12th December, 2018

Advanced Digital Signal Processing: MATLAB Laboratory

Basics of Machine Learning: Linear and Logistic Regression

Nowadays, Machine Learning (ML) is one of the most powerful and widespread techniques adopted by both industry and academia for data analysis and estimation. The main novelty of ML is that differently from the classical statistics approach, whose goal is to estimate the desired parameters given some, possibly noisy, observation (or data), ML engineers have the goal to teach a machine to do this job for them. ML is able to learn from a big amount of data (referred to as training data) some particular pattern within the data, which can be used to build mathematical models able to make predictions on new set of data (test data). Typical ML applications are image/sound recognition, e-mail spam filters, text/speech recognition, anomaly detection, feature extraction, recommendation systems, etc. This laboratory will be focused on classification, mainly binary.

In ML, the overall available data is usually split into two parts: i) the training data, which is used to learn the parameters of the model, and the test data, which is used to test the ability of the learnt model to generalize (or predict) new samples generated from the same distribution. As a first step for all the following exercises, it is highly suggested to get familiar with the datasets by carefully plotting/computing some specific metrics (e.g., histogram, 2-D plot, correlation matrix of the data etc.).

Exercise 1: Polynomial Regression and Model Selection

Polynomial regression is one of the simplest ML algorithms (see text book). In a polynomial regression model of degree d, the ith output (or label) $y^{(i)}$ is estimated by the ith input (or feature) $x^{(i)}$ as

$$\hat{y}^{(i)} = \boldsymbol{x}^{(i)T}\boldsymbol{\theta},$$

where $\boldsymbol{\theta} = [\theta_0, \theta_1, ..., \theta_d]^T$ are the model parameters to be learned during the training phase and $\mathbf{x}^{(i)} = [1, x^{(i)}, x^{(i)2}, ..., x^{(i)d}]^T$ is the feature vector. Given m available data points, the m labels can be thus estimated as

$$\begin{bmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(m)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^{(1)T} \\ \boldsymbol{x}^{(2)T} \\ \vdots \\ \boldsymbol{x}^{(m)T} \end{bmatrix} \boldsymbol{\theta}.$$

The overall estimation error can be thus computed as

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - \hat{y}^{(i)}\right)^2.$$

Dataset Description

The file 'Ex1_PolyReg.mat' contains the training set $(\mathbf{x}_{\text{train}}, \mathbf{y}_{\text{train}})$, which contains 60 examples to be used to estimate the parameters model $\boldsymbol{\theta}$, and the test set $(\mathbf{x}_{\text{test}}, \mathbf{y}_{\text{test}})$, which contains 40 examples to be used for testing the learning capability of the model. The data describes the amount money $\mathbf{y}$ (expressed in thousands of dollars per post, k$/post) that can be earned on a social network (e.g., Instagram) depending on the number of followers $\mathbf{x}$ (expressed in millions of followers).

Exercise Goal

The goal of this exercise is:

- to train a polynomial regression model by learning the parameters vector $\boldsymbol{\theta}$ given the training set $(\mathbf{x}_{\text{train}}, \mathbf{y}_{\text{train}})$, made by a number of training points $(x^{(i)}_{\text{train}}, y^{(i)}_{\text{train}})$, for different polynomial degrees $d = 1 : 11$;
- to show the ability of these polynomial regression models to generalize to the test set $(\mathbf{x}_{\text{test}}, \mathbf{y}_{\text{test}})$, made by a number of test points $(x^{(i)}_{\text{test}}, y^{(i)}_{\text{test}})$, by computing the error $J(\boldsymbol{\theta})$ for both training and data set.

In particular, it is required to:

1. plot the training error $J_{\text{train}}(\boldsymbol{\theta})$ and the test error $J_{\text{test}}(\boldsymbol{\theta})$ versus polynomial degree d;
2. plot training points, test points and the polynomial models for $d = 1, 3, 5, 11$.

Exercise 2: Logistic Regression for Binary Classification - Linear Decision Boundary

Logistic regression is the most common ML algorithm used for the case in which the labels $y^{(i)}$ can assume only a discrete set of values. Such task is also know as classification. Given an n-feature vector $\boldsymbol{x}^{(i)} = [1, x^{(i)}_1, x^{(i)}_2, ..., x^{(i)}_n]^T$ (please note that $x^{(i)}_0 = 1$ is the intercept), the goal of this exercise is to train a logistic regression model for binary classification, i.e., the labels can be either $y^{(i)} = 0$ or $y^{(i)} = 1$.

In logistic regression, the ith label $y^{(i)}$ is estimated by the ith input feature vector $\boldsymbol{x}^{(i)}$ as

$$\hat{y}^{(i)} = g\left(\boldsymbol{x}^{(i)T}\boldsymbol{\theta}\right),$$

where $\boldsymbol{\theta} = [\theta_0, \theta_1, ..., \theta_n]^T$ are the model parameters to be estimated, and

$$g(z) = \frac{1}{1 + \exp(-z)}$$

is the sigmoid function. The overall estimation error for logistic regression can be computed as

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(\hat{y}^{(i)}\right) - \left(1 - y^{(i)}\right)\log\left(1 - \hat{y}^{(i)}\right)\right].$$

In case of logistic regression, it is not possible to estimate the parameters vector $\boldsymbol{\theta}$ in closed-form, as for linear regression. The above error function has to be iteratively minimized by gradient descent algorithm. The gradient descent update rule for the kth parameter θ_k is thus

$$\theta_k(t+1) = \theta_k(t) - \mu\frac{\partial}{\partial\theta_k}J(\boldsymbol{\theta}),$$

where μ is the gradient descent step-size. It is easy to show that the gradient for logistic regression simplifies to

$$\frac{\partial}{\partial\theta_k}J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - y^{(i)}\right)x^{(i)}_k.$$

Dataset Description & Exercise Goal

The file 'Ex2_LogReg.mat' contains the feature matrix $\mathbf{X}$, which is a 100×2 matrix containing the 2 features $(x^{(i)}_1, x^{(i)}_2)$ for 100 examples, and the corresponding binary labels $y^{(i)}$, with $i = 1, 2, ..., 100$. The data describes a set of 100 students that have passed $(y^{(i)} = 1)$, or not failed $(y^{(i)} = 0)$ a specific exam depending on the percentage $x^{(i)}_1$ of lessons attended and the amount of hours/week $x^{(i)}_2$ spent studying at home.

Exercise Goal

The goal of this exercise is:

1. to implement logistic regression by gradient descent in order to estimate the parameters vector $\boldsymbol{\theta}$, given the feature matrix $\mathbf{X}$ and the corresponding labels vector $\mathbf{y}$;
2. plot the logistic regression error $J(\boldsymbol{\theta})$ versus the number of gradient descent iterations;
3. plot the given points and the estimated decision boundary.

Hint: recall that a very important pre-processing step for logistic regression (especially when implemented by gradient descent) is to normalize the features in order to work with features with zero-mean and unit-standard deviation.

Exercise 3: Logistic Regression for Binary Classification - Non-Linear Decision Boundary

Repeat the above exercise for the dataset 'Ex3_LogReg.mat', which contains the feature matrix $\mathbf{X}$, which is a 100×2 matrix containing the 2 features $(x_1^{(i)}, x_2^{(i)})$ for 100 examples, and the corresponding binary labels $y^{(i)}$, with $i = 1, 2, ..., 100$. The dataset represents the outcome of a compliance test (oversimplified for the sake of the exercise) for 118 car wheels ($y^{(i)} = 1$ compliant, $y^{(i)} = 0$ non-compliant) depending on their diameter $x_1^{(i)}$ and width $x_2^{(i)}$, expressed in *cm*. After visualizing the data, you will notice the particular shape of the data, which may require to combine the two features in order to get higher order decision boundary (e.g., elliptical decision boundary).

Hint: Given two features x_1 and x_2, a possibility would be to generate a new set of features containing all the possible feature combinations up to a certain degree d as: $1, x_1, x_2, x_1^2, x_2^2, x_1x_2, x_1^3, x_2^3, x_1^2x_2, ..., x_1^d, x_2^d$.

Exercise 4: Logistic Regression for Instrument Source Recognition

Given a dataset containing the feature related to a number of music recordings, the goal of this exercises is to predict whether a particular sound has been generated by an acoustic, electric or synthetic instrument. For this exercises, it is used the NSynth dataset (information can be found at https://magenta.tensorflow.org/datasets/nsynth). As a first step, the dataset has been simplified by considering only two classes (acoustic and synthetic), and by excluding "categorical" features from the dataset, which can be imported as a MATLAB '.mat' array. The second part of the exercise considers the whole dataset that needs to imported from the given '.cvs' files. The datasets given for the exercise only includes the features extracted from the recordings. The files 'Ex4_LogReg_Instrument_Binary.mat', 'training.csv' and 'test.csv' contain the labels and features (i.e., in form of MATLAB array or text files) extracted from the recordings. The folder "Audio_recording_samples" contains some samples of the audio files used to generate the dataset.

a) Binary Instrument Source Recognition

The goal of this exercise is to apply the logistic regression model implemented above to the more realistic dataset 'Ex4_LogReg_Instrument_Binary.mat'. This includes the training set $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$, containing 8260 examples to be used to estimate the parameters model $\boldsymbol{\theta}$, and the test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$, which contains 2724 examples to be used for testing the learning capability of the model. The dataset is the NSynth dataset (information can be found at https://magenta.tensorflow.org/datasets/nsynth), that, for this part a) of the exercise has been simplified, pre-processed and stored in a MATLAB array '.mat'. The dataset contains a number of recordings of musical notes played by different instruments. Given these recordings, some features have been extracted. In particular, the simplified dataset $\mathbf{X}$ contains the following features (see https://magenta.tensorflow.org/datasets/nsynth for a more detailed description of the features):

- $x_1^{(i)}$: "instrument", a unique sequential identifier for the instrument the note was synthesized from;
- $x_2^{(i)}$: "instrument family", the index of the instrument family this instrument is a member of (e.g., bass, flute, guitar, etc.);

- $x_3^{(i)}$: "note", a unique integer identifier for the note;
- $x_4^{(i)}$: "pitch", the 0-based MIDI pitch in the range [0, 127];
- $x_5^{(i)}$: "velocity", the 0-based MIDI velocity in the range [0, 127].

The vector $\mathbf{y}$ contains the labels associated to the previous features. In particular, all the recordings described by the simplified dataset above can be classified into two classes, acoustic ($y^{(i)} = 0$) and synthetic ($y^{(i)} = 1$) instruments.

Exercise Goal

The goal of the exercise is to train a logistic regression model given the training set $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$, able to predict whether the sound recordings contained into the test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$ correspond to an acoustic or synthetic instrument. After predicting the instrument music sources of the test set $\hat{\mathbf{y}}_{\text{test}}$, compare them with the true labels $\mathbf{y}_{\text{test}}$ by computing the accuracy of the estimation (percentage of recordings correctly estimated).

b) Multi-Class Instrument Source Recognition

This exercise is similar to the previous one, with the difference that now the true NSynth dataset is provided into the '.csv' files 'training.csv' and 'test.csv'. These files can be loaded in MATLAB by double-clicking on the name file through the MATLAB "current folder" window. The first step is to correctly build the feature matrix $\mathbf{X}$ by importing the features from the '.cvs' files into MATLAB (pay attention to the categorical feature 'qualities', which was excluded in the previous exercise for simplicity). Next, the labels vector $\mathbf{y}$ needs to be built, which must contain the values of the "instrument-source' column of the '.csv' files. Notice that now the recordings are classified into 3 classes: acoustic ($y^{(i)} = 0$), electric ($y^{(i)} = 1$) and synthetic ($y^{(i)} = 2$) instruments.

Exercise Goal

The goal of the exercise is to train a logistic regression model given the training set $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$, able to predict whether the sound recordings contained into the test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$ correspond to an acoustic, electric or synthetic instrument. After predicting the instrument music sources of the test set $\hat{\mathbf{y}}_{\text{test}}$, compare them with the true labels $\mathbf{y}_{\text{test}}$ by computing the accuracy of the estimation (percentage of recordings correctly estimated).