Linear Regression with Normal Equations Algorithm (with Lasso)

Table of Contents

Clearing and closing the figures

```
close all;
clc;
clear all;
```

Loading data set and visualizing it

```
fprintf('Loading dataset...\n\n');
load('comp.mat');
```

```
Loading dataset...
```

Performing feature selection to avoid regularization

```
fprintf('Linear Regression using Gradient Descent.....\n');
fprintf('Performing feature selection using Lasso.....\n');
P = lasso(comp(:,1:7),comp(:,8));

% 3 important features selected having performed lasso to get the
 relevant
% features

X(:,1) = comp(:,2);
X(:,2) = (comp(:,3).^2) / 4;
X(:,3) = comp(:,7);
% disp(size(X));

[X, mu, sigma] = normalize(X);
```

```
Linear Regression using Gradient Descent.....
```

Performing feature selection using Lasso.....

Creating training and testing datasets for the learning algorithm

```
Xtrain = X(1:180,:); % disp(size(Xtrain));
Xtest = X(181:end,:); % disp(size(Xtest));
ytrain = comp(1:180,8); % disp(size(ytrain));
ytest = comp(181:end,8); % disp(size(ytest));
m = length(ytrain);

% adding intercept term
Xtrain = [ones(m,1) Xtrain];
%disp(size(Xtrain));
```

Applying normal equations

```
fprintf('Linear Regression using Normal equations....\n');
% Acquiring parameters using normal equations
[theta] = normalEqn(Xtrain, ytrain);
```

Linear Regression using Normal equations....

Training accuracy of our algorithm using Normal equations

```
% Applying learnt parameters on test data
pricetr = Xtrain * theta;

% Showing algorithm accuracy
errortr = abs(pricetr - ytrain) ./ ytrain; % error between actual and
 predicted
accuracytr = 100 - (mean(errortr) * 100); % percentage accuracy
 obtained
fprintf('Training accuracy on training set: %f\n', accuracytr);
```

Training accuracy on training set: 83.532594

Testing our learnt algorithm using Normal equations

```
% Applying learnt parameters on test data
Xtest = [ones(size(Xtest,1),1) Xtest];
%disp(size(Xtest));
pricete = Xtest * theta;

% Showing algorithm accuracy
errorte = abs(pricete - ytest) ./ ytest;  % error between actual and
 predicted values
```

```
accuracyte = 100 - (mean(errorte) * 100); % percentage accuracy
 obtained
fprintf('Testing accuracy on test set: %f\n', accuracyte);
```

```
Testing accuracy on test set: 82.554708
```

Published with MATLAB® R2016b