**Linnaeus University**
Introduction to Machine learning, 2DV516
*Jonas Nordqvist*
`jonas.nordqvist@lnu.se`

# Assignment 2: Linear and logistic regression

## Introduction

In this Assignment you will use MATLAB to handle a number of exercises related to gradient descent, linear regression, and logistic regression. The datasets used in the assignment can be downloaded in Moodle. The exercises are further divided into lectures. That is, for example, the first set of exercises related to regression are suitable to handle after Lecture 3.

## Submission instructions

All exercises are individual. We expect you to submit one m-file for each exercise and provide all functions (as local functions, scripts or classes) used in the exercise[1].

Most exercises can be handled with a single m-file. Certain quantitive questions such as: *What is the expected number of stories in a 900 ft building?*, can simply be handled as a print statement in the m-file. More qualitative questions such as: *Motivate your choice of model.*, should be handled in a text file. (All such answers can be grouped into a single text-file)

Finally, keep all your m-, mat-, csv-, and text-files in a single folder named as `username_A2` and submit a zipped version of this folder.

## Linear and polynomial regression (Lecture 3)

### Exercise 1: Linear regression

In this exercise you will be working with the data in `data_build_stories.mat`. The data consists of 60 training example of buildings, and the only feature in this dataset is *height* measured in ft. The label we want to predict is the number of stories in the buildings. The main objective is to find the hypothesis $f(X) = \beta_1 + \beta_2 X = \texttt{intercept} + \texttt{height} \times X$, which estimates the linear relation between the number of stories in a building and its height.

1. In order to better understand the data it is often a good idea, if possible, to visualize it. Plot the data in the matrices $X$ and $y$ in a scatter plot.

2. Extend $X$ by adding a one column and compute $\beta$ using the normal equation $\beta = (X^T X)^{-1} X^T y$. What is the predicted number of stories for a building of height 900 ft?

3. Implement the cost function $J(\beta)$ for linear regression, which takes $X, y, \beta$ as input and outputs the *mean squared error* (MSE), i.e.

$$J(\beta) = \text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(\beta_0 + \beta_1 x_i - y_i)^2.$$

[1] If you are using local functions, this might lead to multiple copies of certain functions scattered across several m-files. That is okay.

What is the cost when using the $\beta$ computed by the normal equation above?

4. Gradient descent $\beta^{j+1} = \beta^j - \alpha X^T(X\beta^j - y)$ is a numerical method to approximate $\beta$ that uses two hyper parameters: The learning rate (or step length) $\alpha$ and the number of iterations $N$ needed to stabilize $J(\beta)$. You should compute $\beta$ using two versions of gradient descent.

   (a) Without any feature normalization.
   (b) With feature normalization $(X - \mu)/\sigma$.

   In both cases, what hyperparameters $\alpha, N$ are needed to get within 1% of the final cost for the normal equation? What is the predicted number of stories for a building of height 900 ft?

5. Finally, produce a plot containing a scatter plot $X, y$ and the three linear fits computed above. You can do it three separate plots (easy) or try to squeeze in all cases into a single plot.

## Exercise 2: Multivariate regression

In this exercise you will use the data set in `GPUBenchmark.csv`. The first six columns of $X$ are various data related to graphic cards. For example, number of cores and clock speed. See the excel file `NvidiaGPU_Speed.xlsx` for more details. The 7th column is the response $y$. In this case the result of testing the graphic card in a certain benchmarking program. The dataset contains 18 observations. The objective is to fit a linear estimation of the features and its outcome.

1. Multivariate datasets are hard to visualize. However, to get a basic understanding it might be a good idea to produce a plot $X_i$ vs $y$ for each one of the features. Use `subplot(2,3,i)` to fit all six plots into a single figure.

2. Extend $X$ by adding a one column and compute $\beta$ using the normal equation $\beta = (X^TX)^{-1}X^Ty$. What is the predicted benchmark result for a graphic card with the following feature values?

   $$2432,\ 1607,\ 1683,\ 8,\ 8,\ 256$$

   The actual benchmark result is 114. What is the error?

3. If your version of gradient descent and cost function is not adapted for multivariate linear regression this is the time to do so. Once adapted ...

   (a) What is the cost when using the $\beta$ computed by the normal equation above?
   (b) What gradient descent hyperparameters $\alpha, N$ are needed to get within 1% of the final cost for the normal equation? We strongly recommend you to apply feature normalization before running gradient descent.
   (c) What is the predicted benchmark result (and error) for the example graphic card presented above?

## Exercise 3: Polynomial regression

The data for this exercise is found in `housing_price_index.mat`. The data consists of housing price index for houses in Småland from the year 1975 to 2017. The objective is to predict the pricing index using only the year.

1. Plot the data in the matrix `housing_price_index`.

2. As you probably notice the relationship among the variables doesn't seem to be linear. Try to fit (and plot using `subplot(9,3,i)`) all polynomial models $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_d X^d$ for degrees $d \in [1, 9]$. Which polynomial degree do you think gives the best fit? Motivate your answer!

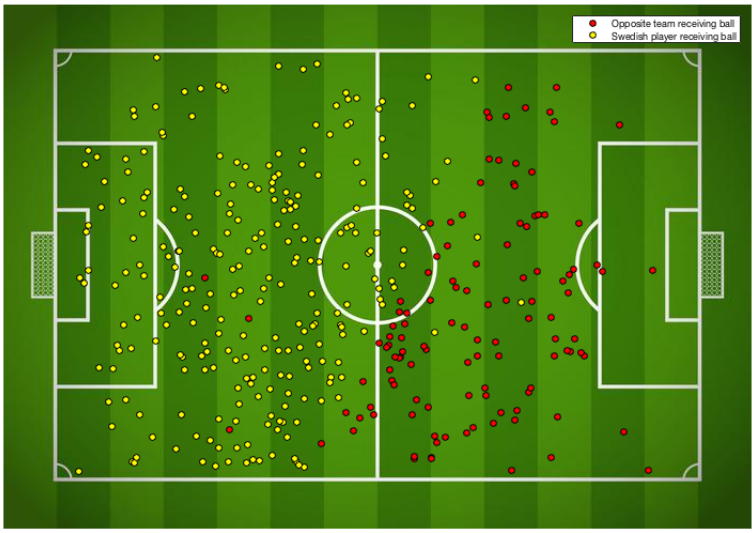
3. Jonas Nordqvist bought in 2015 a house in Växjö for 2.3 million SEK. What can he expect to get, using your "best fit model", for his house when he (after completing his PhD) sells his house in 2022 (to start his new career as a data analyst in Stockholm)? Is your answer realistic?

# Logistic regression (Lecture 4)

## Exercise 4: The clever football coach

In this exercise you will use logistic regression to help the Swedish football team. The goalkeeper coach of the team have studied the goalkeepers kicks and kept track of when the Swedish players are at the receiving end of the kicks, and when the other team are the receivers. Your objective is to use logistic regression to model where the goalkeeper should kick the ball in order to increase the chance that the receiving player is from the Swedish team.

The data consists of a data matrix `football_X` with the coordinates of the players receiving the balls of the goalkeeper's kicks, and a vector of binary labels `football_y` where 1 means that the player was Swedish and 0 that the player was from the opposing team. Provided in this Assignment is an m-file, `plotFootball.m`. Run the command `plotFootball(football_X, football_y)`, and you will see the image below.




1. Logistic regression requires several functions, of which the first we should implement is the sigmoid function. The function should take any matrix as input and output a matrix of the same size where you apply the sigmoid function on each element. Implement the function in MATLAB and test your function using `sigmoid([0 1; 2 3])`. You should then expect the following output

$$\texttt{ans} = \begin{pmatrix} 0.5000 & 0.7311 \\ 0.8808 & 0.9526 \end{pmatrix}.$$

For large values of $z$, sigmoid(z) is close to one. What is the smallest value of $z$ such that MATLAB interprets sigmoid$(z) - 1$ as zero? This test can be performed by the logical operation `== 0`, e.g. `(sigmoid(4)-1)==0` will output `FALSE`.

2. Implement the cost function for logistic regression. Testing it for the football dataset using $\beta = 0$ should yield `cost = 0.6931`.

3. Normalize data ($\mu$, $\sigma$) and implement gradient descent for logistic regression.

4. Train a model on the provided dataset using gradient descent. Print the hyperparameters $\alpha$ and $N_{iter}$ (number of iterations) and plot the cost function $J(\beta)$ as a function over iterations. **Hint**: Skip the first 10-20 iterations in the plot to better capture details for larger iterations.

5. Once finding the optimal $\beta$ you can plot the data and the decision boundary using

   `plotFootball(football_X, football_y, beta, mu, sigma).`

6. What is the training error (number of non-correct classifications) and the training accuracy (percentage of correct classifications) for your model?

## Exercise 5: Multivariate Logistic Regression

You will now try to classify woman breast cancer tumours. This dataset `breast-cancer.mat` contains 683 observations and has 9 features and (in column 10) binary labels of either benign (2) or malignant (4). A full description of the dataset is available is this note[2] (note that the ID has been removed from the original data).

1. Read data and shuffle the rows in the raw data matrix:

```
data = load('breast-cancer.mat')
data = data.breast_cancer;
data = data(randperm(size(data,1)),:); % Shuffle rows
```

2. Replace the responses 2 and 4 with 0 and 1 (why?) and divide the dataset into a training set and a test set. How many observations did you allocated for testing, and why this number?

3. Normalize the training data and train a *linear logistic regression model* using gradient descent. Print the hyperparameters $\alpha$ and $N_{iter}$ and plot the cost function $J(\beta)$ as a function over iterations.

4. What is the training error (number of non-correct classifications) and the training accuracy (percentage of correct classifications) for your model?

5. What is the test error and the test accuracy for your model?

6. Repeated runs will (due to the shuffling) give different results. Are they qualitatively the same? Do they depend on how many observations you put aside for testing? Is the difference between training and testing expected?

## Exercise 6: Nonlinear logistic regression

For this exercise you will work with the microchip dataset in `microchiptests.csv`. It contains 118 observations $(X_1, X_2, y)$ where $X_1$ and $X_2$ (columns 1 and 2) are two measurements used to evaluate the construction quality of a microchip and $y$ (column 3) indicates whether the microchip later on was found to be OK (1) or flawed (0).

[2] https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.names

1. Plot the data in $X$ and $y$ using different symbols or colors for the two different classes. Notice also that $X_1$ and $X_2$ are already normalized. Hence, no need for normalization in this exercise

2. Since we are dealing with non-linear data we will use a polynomial expression of the features in order to find a good model. The first step is to implement a feature mapping function. That is a function that takes two features $X_1$, $X_2$ and a degree $d$ as input and outputs all combinations of polynomial terms of degree less than $d$ of the variables $X_1$ and $X_2$. A suggestion on the main part of the code is given below in the compendium in the section *Non-linear boundaries*.

3. Use gradient descent to find $\beta$ in the case of a quadratic model.

$$X\beta = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_1 X_2 + \beta_5 X_2^2.$$

   Print the hyper parameters $\alpha$ and $N_{iter}$, print the final $\beta$, and plot the corresponding decision boundary (together with the $X, y$ scatter plot).

4. Repeat 3. but this time using the predefined optimization method `fminunc` that comes with MATLAB. The results should be similar.

5. Finally, use `fminunc` to construct nine different classifiers, one for each of the degrees $d \in [1, 9]$, and produce a figure containing a $3 \times 3$ pattern of subplots showing the corresponding decision boundaries. Also, for each case, print the number of training errors.

Which of the models plotted in 6.5 is the best? Higher order polynomials indicate clear cases of overfitting. How to properly identify the best possible fit, and how to improve logistic regression using an approach called *regularization* will be the topic for the next lecture where we will revisit the microchip dataset.

## Exercise 7: Image Recognition (VG Exercise)

*This exercise is optional for passing the assignment, but required in order to obtain higher grades (A-B).*

In this final exercise, we will use logistic regression for classifying handwritten digits. Input to the algorithm is an image with a handwritten digit (0-9) and the output should be a classification 0-9.

The data and a description of it is available at http://yann.lecun.com/exdb/mnist/. This is not formatted for MATLAB in particular, and have to be imported.

The object is to train a logistic regression classifier to perform as good as possible on recognizing handwritten images. Describe your effort and what you found out to be the best setting to lower the test error. Note that the computations might be (very) heavy, so start of by a smaller subset rather than using the entire dataset. Preferably, the final model should be trained using the full training set. However, this requires some time to train, so a subset of the training data is also okey if you have trouble running the full training set.

The description of this exercise is deliberately vague as you are supposed to, on your own, find a suitable way to solve this problem in detail. This is why it is important that you document your effort and progress in your report.

In the file `mnist.m` you will find some hints on getting started.

Your output from this assignment should be the test error for your best model and a description of your work procedure. In the next assignment we will try to solve this problem using *Support vector machines*, and see if we can obtain a better result.

## Accuracy, regularization and model selection (Lecture 5)

In the following exercises 8 and 10 you are welcome to use the built-in library in MATLAB. The command `mdl = fitlm(X,y)` returns linear regression model of the pair $(X, y)$[3] as a structure array `mdl`. You can access its properties by for instance `mdl.Coefficients`.

### Exercise 8: Forward selection

In this exercise we will go back to the dataset used in Exercise 2, `GPUbenchmark.csv`. On this dataset we will use *forward selection* in order to obtain a model of (eventually) fewer features. One part of this exercise is to implement $k$-fold cross validation.

1. Implement $k$-fold cross-validation as described in the lecture.
2. Implement the forward selection algorithm as described in the slides of Lecture 5. In the loop use the training MSE to find the best model in each iteration. The algorithm makes $p+1$ models $\mathcal{M}_0, \ldots, \mathcal{M}_p$, where $\mathcal{M}_i$ is the best model using $i$ features. Among these $p+1$ models select the best using $k$-fold cross-validation.
3. Apply your forward selection on the `GPUbenchmark.csv`. Use 3-fold cross-validation to find the best model among all $\mathcal{M}_i$, $i = 1, \ldots, 6$. Which is the best model? Which is the most important feature? Was this expected?
4. Instead of using the cross-validation to choose the best model, use AIC or AICc. Are the results the same? Is the same model still the best?

### Exercise 9: Regularization

In this exercise we will learn about regularization. We will adjust our cost function and gradient descent to include the ridge regularization term. We will use the same dataset as in Exercise 6 `microchipstests.csv`.

1. Implement the new functions `costLogisticReg` and `gradientDescentReg`, according to the description in the Complementary material. The cost at $\beta = 0$ should be `cost = 0.1923`.
2. Perform gradient descent in the same way as you just did in the Exercise 6 for the degrees $d \in [1, 9]$, but now with the regularization term included. Use the regularization parameter $\lambda = 1$. Also, plot the corresponding decision boundaries.
3. Make a plot on training error for all the models 1-9 versus their degree. In the same graph also include the results of the unregularized models made in Exercise 6. What can we read from the graphs?
4. Perform cross-validation on all 9 unregularized models and the 9 regularized models and plot the validation error versus degree for both types in the same plot. What is your conclusion?
5. Try a (much) larger value for $\lambda$ and study the decision boundary for the different classifiers. What happens?

[3] Note that you should not include the column of ones for $X$.

### Exercise 10: Coefficient accuracy

In this exercise we will estimate the accuracy in our coefficients from the dataset used in Exercise 1, `data_build_stories.mat`.

1. One evidence that would strengthen our choice of model is if the residuals (the errors) of the model seem approximately random. Plot the errors of the height and stories data. What can you tell from the plot? What is the mean of the error vector?

2. Compute the correlation $r$ between $X$ and $Y$. What does the correlation tell us? Is the relation strong? Is this inline with your answer in the previous question?

3. Besides providing the model `fitlm` also provides the standard errors of the estimated coefficients, in the table `Coefficients`. Using the standard error you can compute the confidence interval for the estimates. Compute the confidence intervals for the intercept. Does the interval include 0? Is the interval for the intercept realistic?

4. Compute the confidence interval for the slope. Is there at least a 95% probability that there is a (positive) relationship between $X$ and $Y$?