MO444 - Pattern Recognition and Machine Learning
Practical Assignment #1

Osvaldo Xavier Dias Junior*
Prof. Anderson Rocha†

Abstract

This report is regarding the first assigment of the module MO444 (Pattern Recognition and Machine Learning), and it presents results of linear regression models, gradient descent and normal equations algorithms applied to the dataset "Dataset of popularity of Social Nets". The models were modified throughout the experiments to verify the dependence of performace related to feature scaling, regularization, increase of complexity and remolval of discrete features. The predictions obtained for all the models were not good, because of the complexity of the proposed problem and by the fact that probably a linear regression it is not the fittest model to work in this dataset. However, some techniques applied in the pre processment of data and in the model enhanced the result.

1. Introduction

The linear regression is a mathematical model that corresponds to a linear relationship between input variables and output variables, as follows:

$$h = \theta^{\intercal} X \tag{1}$$

Where X is the vector of the inputs, θ is the vector of the parameters and h is the vector of predicted targets. This model can be transcripted to an algorithm to do continuos value prediction. Once you have a dataset of inputs and outputs, it is possible to use an method of minimization called gradient descent, in order to find the vector θ.
Gradient descent of a linear regression model is an iterative method of minimization that corresponds to the minimization of a cost function, which is going to be defined as:

$$J(\theta) = \frac{1}{2m} \sum_{m=1}^{m} (h - Y)^2 \tag{2}$$

Where m is the number of examples in the dataset. To reach the goal of find the values of θ and therefore apply gradient descent. It will be needed to perform the following calculations:

$$\theta^{new} = \theta_n^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_n} \tag{3}$$

Where α is the learning rate and n is the number of features. Using the previous equation it is possible to calculate the values of θ iteratively. Another option to find the vector θ is to use normal equations:

$$\theta = (X^{\intercal} X)^{-1} X^{\intercal} Y \tag{4}$$

Where Y are the given targets. Once the cost function is a convex function, this equation gives the optimal solution of the problem or global minima of the cost function.
In the experiment proposed in this assignment is mandatory to use the methods/algorithms shown before to predict the number of shares in a social network, based on the information of the datasets (e.g number of words in the title, number of images, number of words in the content, title subjectivity e etc.)

2. Proposed Solutions

The solutions proposed were models based on linear regression using gradient descent, the normal equation solution was also calculated for comparison. The features were processed in order to achieve a better result on gradient descent. It follows a list of techniques applied, in chrological sequence, to the dataset of the calculations of gradient descent.

1. Feature scaling (Processing Features)
2. Remove outliers' targets (Processing Features)
3. Remove discrete features (Processing Features)
4. Increase complexity (Model Modification)
5. Regularization (Model Modification)

*Is with the Institute of Computing, University of Campinas (Unicamp). **Contact**: `osvaldoxdjr@gmail.com`
†Is with the Institute of Computing, University of Campinas (Unicamp). **Contact**: `anderson.rocha@ic.unicamp.br`

3. Experiments and Discussion

First of all, to program the algorithms the programming language chosen was Python, inside an Ubuntu enviroment. The train dataset provided was divided into two: 80% for trainning and 20% for validation, this was done to avoid testing the proposed solutions in the test dataset, because if the model is tested every time on the test dataset it can drives the model to an overfited solution for the test dataset, which is not good beacuse the test dataset should work as real data of the problem, and thus must not be revealed until the model is fully validated.
To start the experiments it was selected all examples in the trainning dataset and ran the algorithm to calculate gradient descent and the normal equation, however the algorithm did not converge or present any valid value for θ and $J(\theta)$ so it was concluded that it would be necessary to perform a feature scaling to adjust the emphasis of the features, because there was a considerable variance among each feature.
It was chosen a min-max feature scaling:

$$X_n = \frac{X_n - max(X_n)}{max(X_n - min(X_n))} \tag{5}$$

This feature scaling was performed for each feature and the algorithm for gradient descent and normal equation ran, the graphical results are shown in Figure 1, $J(\theta)$ was calculated for the trainning dataset in this chart.

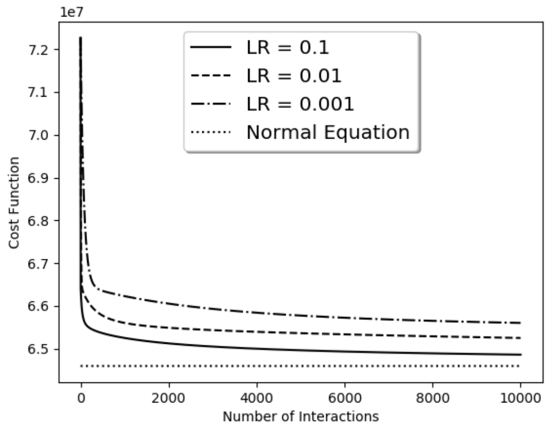

Figure 1. This figure shows the cost function vs number of iteractions for three differente learning rates $(0.1, 0.01, 0.001)$. It is also plotted a horizontal line to highlight the normal equation cost.

Despite the fact that the models are converging, the first analysis to be made is that the cost function (quadratic error) is high for all iteractions, this occurs due to: the complexity of the problem, the model is linear and simple and the trainning data is still raw for trainning. Another interesting aspect is the convergence, the models are converging for the different learning rates and as you decrease the learning rate, the models become slower in the task of reaching the optimal solution cost (normal equation cost trace).
The equation 6 shows the metric to evaluate the parameters predicted provided from gradient descent and normal equation. This new metric is the mean absolute error (MAE), measured in shares, that explicits how far is the prediction from the real value considering its mean. The formula for mean absolute error is:

$$E_{mae}(\theta) = \frac{1}{m}\sum_{m=1}^{m} |(h - Y)| \tag{6}$$

This evaluation formula was chosen to give a more intuitive value for the error of prediction, instead of using the equation 2, however notice that this metric is just to evaluate the model, the gradient descent algorithm is performed using equation 2 as cost function. It folows the table presenting the evaluation of the gradient descent $E_{mae}(\theta)$, evaluated in the trainning and validation dataset for θ obtained from gradient descent and normal equation. The learning rate is equals to 0.1 related to the table below:

Table 1. Fist Experiment

	Gradient Descent		Normal Equation	
	Train Set	Val. Set	Train	Val. Set
$E_{mae}(\theta)$	3104	2903	3092	2908

The first experiment shows an error of prediction around 3000 shares for all trainning datasets, there are two facts to notice: first, both trainning dataset and validation data set are providing an similar absolute mean error. Second, as expected, the normal equation associated error is lower than the error associated from the gradient descent.
From here, the next experiments will be performed with learning rate fixed at 0.1, because figure 1 revealed that this learning rate converges the gradient descent faster, furthermore it will not be plotted any other figure like figure 1 to show gradient descent response, the reason for that is that the response will be similar to the next experiments and the focus is to analyze the error metric adopted.
The next strategy to run the second experiment is to remove outliers targets' examples, this idea came across taking a closer look into the histrogram's (figure 2) targets of trainning dataset.
This histogram was plotted for 500 bins and a maximum value of target of 20000 shares (there are a few outilers above 20000 shares), which is where all the major of ocurrances occur.
The strategy here was to define a threshold of 5000 shares and remove all targets' examples whose targets are greater than this threshold, in order to train the model for the targets with major occurances, accepting the risk of greater errors

for predicting higher values of targets. Applying this technique it was removed around 3000 examples.

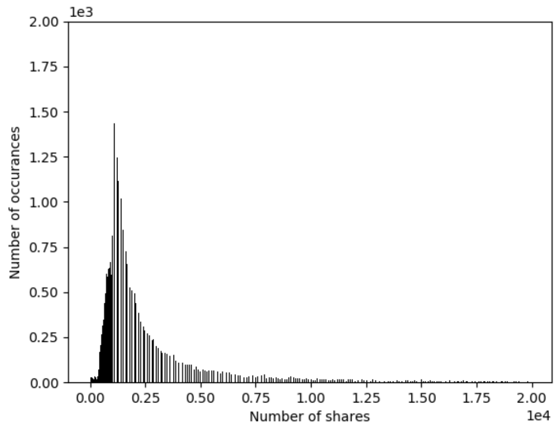

Figure 2. This figure shows the target's histogram, considering the only the targets below 20000 shares.

After the removal of this outliers' examples and running the algorithm it was achieved the results shown in table 2:

Table 2. Second Experiment

	Gradient Descent		Normal Equation	
	Train Set	Val. Set	Train	Val. Set
$E_{mae}(\theta)$	716	2174	709	2165

It is possible to verify good performance enhancement in the four datasets and for both gradient descent and normal equation method. Besides that, it is clear that the mean absolute error of the trainning dataset decreased more than the error for the validation dataset. The previous phenomenon happened because the validation dataset works like a test dataset. During the trainning, the trainning dataset is the only one considered, the validation dataset does not participate of trainning, hence the hypotesis is supposed to be more fitted to the trainning dataset rather than the validation dataset.
The last pre processing data technique to be used is the removal of discrete features. Indeed this dataset contains discrete features or the called dummy features (features that only have two possible values: 0 - False or 1 - True), this removal of features should have worked because once the featuring scaling was performed for all the features, transforming their values to the range 0 to 1, which means that 1 is the highest emphasis and 0 is the lowest enphasis. The discrete features, after the feature scaling, were not affected. It means that they are in the extreme values of the new range of the features, which could be a bad thing for the model trainning. So, it was removed the discrete features and the results can be seen in table 3.

Table 3. Third Experiment

	Gradient Descent		Normal Equation	
	Train Set	Val. Set	Train	Val. Set
$E_{mae}(\theta)$	738	2195	728	2185

To this problem the removal of discrete features almost did not have any effect on the mean absolute error, which was unexpected for the reasons listed before. Probably these discrete features were not so relevant for the trainning to trace patterns of the data.
The next step is to try to increase complexity in order to fit a possible non linearity that the dataset requires. To perform this task it was squared element-wise the set of features, this new squared set of features were concatenated to the previous set of features. Thus, this fact demandaded a dimensionality increase for θ. Therefore, the parameters θ and the features X became:

$$\theta^{\intercal} = [\theta_0, \theta_1, \theta_2, ..., \theta_n, \theta_1^2, \theta_2^2, ..., \theta_{2n}^2] \tag{7}$$

$$X^{\intercal} = [X_0, X_1, X_2, ..., X_n, X_1^2, X_2^2, ..., X_{2n}^2] \tag{8}$$

With the new set of features and parameters the algorithms were ran and the following table presents the results:

Table 4. Fourth Experiment

	Gradient Descent		Normal Equation	
	Train Set	Val. Set	Train	Val. Set
$E_{mae}(\theta)$	731	2189	720	2175

There was no significat change on the error metric results, it might indicates that the linear regression must not be good to attack this kind of problem, because obviously this is not a overfitting model, thus a increase of complexity would tend to minimize the error and that did not happen. It seems that the condition of this current model is underfit, because the error is too high.
Finally, the last model modification is to insert regularization to it. And by adding regularization it means modifing both gradient descent and normal equation:

$$J(\theta) = \frac{1}{2m}\left[\sum_{m=1}^{m}(h-Y)^2 + \lambda\sum_{n=1}^{n}\theta_j^2\right] \tag{9}$$

$$\theta^{new} = \theta_n^{old} - \alpha\left[\frac{\partial J(\theta)}{\partial \theta_n} + \frac{\lambda}{m}\theta_n\right] \tag{10}$$

$$\theta = (X^{\intercal}X + \lambda I_m)^{-1}X^{\intercal}Y \tag{11}$$

Where I_m is an identity matrix, but with the first row composed only by zeros. The results of the model response

Table 5. Fifth Experiment

	Gradient Descent		Normal Equation	
	Train Set	Val. Set	Train	Val. Set
$E_{mae}(\theta)$	731	2189	725	2181

regarding the addition to regularization, for λ equals to 1, can be seen in table 5.

The regularization, as some of the previous techniques, did not present a relevant change in the results for the model trainned by gradient descent and obtained from normal equations. It was expected that the regularization did not show any effect, because regularization is often used to overfitted models or models with high dimensionality, which is not the case. Regularization penalizes the increase of the parameters during optimization, however as previous results shown the gradient descent optimization method was converging, as epochs were passing, to the θ obtained from normal equation, therefore this penalization was not supposed to imply an model significant change, corroborating to explain the innocuity of the regularization in this case. Finally, after the model is validated throught the experiments it is time to use the test dataset. It will be used only one time to test our model in real conditions and check if the model is working properly to the given problem. The results of the test are show in table 6.

Table 6. Test Experiment

	Gradient Descent	Normal Equation
	Test Set	Test Set
$E_{mae}(\theta)$	2607	2610

The results are close from those obtained in the fifth experiment showing that the model replicates very similar the behaviour on the validation dataset to the test dataset, hence it can be considerade a good validation process achievement. On the other hand, the results were not good, because the model absolute mean error is around 2600. This can be considerade good if you are trying to predict an example that gives a target of 69000 shares and the predicted result gives 66400, the relative error is approximately 3.76%, but imagine a case that you are trying to predict an example of a target value of 500 shares and as a predicted result you have 3100, this is not a good prediction at all, consisting of a relative error of 520%.

4. Conclusions and Future Work

Considering that several examples were removed (outliers' target examples) and given the constrains to work with a linear regresson, the results are consistent with the modelling and the tools available to developed both methods gradient descent and normal equation.
It was difficult to analyze the phenomenons that occured during the tests due to the lack of complexity of an linear regression model, by virtue of that most of experiments did not present significant affect on the results of the model.
A future work on this dataset is to try more complex models maybe logistic regression or any other model that has a intrisc non linearity envolved.

5. References

1. http://www.ic.unicamp.br/ rocha/teaching/2018s1/mo444/index.html

2. Pattern Recognition and Machine Learning. Christopher M. Bishop. Springer. (2006) Acrononym: PRML