How do I perform Lasso Regression in Python, step-by-step?

Authored by

stats writer

April 22, 2024

Lasso Regression is a type of linear regression that is used to perform feature selection and regularize the coefficients in a model. It is widely used in machine learning and data analysis. In order to perform Lasso Regression in Python, follow these steps:

1. Import the necessary libraries: The first step is to import the required libraries like numpy, pandas, and sklearn.

2. Load the dataset: Next, load the dataset that you want to use for Lasso Regression.

3. Preprocess the data: Before performing Lasso Regression, it is important to preprocess the data by handling missing values, scaling the data, and encoding categorical variables.

4. Split the data: Split the dataset into training and testing sets.

5. Fit the model: Use the sklearn Lasso Regression function to fit the model on the training data.

6. Choose the optimal alpha value: Lasso Regression has a regularization parameter, alpha, which controls the amount of regularization applied. Use cross-validation to choose the optimal alpha value.

7. Evaluate the model: Use the testing data to evaluate the performance of the model.

8. Interpret the results: Once the model is trained and evaluated, interpret the results to understand the significance of the features in the model.

Overall, performing Lasso Regression in Python involves importing libraries, preprocessing the data, fitting the model, choosing the optimal alpha value, and evaluating and interpreting the results. Following these steps will help you effectively use Lasso Regression for feature selection and regularization in your data analysis and machine learning projects.

Lasso Regression in Python (Step-by-Step)

Lasso regression is a method we can use to fit a regression model when **multicollinearity** is present in the data.

In a nutshell, least squares regression tries to find

coefficient estimates that minimize the sum of squared residuals (RSS):

RSS = Σ(yi - ?i)2

where:

Σ: A greek symbol that means *sum*yi: The actual response value for the ith observation?i: The predicted response value based on the multiple linear regression model

Conversely, lasso regression seeks to minimize the following:

RSS + λΣ|βj|

where *j* ranges from 1 to *p* predictor variables and λ ≥ 0.

This second term in the equation is known as a *shrinkage penalty*. In lasso regression, we select a value for λ that produces the lowest possible test MSE (mean squared error).

This tutorial provides a step-by-step example of how to perform lasso regression in Python.

Step 1: Import Necessary Packages

First, we'll import the necessary packages to perform lasso regression in Python:

```
import pandas as pd
from numpy import arange
from sklearn.linear_modelimport LassoCV
from sklearn.model_selectionimport RepeatedKFold
```

Step 2: Load the Data

For this example, we'll use a dataset called mtcars, which contains information about 33 different cars. We'll use hp as the response variable and the following variables as the predictors:

mpgwtdratqsec

The following code shows how to load and view this dataset:

```
#define URL where data is located
url = "https://raw.githubusercontent.com/Statology/Python-Guides/main/mtcars.csv"
```

```
#read in data
data_full = pd.read_csv(url)

#select subset of data
data = data_full]

#view first six rows of data
data

mpg wt drat qsec hp
0 21.0 2.620 3.90 16.46 110
1 21.0 2.875 3.90 17.02 110
2 22.8 2.320 3.85 18.61 93
3 21.4 3.215 3.08 19.44 110
4 18.7 3.440 3.15 17.02 175
5 18.1 3.460 2.76 20.22 105
```


Step 3: Fit the Lasso Regression Model

Next, we'll use the **LassoCV()** function from sklearn to fit the lasso regression model and we'll use the **RepeatedKFold()** function to perform k-fold cross-validation to find the optimal alpha value to use for the penalty term.

Note: The term "alpha" is used instead of "lambda" in

Python.

For this example we'll choose k = 10 folds and repeat the cross-validation process 3 times.

Also note that LassoCV() only tests alpha values 0.1, 1, and 10 by default. However, we can define our own alpha range from 0 to 1 by increments of 0.01:

```
#define predictor and response variables
X = data]
y = data

#define cross-validation method to evaluate model
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

#define model
model = LassoCV(alphas=arange(0, 1, 0.01), cv=cv, n_jobs=-1)

#fit model
model.fit(X, y)

#display lambda that produced the lowest test MSE
print(model.alpha_)
```

0.99

The lambda value that minimizes the test MSE turns out to be 0.99.

Step 4: Use the Model to Make Predictions

Lastly, we can use the final lasso regression model to make predictions on new observations. For example, the following code shows how to define a new car with the following attributes:

mpg: 24wt: 2.5drat: 3.5qsec: 18.5

The following code shows how to use the fitted lasso regression model to predict the value for *hp* of this new observation:

```
#define new observation
new =

#predict hp value using lasso regression model
model.predict()
array()
```

Based on the input values, the model predicts this car

to have an *hp* value of 105.63442071.

You can find the complete Python code used in this example here.