1.

(1)

Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost function:

$$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Gradient descent:

Repeat $\{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \ldots, \theta_n)$$

$\}$ (simultaneously update for every $j = 0, \ldots, n$)

New algorithm $(n \geq 1)$:

Repeat $\{$

$$\frac{\partial}{\partial \theta_0} J(\theta)$$

$$\theta_j := \theta_j - \alpha \boxed{\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}}$$

(simultaneously update θ_j for $j = 0, \ldots, n$)

$\}$

$x_0^{(i)} = 1$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$\ldots$

根据以上公式，将θ^0初始值全为 0，学习率α = 1，代入，即可求出

θ^1 = [93 8376 6864.6 8059.8 8501.8]。

(2)

J(θ^0) = 4328.5, J(θ^1) = 3.7431 * 10^12。

J(θ^1) > J(θ^0); 所以 θ^1 不可以使线性回归中的代价函数 $J(\boldsymbol{\theta})$下降。

(3)

可以。将学习率 α 的值取小一些。比如 0.000034（这是经过实践测出来的较好的一个值）。代入算出第一次迭代后的代价 $J(\boldsymbol{\theta})$为 1.7808。

```
for i=1:iterNum
    theta = theta - 0.000034 * grad;
    [cost, grad] = costFunction(theta, X, y);
end;
disp(theta);
```

```
function [J, grad] = costFunction(theta, X, y)
m = length(y); % number of training examples
J = 0;
grad = zeros(size(theta));
for i = 1:m
    J = J + (theta' * X(i,:)' - y(i))^2;
    grad = grad + (theta' * X(i,:)' - y(i)) * X(i,:)';
end
J = J / (2*m);
disp(J);
grad = grad ./ m;
end
```

(4)

编辑器 - D:\Program Files\MATLAB\R2016a\bin\linearRegression.m

linearRegression.m × logisticRegression.m × +

```
function linearRegression()
x = [1 87 72 83 90;1 89 76 88 93;1 89 74 82 91;1 92 71 91 89;1 93 76 89 94]
y = [89;91;93;95;97]
x1 = x'
x2 = x1 * x
x3 = inv(x2)
result1 = x3 * x1 * y

a = [0 0 0 0 0;0 1 0 0 0;0 0 1 0 0;0 0 0 1 0;0 0 0 0 1]
x4 = x2 + a
x5 = inv(x4)
result2 = x5 * x1 * y
end
```

根据标准化方程算法求得的最优的多元线性回归方程为:
m = -19.50+1.69p+0.38c-0.31e-0.44ch;
代入数据求得要求的同学数学分数 m = 89.51。
(5)

If $\lambda > 0$,

$$\theta = \Bigg(\underbrace{X^T X + \lambda \begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}}_{\text{Invertible}}\Bigg)^{-1} X^T y$$

利用标准方程求出最优的 L2 正则化多元线性回归方程为:
m = -19.99+1.47p+0.07c-0.23e-0.06ch;
代入数据求得要求的同学数学分数 m = 88.95。

```cpp
#include <iostream>
#include <cmath>
using namespace std;
int main() {
    double a1,a2,a3,a4,a5,result;
    double t0 = -19.99, t1 = 1.47, t2 = 0.07, t3 = -0.23, t4 = -0.06;
    a1 = pow((t0 * 1 + t1 * 87 + t2 * 72 + t3 * 83 + t4 * 90 - 89), 2);
    a2 = pow((t0 * 1 + t1 * 89 + t2 * 76 + t3 * 88 + t4 * 93 - 91), 2);
    a3 = pow((t0 * 1 + t1 * 89 + t2 * 74 + t3 * 82 + t4 * 91 - 93), 2);
    a4 = pow((t0 * 1 + t1 * 92 + t2 * 71 + t3 * 91 + t4 * 89 - 95), 2);
    a5 = pow((t0 * 1 + t1 * 93 + t2 * 76 + t3 * 89 + t4 * 94 - 97), 2);
    result = (a1 + a2 + a3 + a4 + a5) / 10;
    cout << result << endl;
}
```

对于(4)中方程，求得的代价函数 $J(\boldsymbol{\theta})$的值为 0.16947;
而对于(5)中方程，$J(\boldsymbol{\theta})$的值为 0.46755。
所以(4)中求得的结果更好。
2.
(1)

假设函数 $h_\theta(X) = \dfrac{1}{1+e^{-\theta^T X}}$,

代价函数 $J(\theta) = -\dfrac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta\left(x^{(i)}\right) + (1-y^{(i)})\log(1-h_\theta\left(x^{(i)}\right))\right]$,

通过梯度下降的方法最小化 $J(\theta)$, 即 $\theta_j = \theta_j - \alpha \sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$,

最终求得θ = [-2.6653, 2.2190, 1.0641, -1.7730, 2.2363]。

```
function [J, grad] = costFunction(theta, X, y)
m = length(y); % number of training examples
J = 0;
grad = zeros(size(theta));
for i = 1:m
    J = J + (-y(i)) * log(sigmoid(theta' * X(i,:)')) - (1-y(i)) * log(1-sigmoid(theta' * X(i,:)'));
    grad = grad + (sigmoid(theta' * X(i,:)') - y(i)) * X(i,:)';
end
J = J / m;
grad = grad ./ m;
end
```

(2)

影响子宫内膜癌发病的最直接的因素为是否使用过非雌激素即 Nonest.

(3)

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

求出来的回归模型为，其中求得的 θ 为 [-0.1246, 0.1045, 0.0471, -0.0259, 0.0813]。

部分代码如图：

其中，这里迭代次数选取 20000，学习率 α 取 0.01。

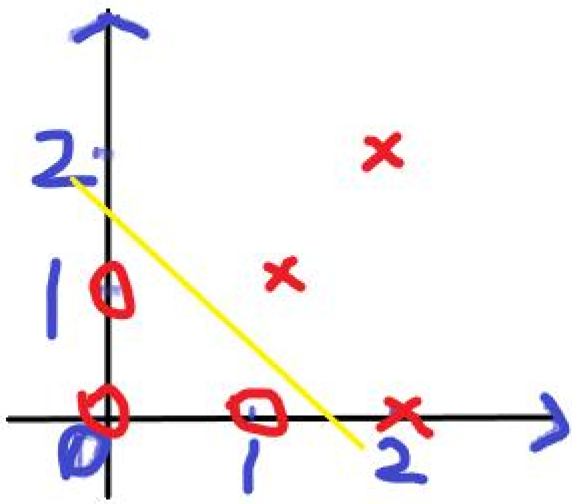
```
for i=1:iterNum
    theta = theta - 0.01 * grad;
    [cost, grad] = costFunction(theta, X, y);
    costList(i,1)= cost;
    disp(cost);
end;
disp(theta);
```

```
function g = sigmoid(z)
g = 1 ./ (1 + exp(-z));
end

function [J, grad] = costFunction(theta, X, y)
m = length(y); % number of training examples
J = 0;
grad = zeros(size(theta));
for i = 1:m
    J = J + ((-y(i)) * log(sigmoid(theta' * X(i,:)')) - (1-y(i)) * log(1-sigmoid(theta' * X(i,:)')));
    grad(1,1) = grad(1,1) + ((sigmoid(theta' * X(i,:)') - y(i)) * X(i,1)');
    grad(2:5,1) = grad(2:5,1) + ((sigmoid(theta' * X(i,:)') - y(i)) * X(i,2:5)' + theta(2:5,1));
end
J = J + 0.5 * sum(theta .* theta);
J = J / m;
grad = grad ./ m;
end
```

3.

(1)



超平面方程：X1 + X2 - 1.5 = 0;

(2)

如果这个数据点本身在 margin 之外“+”的那一侧，那么判决边界不受影响。

如果这个数据点在 margin 之内，或者在 margin 之外“-”的那一侧，那么这个点一定会成为新的支持向量。但是，判决边界并不一定发生变化，因为这个数据点可能能够被目标函数中的容错项处理掉。

由于新增了训练样本点，线性回归曲线需要根据新增的点重新拟合。所以线性回归必然会受影响。

(3)

支持向量：(1,0), (0,1), (2,0), (1,1)。

距离之和：$\sqrt{2}$ /2。

(4)

$$\mathcal{L}(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i\Big(y_i(w^Tx_i+b)-1\Big)$$

然后令

$$\theta(w)=\max_{\alpha_i\geq 0}\mathcal{L}(w,b,\alpha)$$

要求约束条件得到满足的情况下最小化 $\frac{1}{2}\|w\|^2$ ，实际上等价于直接最小化 $\theta(w)$ 。

目标函数变成了 $\min\limits_{w,b}\theta(w)=\min\limits_{w,b}\max\limits_{\alpha_i\geq 0}\mathcal{L}(w,b,\alpha)=p^*$ ，这里用 p^* 表示这个问题的最优值。

不妨把最小和最大的位置交换一下，变成： $\max\limits_{\alpha_i\geq 0}\min\limits_{w,b}\mathcal{L}(w,b,\alpha)=d^*$ ，交换以后的新问题是原始问题的对偶问题，这个新问题的最优值用 d^* 来表示。而且有

$d^* \leq p^*$，在满足某些条件的情况下，这两者相等，这个时候就可以通过求解对偶问题来间接地求解原始问题。

将 $\mathcal{L}(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i\Big(y_i(w^Tx_i+b)-1\Big)$ 化简后得到

$$\begin{aligned}\mathcal{L}(w,b,\alpha)&=\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_jy_iy_jx_i^Tx_j-\sum_{i,j=1}^{n}\alpha_i\alpha_jy_iy_jx_i^Tx_j-b\sum_{i=1}^{n}\alpha_iy_i+\sum_{i=1}^{n}\alpha_i\\&=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_jy_iy_jx_i^Tx_j\end{aligned}$$

求对 α 的极大即：

$$\begin{aligned}&\max_{\alpha}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_jy_iy_jx_i^Tx_j\\&s.t.,\ \alpha_i\geq 0, i=1,\ldots,n\\&\sum_{i=1}^{n}\alpha_iy_i=0\end{aligned}$$

根据 $w=\sum_{i=1}^{m}\alpha_iy^{(i)}x^{(i)}.$，$b^*=-\frac{\max_{i:y^{(i)}=-1}w^{*T}x^{(i)}+\min_{i:y^{(i)}=1}w^{*T}x^{(i)}}{2}.$，即可求出 b，w，最终得出分离超平面和分类决策函数。