,
                                selection='cyclic', tol=0.0001,
                                warm_start=False),
                max_features=None, norm_order=1, prefit=False, threshold=None)
```

In [6]:
```
selection.get_support()
```

Out[6]:
```
array([ True,  True, False, False, False, False, False, False, False,
       False, False,  True, False, False, False, False,  True,  True,
       False,  True,  True, False, False, False,  True, False, False,
       False, False,  True, False,  True, False, False, False, False,
       False, False, False,  True,  True, False,  True, False, False,
        True,  True, False, False, False, False, False,  True, False,
       False,  True,  True,  True, False,  True,  True, False, False,
       False,  True, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False])
```

In [7]:
```
# print the number of total and selected features

# this is how we can make a list of the selected features
selected_feats = X_train.columns[(selection.get_support())]

# let's print some stats
print('total features: {}'.format((X_train.shape[1])))
print('selected features: {}'.format(len(selected_feats)))
print('features with coefficients shrank to zero: {}'.format(
    np.sum(selection.estimator_.coef_ == 0)))
```

```
total features: 82
selected features: 22
features with coefficients shrank to zero: 60
```

In [8]:
```
selected_feats
```

Out[8]:
```
Index(['MSSubClass', 'MSZoning', 'Neighborhood', 'OverallQual', 'OverallCond',
       'YearRemodAdd', 'RoofStyle', 'MasVnrType', 'BsmtQual', 'BsmtExposure',
       'HeatingQC', 'CentralAir', '1stFlrSF', 'GrLivArea', 'BsmtFullBath',
       'KitchenQual', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageFinish', 'GarageCars', 'PavedDrive'],
      dtype='object')
```

In [9]:
```
pd.Series(selected_feats).to_csv('selected_features.csv', index=False)
```

```
e:\users\user.desktop-3hhgvth\anaconda3\envs\mytfenv\lib\site-packages\ipykernel_launche
r.py:1: FutureWarning: The signature of `Series.to_csv` was aligned to that of `DataFram
e.to_csv`, and argument 'header' will change its default value from False to True: pleas
e pass an explicit value to suppress this warning.
  """Entry point for launching an IPython kernel.
```

In [ ]: