Assignment 2

Computational Intelligence, SS2020

Team Members		
Last name	First name	Matriculation Number
Blöcher	Christian	01573246
Bürgener	Max	01531577

1 Linear regression

1.1 Derivation of Regularized Linear Regression

- Why is the design matrix $\boldsymbol{X}$ containing $n+1$ and not just n?
  - The simplest form of linear regression is calculated by:

$$y(\boldsymbol{x}, \boldsymbol{\omega}) = \underbrace{\omega_0}_{bias} + \boldsymbol{\omega}^T \boldsymbol{w}$$

  - The additional column is used to compensate the sum of the bias ω_0 and thus to simplify our calculation.

$$\boldsymbol{X} = \begin{bmatrix} 1 & \dots & x_1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & x_N \end{bmatrix} \begin{bmatrix} \omega_0 \\ \vdots \\ \omega_N \end{bmatrix}$$

- What is the dimension of the gradient vector?
  - The gradient contains the derivation of all $J(\boldsymbol{\theta})$ for all variables. That means the gradient has the dimension $m \times 1$ and it contains a group of targets.

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_m} \end{bmatrix}$$

- What is the definition of the Jacobian matrix and what is the difference between the gradient and the Jacobian matrix?
  - The Jacobian matrix elements are calculated by the derivatives of all outputs of a vector with respect to all parameters.

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

  - The gradient is computed from a scalar. It is a vector which contains the derivatives of a function for all parameters.

- What is the dimension of the Jacobian matrix and what is it equal to?
  - The dimension of the Jacobian matrix equals the dimension of the Design matrix

- Minimization of the regularized linear regression cost function

$$\begin{aligned} J(\boldsymbol{\theta}) &= \tfrac{1}{m} \|\boldsymbol{X\theta} - \boldsymbol{y}\|^2 + \tfrac{\lambda}{m} \|\boldsymbol{\theta}^2\| \\ J(\boldsymbol{\theta}) &= \tfrac{1}{m} \left[(\boldsymbol{X\theta} - \boldsymbol{y})^T \cdot (\boldsymbol{X\theta} - \boldsymbol{y}) \right] + \tfrac{\lambda}{m} (\boldsymbol{\theta^T \theta}) \quad | \, \tfrac{\partial}{\partial \boldsymbol{\theta}} \end{aligned}$$

Using Hint 2 from our exercise sheet the calculation is simplified to:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \tfrac{2}{m} (\boldsymbol{X\theta} - \boldsymbol{y})^T \cdot \boldsymbol{X} + \tfrac{2\lambda}{m} \, \boldsymbol{\theta}^T \quad |\stackrel{!}{=} 0$$

We set the gradient to zero to calculate our solution θ

$$
\begin{aligned}
\tfrac{2}{m}(\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{y}^T\boldsymbol{X}) + \tfrac{2\lambda}{m}\boldsymbol{\theta}^T &= 0 \\
\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{\theta}^T &= \boldsymbol{y}^T\boldsymbol{X} \\
\boldsymbol{\theta}^T(\boldsymbol{X}^T\boldsymbol{X} + \lambda E) &= \boldsymbol{y}^T\boldsymbol{X} \\
\boldsymbol{\theta}^T &= (\boldsymbol{y}^T\boldsymbol{X})(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{E})^{-1} \quad | \; ()^T
\end{aligned}
$$

Due to the property of transpose: $(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T$ we get:

$$
\begin{aligned}
\theta &= (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{E})^{-1^T}(\boldsymbol{y}^T\boldsymbol{X})^T \\
\theta &= (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{E})^{-1}(\boldsymbol{X}^T\boldsymbol{y})
\end{aligned}
$$

1.2 Linear Regression with polynomial features

As can be seen in figure **??** the training cost decreases with increasing polynomial degree n. That means the lowest cost on the training set can be achieved by using polynomials up to the highest degree $n = 30$ (s. figure **??**), but then the testing cost is maximised. This is due to overfitting: By finding parameters that suit the training data best, the solution becomes too specific for the validation and testing sets, leading to greater errors. Looking at figure **??** all training data points are in the close vicinity of the polynomial but the output function barely resembles the data. The best results can be obtained with degree $n = 13$, which minimises the validation cost (s. figure **??**) and leads to a very low testing cost. Having a validation set in addition to the training set helps in finding the right degree for the linear regression process and greatly improves the quality of the solution.

1.3 Linear Regression with polynomial features

1.4 (Bonus) Linear Regression with radial basis functions

The results of the Linear Regression approach with radial basis functions are similar to the ones with polynomial functions: Increasing the degree/number of kernels l leads to lower training cost (s. figure **??**) - minimised for $l = 40$ (s. figure **??**) - but choosing l too high results in overfitting (s. figure **??**). The cost function of the validation set is minimised for degree $l = 9$ (s. figure **??**), leading to a low but not quite minimised testing cost. The actual testing cost minimum is found with degree $l = 10$. Improving on the results of Linear Regression with polynomial basis functions it is about 20% lower than the testing cost minimum with (higher) degree $n = 13$ (s. figures **??** and **??**).

2 Logistic Regression

2.1 Derivation of Gradient

$$J(\boldsymbol{\theta}) \quad = \quad -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\ \log(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) + (1-y^{(i)})\ \log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))\right) \quad \mid \frac{\partial}{\partial \boldsymbol{\theta}_j}$$

Derivation of the hypothesis function $h_{\boldsymbol{\theta}}(\boldsymbol{x})$:

$$\frac{\partial h_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\sigma(\theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n) = x_j^{(i)}$$

Using the chain rule and the given hint for the derivation of the sigmoid function σ, we get:

$$\begin{aligned}
\frac{J(\boldsymbol{\theta})}{\partial \theta_j} \quad &= \quad -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\frac{1}{h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})}h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\left(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\right)\ x_j^{(i)} + \ldots\right. \\
&\qquad \left.\ldots(1-y^{(i)})\frac{1}{1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})}\left(-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\ (1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))\ x_j^{(i)}\right)\right) \\
\frac{J(\boldsymbol{\theta})}{\partial \theta_j} \quad &= \quad -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) + h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\right)x_j^{(i)} \\
\frac{J(\boldsymbol{\theta})}{\partial \theta_j} \quad &= \quad \frac{1}{m}\sum_{i=1}^{m}(y^{(i)} + h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\ x_j^{(i)}
\end{aligned}$$

2.2 Logistic Regression training with gradient descent

-
-
-
-