```
A problem with linear regression is that estim
can become large, making the model sensitive t
This is particularly true for problems with fe
input predictors (p) than variables than sampl

One approach to address the stability of regre
loss function to include additional costs for
coefficients. Linear regression models that us
during training are referred to collectively a

Least Angle Regression, LAR or LARS for short,
solving the optimization problem of fitting th
LARS is a forward stepwise version of feature
be adapted for the Lasso model.

Unlike the Lasso, it does not require a hyperp
weighting of the penalty in the loss function.
discovered automatically by LARS.
```

A problem with linear regression is that estimated coefficients of the model can become large, making the model sensitive to inputs and possibly unstable. This is particularly true for problems with few observations (samples) or more input predictors (p) than variables than samples (n).

One approach to address the stability of regression models is to change the loss function to include additional costs for a model that has large coefficients. Linear regression models that use these modified loss functions during training are referred to collectively as penalized linear regression.

Least Angle Regression, LAR or LARS for short, is an alternative approach to solving the optimization problem of fitting the penalized model. Technically, LARS is a forward stepwise version of feature selection for regression that can be adapted for the Lasso model.

Unlike the Lasso, it does not require a hyperparameter that controls the weighting of the penalty in the loss function. Instead, the weighting is discovered automatically by LARS.

```
# load and summarize the housing dataset
from pandas import read_csv
from matplotlib import pyplot
# load dataset
url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'
dataframe = read_csv(url, header=None)
# summarize shape
print(dataframe.shape)
# summarize first few lines
print(dataframe.head())
```

```
(506, 14)
      0     1     2   3     4     5   ...  8      9      10      11    12    13
```

```
0  0.00632  18.0  2.31   0  0.538  6.575  ...   1  296.0  15.3  396.90  4.98  24.0
1  0.02731   0.0  7.07   0  0.469  6.421  ...   2  242.0  17.8  396.90  9.14  21.6
2  0.02729   0.0  7.07   0  0.469  7.185  ...   2  242.0  17.8  392.83  4.03  34.7
3  0.03237   0.0  2.18   0  0.458  6.998  ...   3  222.0  18.7  394.63  2.94  33.4
4  0.06905   0.0  2.18   0  0.458  7.147  ...   3  222.0  18.7  396.90  5.33  36.2

[5 rows x 14 columns]
```

```
# define model
from sklearn.linear_model import Lars
model = Lars()
```

```
# evaluate an lars regression model on the dataset
from numpy import mean
from numpy import std
from numpy import absolute
from pandas import read_csv
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedKFold
from sklearn.linear_model import Lars
# load the dataset
url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'
dataframe = read_csv(url, header=None)
data = dataframe.values
X, y = data[:, :-1], data[:, -1]
# define model
model = Lars()
# define model evaluation method
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
# evaluate model
scores = cross_val_score(model, X, y, scoring='neg_mean_absolute_error', cv=cv, n_jobs=-
# force scores to be positive
scores = absolute(scores)
print('Mean MAE: %.3f (%.3f)' % (mean(scores), std(scores)))
```

```
Mean MAE: 3.432 (0.552)
```

```
# make a prediction with a lars regression model on the dataset
from pandas import read_csv
from sklearn.linear_model import Lars
# load the dataset
url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'
dataframe = read_csv(url, header=None)
data = dataframe.values
X, y = data[:, :-1], data[:, -1]
# define model
model = Lars()
# fit model
model.fit(X, y)
# define new data
```

```
row = [0.00632,18.00,2.310,0,0.5380,6.5750,65.20,4.0900,1,296.0,15.30,396.90,4.98]
# make a prediction
yhat = model.predict([row])
# summarize prediction
print('Predicted: %.3f' % yhat)
```

```
Predicted: 29.904
```

```
# use automatically configured the lars regression algorithm
from numpy import arange
from pandas import read_csv
from sklearn.linear_model import LarsCV
from sklearn.model_selection import RepeatedKFold
# load the dataset
url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'
dataframe = read_csv(url, header=None)
data = dataframe.values
X, y = data[:, :-1], data[:, -1]
# define model evaluation method
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
# define model
model = LarsCV(cv=cv, n_jobs=-1)
# fit model
model.fit(X, y)
# summarize chosen configuration
print('alpha: %f' % model.alpha_)
```

```
alpha: 0.001623
```