

function ungradedRandomPlotting(X, y, Xval, yval)	
	% we'll check the cross validation error
	m = size(X, 1);
	% vectors for storing training and validation errors
	error_train = zeros(m,1);
	error_val = zeros(m,1);
	% the lambda
	lambda = 0.01;
	% number of times to loop for different lambda values
	loops = 20;
	% pick a random value "loops" times
	for l=1:loops
	% Concretely, to determine the training and validation sets, we randomly select
	% i examples, (:1a:) for the training set and the remaining
	% (:1b:) and i examples for the validation set.
	% Compute the training error using the linear model with parameters theta using
	% theta on the training set.
	% i = number of training examples
	for i=1:m
	% ---
	% test set
	% ---
	% (:1a:) random permutation of indices 1:m
randperm(N) 生成一个1到N的无重复整数的随机排列	sel = randperm(m);
取前n个	sel = sel(1:i);
	% create a random vector of size m

	y_sel = y(sel)
	% (:2:) learn
	theta = train(theta, X_sel, y_sel)
	% (:3a:) evaluate
	[J, grad] = cost(theta, X_val, y_val)
	% accumulate
	error_train(i) = J
	% ---
	% cross validation
	% ---
	% (:1b:) ...
	sel = randperm(m, 1000)
	sel = sel(1:1000)
	X_sel = X_val(sel, :)
	y_sel = y_val(sel)
	% (:3b:) ...
	[J, grad_val] = cost(theta, X_val, y_val)
	error_val(i) = J
	end
	end
	% finding the average
	error_train = error_train ./ length(error_train)
	error_val = error_val ./ length(error_val)
	% least but not last, do some
	plot(1:m, error_train, 1:m, 'b')
	xlabel('Number of training examples')
	ylabel('Error');

	<code>axis([0 13 0 100]);</code>
	<code>legend('Train', 'Cross Valid</code>
	<code>end</code>

## Learning Curve学习曲线

```
function [error_train, error_val] = learningCurve(X, y, Xval, yval, lambda)
```

```
% Number of training examples
```

```
m = size(X, 1);
```

```
% You need to return these values correctly
```

```
error_train = zeros(m, 1);
```

```
error_val = zeros(m, 1);
```

```
% ----- Sample Solution -----
```

```
for i = 1:m,
```

```
    theta = trainLinearReg(X(1:i,:), y(1:i,:), lambda);
```

```
    error_train(i) = linearRegCostFunction(X(1:i,:), y(1:i,:), theta, 0);
```

```
    error_val(i) = linearRegCostFunction(Xval, yval, theta, 0);
```

```
end
```

## Validation Curve验证曲线

```
function [lambda_vec, error_train, error_val] = validationCurve(X, y, Xval, yval)
```

```
% Selected values of lambda (you should not change this)
```

```
lambda_vec = [0 0.001 0.003 0.01 0.03 0.1 0.3 1 3 10]';
```

```
% You need to return these variables correctly.
```

```
error_train = zeros(length(lambda_vec), 1);
```

```
error_val = zeros(length(lambda_vec), 1);
```

```
for i = 1:length(lambda_vec),
```

```
    lambda = lambda_vec(i);
```

```
    theta = trainLinearReg(X, y, lambda);
```

```
    error_train(i) = linearRegCostFunction(X, y, theta, 0);
```

```
    error_val(i) = linearRegCostFunction(Xval, yval, theta, 0);
```

```
end
```

