Machine Learning Homework

ZiUNO

2019 年 4 月 8 日

目录

1 Homework **2**
1.1 Plotting 2
1.1.1 plotData 2
1.2 Gradient descent 2
1.2.1 computeCost 2
1.2.2 gradientDescent 3
1.3 Visualizing J(theta_0, theta_1) 3
1.4 Feature Normalization 3
1.5 Gradient Descent 3
1.5.1 gradientDescentMulti 3
1.5.2 ex1_multi 3
1.6 Normal Equations 4
1.6.1 normalEqn 4
1.6.2 ex1_multi 4

2 Homework **4**
2.1 Plotting 4
2.1.1 plotData 4
2.2 Compute Cost and Gradient 4
2.2.1 sigmoid 4
2.2.2 costFunction 5
2.3 Optimizing using fminunc 5
2.4 Predict and Accuracie 5
2.4.1 predict 5

2.5 Regularized Logistic Regression 5
2.5.1 costFunctionReg 5
2.6 Regularization and Accuracies 6
2.6.1 ex2_reg 6

3 Homework 6
3.1 Find Closest Centroids 6
3.1.1 findClosestCentroids 6
3.2 Compute Means 6
3.2.1 computeCentroids 6
3.3 K-Means Clustering 6
3.4 Load Example Dataset 6
3.5 Principal Component Analysis 6
3.5.1 pca 6
3.6 Dimension Reduction 7
3.6.1 projectData 7
3.6.2 recoverData 7

1 Homework

1.1 Plotting

1.1.1 plotData

```
plot(x, y, 'rx', 'MarkerSize', 10);
xlabel('Population of Ciry in 10,000s');
ylabel('Profit in $10,000s');
```

1.2 Gradient descent

1.2.1 computeCost

```
J = 1 / (2 * m) * sum((x * theta - y).^2);
```

1.2.2 gradientDescent

```
p = theta(1) - alpha * ...
  1 / m * sum((X * theta - y).* X(:,1));
q = theta(2) - alpha * ...
  1 / m * sum((X * theta - y).* X(:,2));
theta(1) = p;
theta(2) = q;
```

1.3 Visualizing J(theta_0, theta_1)

1.4 Feature Normalization

```
mu = mean(X);
sigma = std(X);
X_norm = (X - repmat(mu,size(X, 1), 1))./...
  (repmat(sigma, size(X, 1), 1));
```

1.5 Gradient Descent

1.5.1 gradientDescentMulti

```
theta = theta - alpha * ...
  1.0 / m * X' * (X * theta - y);
```

1.5.2 ex1_multi

```
alpha = 1.1;

X = [1650, 3];
X = (X - mu)./sigma;
X = [ones(1, 1), X];
```

```
price = X * theta;
```

1.6 Normal Equations

1.6.1 normalEqn

```
theta = pinv(X' * X) * X' * y;
```

1.6.2 ex1_multi

```
X = [1, 1650, 3];
price = X * theta;
```

2 Homework

2.1 Plotting

2.1.1 plotData

```
X_neg = X.*repmat(y, 1, 2);
X_pos = X.*repmat((ones(size(y, 1), 1) - y), ...
  1, 2);
X_neg(all(X_neg==0, 2), :) = [];
X_pos(all(X_pos==0, 2), :) = [];
plot(X_neg(:, 1), X_neg(:, 2), 'k+',...
  X_pos(:, 1), X_pos(:, 2), 'ko',...
  'MarkerFaceColor', 'y');
```

2.2 Compute Cost and Gradient

2.2.1 sigmoid

```
g = 1.0 ./ (1.0 + exp(-z));
```

2.2.2 costFunction

```
h = sigmoid(X * theta);
c0 = - log(1 - h);
c1 = - log(h);
J = 1 / m * (y' * c1 + (1 - y)'* c0);
grad = 1.0 / m * (X' * (h - y));
```

2.3 Optimizing using fminunc

2.4 Predict and Accuracie

2.4.1 predict

```
h = sigmoid(X * theta);
p(h >= 0.5, 1) = 1;
p(h < 0.5, 1) = 0;
```

2.5 Regularized Logistic Regression

2.5.1 costFunctionReg

```
h = sigmoid(X * theta);
c0 = - log(1 - h);
c1 = - log(h);
J = 1.0 / m * ( y' * c1 + ( 1 - y )' * c0)...
  + lambda / (2 * m) * norm(theta(2:end))^2;
reg = (lambda/m) .* theta;
reg(1) = 0;
grad = 1.0 / m * (X' * (h - y)) + reg;
```

2.6 Regularization and Accuracies

2.6.1 ex2_reg

```
lambda = 1;
```

3 Homework

3.1 Find Closest Centroids

3.1.1 findClosestCentroids

```
for i = 1:size(X, 1)
    Xi = repmat(X(i, :), K, 1);
    dis = sqrt(sum((Xi - centroids).^2, 2));
    [~, idx(i, :)] = min(dis);
end
```

3.2 Compute Means

3.2.1 computeCentroids

```
for i = 1:K
    centroids(i, :) = mean(X(idx == i, :), 1);
end
```

3.3 K-Means Clustering

3.4 Load Example Dataset

3.5 Principal Component Analysis

3.5.1 pca

```
cova = X' * X / m;
[U, S, ~] = svd(cova);
```

3.6 Dimension Reduction

3.6.1 projectData

```
Z = X * U(:, 1:K);
```

3.6.2 recoverData

```
X_rec = Z * U(:, 1:K)';
```