MGT-302 Homework 1

Alexei Ermochkine (sciper: 302791)

March 2022

1 Linear regression

1.1 Gradient descent

1.1.1 (a) Gradient of the cost function

$$\nabla J(\theta) = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)} (h_\theta(x^{(i)}) - y^{(i)})$$

1.1.2 (b),(c),(d) Gradient descent code

linreg.m trains the model on the training dataset and plots the cost function as a function of the number of iterations. It was found that $\alpha = 1$ is the step size that makes the method converge the fastest in this case.

The optimal value of theta is $\hat{\theta} = \begin{pmatrix} 0 & 0.3907 & 0.1926 & -0.1186 & 0.4997 & -0.6876 \end{pmatrix}$

linregtest.m tests the obtained values of θ on the test dataset. The objective function is $= 0.0035$ when using the model θ stated above.

1.2 Stochastic gradient descent

1.2.1 (a) SGD vs GD

In SGD, at any given iteration, instead of summing over all the samples in

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

We compute the right-most term for only one element of the sample set:

$$\theta_j = \theta_j + \alpha \frac{1}{m} (x_j^{(k)} (h_\theta(x^{(k)}) - y^{(k)}))$$

where $k = floor(199 * rand(1) + 1)$ a randomly chosen index between 1 and 200.

linreg_rand.m contains the SGD algorithm.

The SGD algorithm converges way slower than normal GD. For a constant $\alpha = 1$, it takes 10^6 iterations to get to $\approx \hat{\theta}$ with errors in the 10^{-3} range (very approximate observation since for each execution the final θ is different).

1.2.2 (b) SGD with adaptive step size

For an adaptive step size $\alpha_{iter} = \frac{b}{1+iter}$ ($b = 1000$ being the optimal parameter after tuning), we observe a better convergence with errors compared to $\hat{\theta}$ reduced to the 10^{-4} range for 10^6 iterations.

2 K-means clustering

The K-means algorithm is implemented in **kmean.m**. Please note that the script is in **two sections**, the first one executes K-means for one particular K and plots the result, the second generates the fig.2 plot and is very slow.

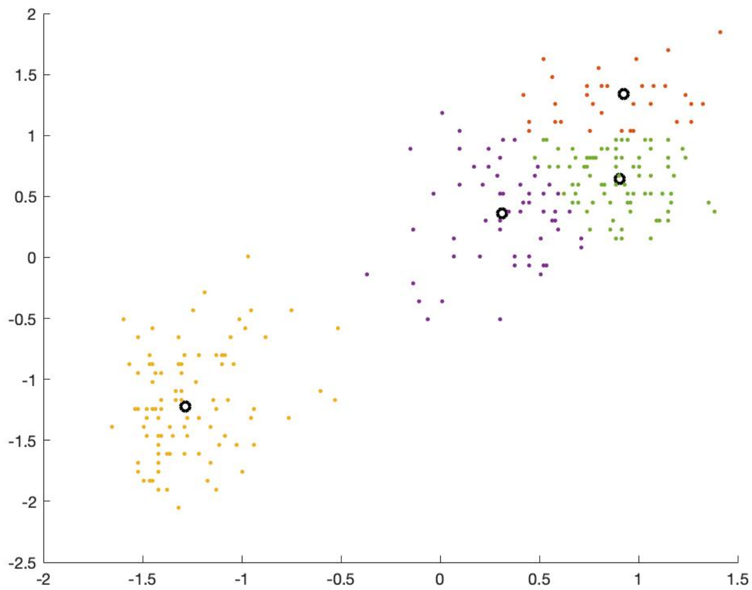

Figure 1: Output of script for K=4

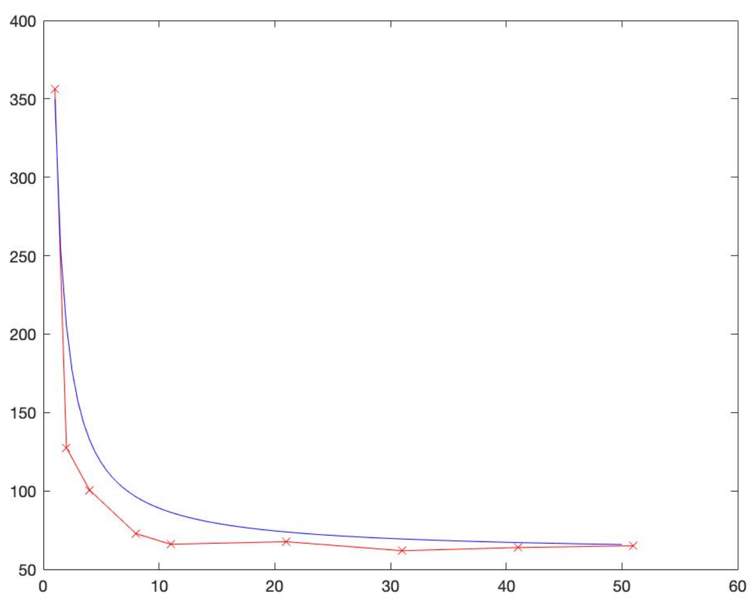

Figure 2: Cost function for increasing values of K (in red) and an empirical function modeling the behavior (in blue)

3 Support Vector Machine (SVM)

3.1 Boundaries for different C values

The script **SVMscrip.m** does just that.

3.2 optimal value of C

C=1 seems to work best for this case.