Combine Regularised Linear Regression and K -Means Clustering for Increase Recommendation Accusatory

Chameera Wijebandara

Computer Science and Engineering, University of Moratuwa
Sri Lanka

chameerawijebandara@gmail.com

***Abstract*— Gradient descent is a flexible and powerful tool for machine learning, but the high computational complexity. And k-means clustering is a method of vector quantization originally from signal processing, that is popular for cluster analysis in data mining. In this paper, we propose a new approach for fast computation of user patten recognition and crusting user documents with a focus on large spatial data sets. We had collect user data about internet browsing and try to identify pattens of the visiting web sits. And also we are going to analyse usage of media files (.avi, .wmv, .mkv) and analysis user pattens and categorize media file and suggest items to user with new experiences. My approach with this problem is quoit strait for word. In this research paper we discuss about what features we are going to pick as most suitable futures, how combine k- mean with reg ration for achieve high accurate recommendation for media file recommendation.**

***Keywords*— Gradient descent, k-means clustering, Recommendation Systems, Mashing learning, Features selection, Purohitha**

I. Introduction

“Purohitha” is a resultant product of this interesting research. Lets get some idea about what Purohitha is. People use computers for various aspects. And browsers and many

of other applications maintain their own history about what user does. But most of history that maintain by applications are only data user cannot get any idea looking only those kind of history.

There are lots of remanding applications for PCs but vast majority of them we have to create alarm /reminder manually. Any of those applications not going to see user history and notify what user have to next. Forgetting work to be done is big problems for most of us and also tight scheduled people. As an example, what happen if student at CSE how normally logging to model every day but one day didn’t log in to model and there was a deadline in that day

Purohitha, which is gives intelligence to desktops. It monitors user activities over Internet, usage of off-line applications and documents used by user then identify the pattern of user and suggest activities for users what is better to do next. Connect with Google calendar and notify event in near future. Give ability to user to add new events to notify in future. Connect with user’s email accounts and gather and notify event in

near future. Ability to integrate with social media and import events. For users convenience this notification will display as a part of the desktop wallpaper For get good performances of Purohitha I have use several tectonics. In this paper I have discus about those tectonics and therms.

For the recommendation peppers I have mostly use Gradient descent method which is commonly used in machine learning applications. Features selection fro the logarithm and normalise those features is very impotent thing in achieve good performance. And also degree of the gradient descent, learning rate, regularisation parameters are very important in come up with perfect solution.

And other impotent approach is combine K-mean crusting algorithm with regression it gives much good performans to our algorithm.

II. Regularised Linear Regression

Gradient descent is a first-order optimisation algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point. If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function; the procedure is then known as gradient ascent.

A. Visualising the dataset

We will begin by visualising the dataset containing historical records on the

change in the water level, x, and the amount of water flowing out of the dam, y.

- A training set that your model will learn on: X, y
- A cross validation set for determining the regularisation parameter : Xval, yval
- A test set for evaluating performance. These are “unseen” examples

which your model did not see during training: Xtest, ytest

The next step will plot the training data (Figure 1). In the following parts, you will implement linear regression and use that to fit a straight line to the data and plot learning curves. Following that, you will implement polynomial regression to find a better fit to the data.

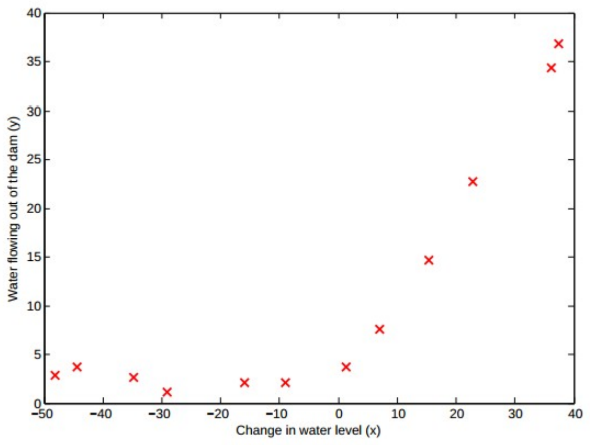


Fig. 1 Visualising the dataset

B. Regularised linear regression cost function

Recall that regularised linear regression has the following cost function:

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2\right) + \frac{\lambda}{2m}\left(\sum_{j=1}^{n}\theta_j^2\right),$$

where λ is a regularisation parameter which controls the degree of regularisation (thus, help preventing over fitting). The regularisation term puts

a penalty on the overall cost J. As the magnitudes of the model parameters

θ_j increase, the penalty increases as well. Note that you should not regularise the θ_0 term.

C. Regularised linear regression gradient

Correspondingly, the partial derivative of regularised linear regression's cost

for θ_j is defined as

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \qquad \text{for } j \geq 1$$

D. Fitting linear regression

Once our cost function and gradient are working correctly, the next part will run the code in train Linear Reg to compute the optimal values of θ. In this part, we set regularisation parameter λ to zero. Because our current implementation of linear regression is trying to fit a 2-dimensional θ, regularisation will not be incredibly helpful for a θ of such low dimension.

The best fit line tells us that the model is not a good fit to the data because the data has a non-linear pattern. While visualising the best fit as shown is one possible way to debug learning algorithm, it is not always easy to visualise the data and model.

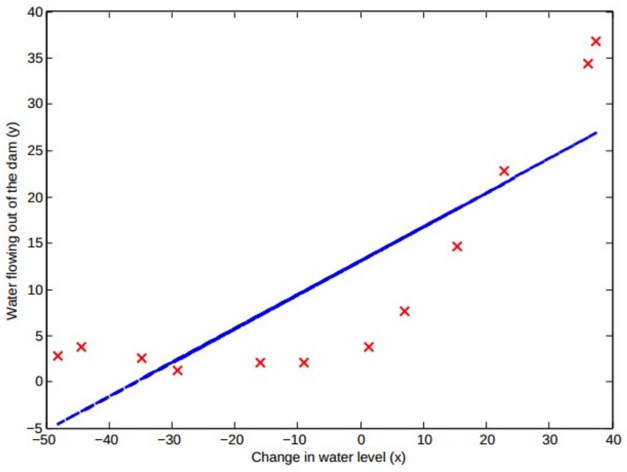


Fig. 2 : Linear Fit

III. Bias-variance

An important concept in machine learning is the bias-variance trade-off. Models with high bias are not complex enough for the data and tend to under fit, while models with high variance over fit to the training data.

A. Learning curves

Then I implement code to generate the learning curves that will be useful in debugging learning algorithms. To plot the learning curve, we need a training and cross validation set error for different training set sizes. To obtain different training set sizes, we have to use different subsets of the original training set X.

In particular, note that the training error does not include the regulation term. One way to compute the training error is to use our existing cost function and set λ to 0 only when using it to compute the training error and cross validation error. When we computing the training set error.

In Figure 3, you can observe that both the train error and cross validation error are high when the number of training examples is increased. This reflects a high bias problem in the model – the linear regression model is too simple and is unable to fit our dataset well. In the next section, you will implement polynomial regression to fit a better model for this dataset.

IV. Polynomial regression

The problem with our linear model was that it was too simple for the data and resulted in underrating (high bias). address this problem by adding more features.

For use polynomial regression, our hypothesis has the form:

$$h_\theta(x) = \theta_0 + \theta_1 * (\text{waterLevel}) + \theta_2 * (\text{waterLevel})^2 + \cdots + \theta_p * (\text{waterLevel})^p$$
$$= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_p x_p.$$

A. Learning Polynomial Regression

Keep in mind that even though we have polynomial terms in our feature vector, we are still solving a linear regression optimisation problem. The polynomial terms have simply turned into features that we can use for linear regression. We are using the same cost function and gradient .

polynomial fit is very complex and even drops off at the extremes. This is an indicator that the polynomial regression model is overfitting the training data and will not generalising well.

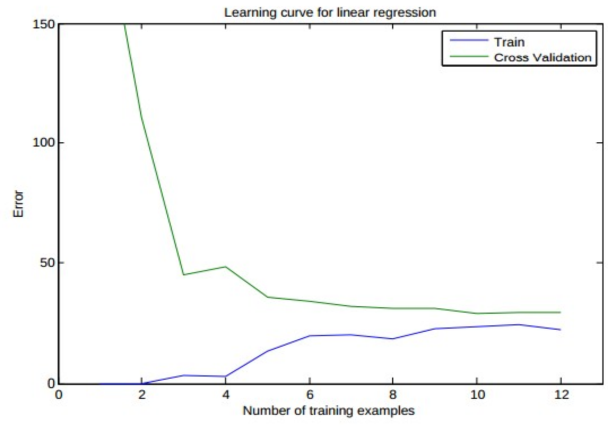


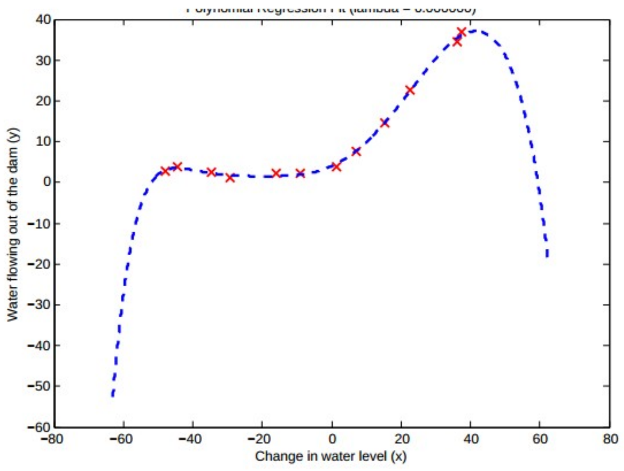
Fig. 3 Linear regression learning curve

To better understand the problems with the unregulated (λ = 0) model, you can see that the learning curve (Figure 5) shows the same effect where the low training error is low, but the cross validation error is high. There's a gap between the training and cross validation errors, indicating a high variance problem.

One way to combat the overfitting (high-variance) problem is to add regularisation to the model. In the next section, you will get to try different λ parameters to see how regularisation can lead to a better model.

B. Adjusting the regularisation parameter

In this section, you will get to observe how the regularisation parameter affects the bias-variance of regularised polynomial regression. For each of these values, the script should generate a polynomial fit to the data



and also a learning curve.

Fig. 4 Polynomial fit, λ = 0

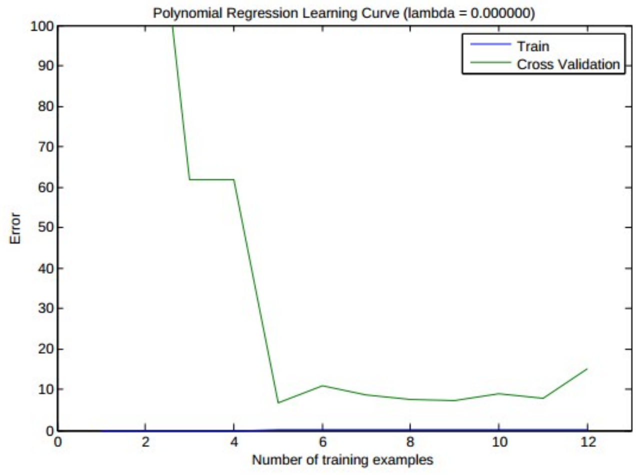


Fig. 5 Polynomial learning curve, λ = 0

C.

For λ = 1, you should see a polynomial fit that follows the data trend well (Figure 6) and a learning curve (Figure 7) showing that both the cross validation and training error converge to a relatively low value. This shows the λ = 1 regularized polynomial regression model does not have the high bias or high-variance problems. In effect, it achieves a good trade-off between bias and variance.

D. Selecting λ using a cross validation set

In this section, you will implement an automated method to select the λ parameter. Concretely, you will use a cross validation set to evaluate how good each λ value is. After selecting the best λ value using the cross validation set, we can then evaluate the model on the test set to estimate how well the model will perform on actual unseen data.

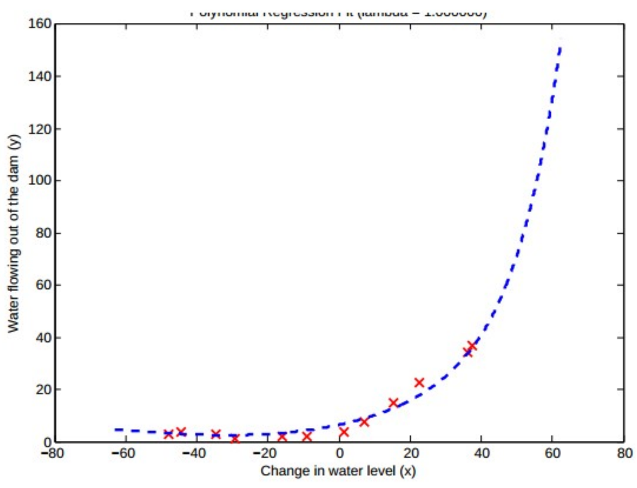


Fig. 6 Polynomial fit, λ = 1

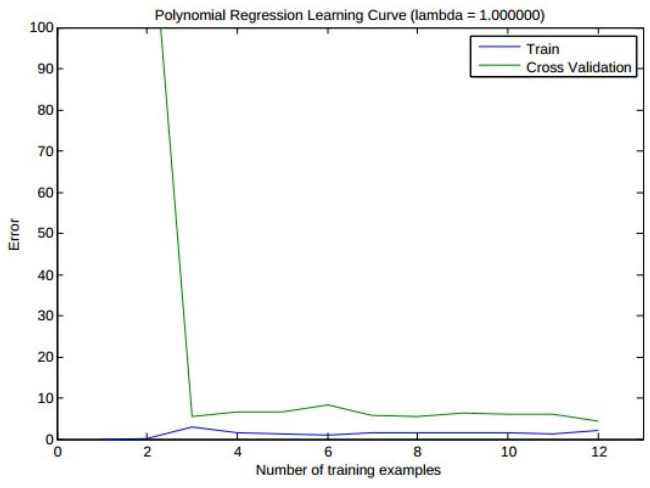


Fig. 7 Polynomial learning curve, λ = 1

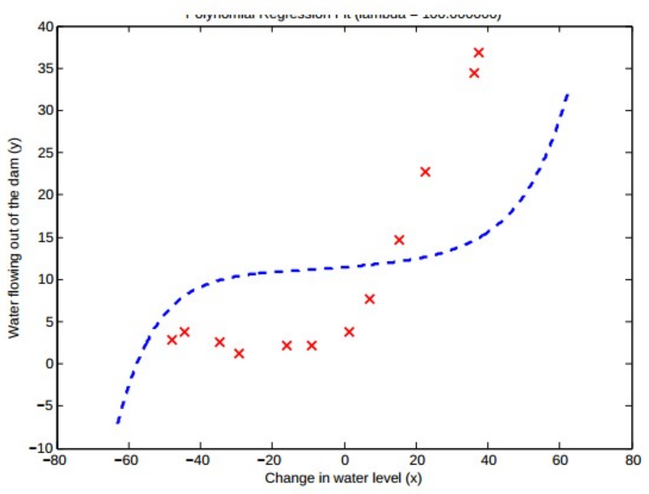


Fig. 8 Polynomial fit, λ = 100

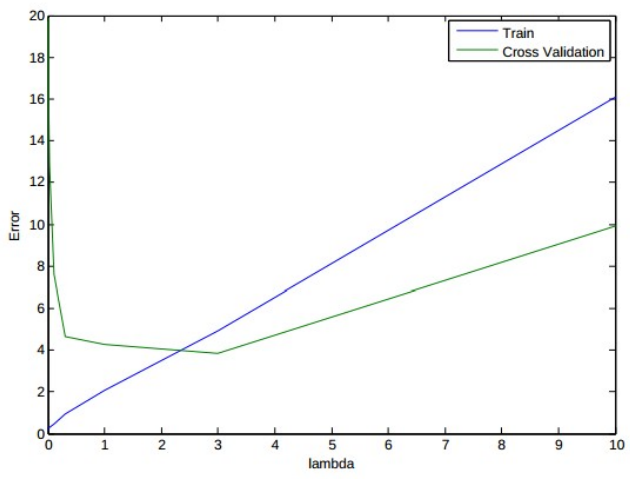


Fig. 9 Selecting λ using a cross validation set

E. Plotting learning curves with randomly selected examples

Concretely, to determine the training error and cross validation error for i examples, you should first randomly

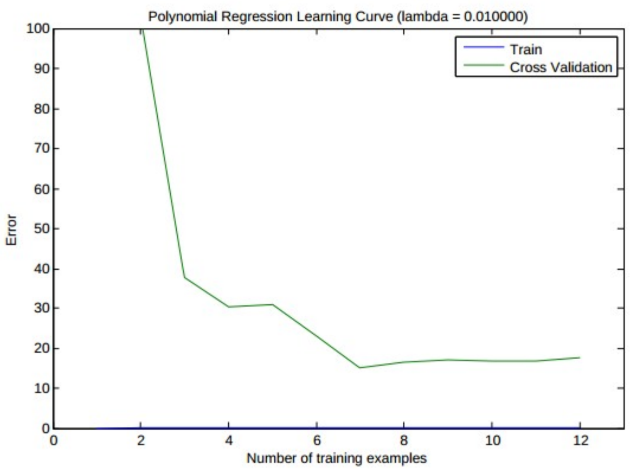


Fig. 10 Polynomial learning curve, λ = 0

select i examples from the training set and i examples from the cross validation set. You will then learn the parameters θ using the randomly chosen training set and evaluate the parameters θ on the randomly chosen training set and cross validation set. The above steps should then be repeated multiple times (say 50) and the averaged error should be used to determine the training error and cross validation error for i examples.

V. K-means Clustering

k-means clustering is a method of vector quantisation originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

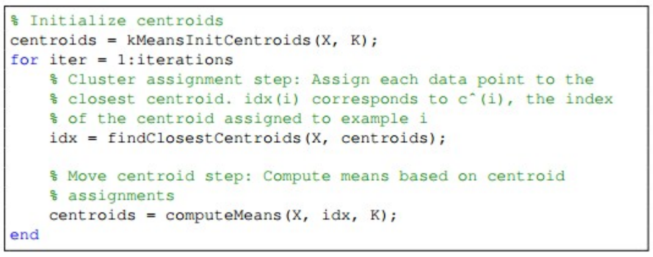
The problem is computationally difficult (NP-hard); however, there are efficient heuristic algorithms that are commonly employed and converge quickly to a local optimum. These are usually similar to the expectation-maximisation algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data; however, k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximisation mechanism allows clusters to have different shapes.

A. Implementing K-means

The K-means algorithm is a method to automatically cluster similar data
examples together. Concretely, you are given a training set {x (1) , ..., x (m) } (where x (i) ∈ R n), and want to group the data into a few cohesive “clusters”.

The intuition behind K-means is an iterative procedure that starts by guessing the initial centroids, and then refines this guess by repeatedly assigning examples to their closest centroids and then recomputing the centroids based on the assignments.

The K-means algorithm is as follows:



```
% Initialize centroids
centroids = kMeansInitCentroids(X, K);
for iter = 1:iterations
    % Cluster assignment step: Assign each data point to the
    % closest centroid. idx(i) corresponds to cˆ(i), the index
    % of the centroid assigned to example i
    idx = findClosestCentroids(X, centroids);

    % Move centroid step: Compute means based on centroid
    % assignments
    centroids = computeMeans(X, idx, K);
end
```

The inner-loop of the algorithm repeatedly carries out two steps: (i) As-signing each training example x (i) to its closest centroid, and (ii) Recomputing the mean of each centroid using the points assigned to it. The K-means
algorithm will always converge to some final set of means for the centroids. Note that the converged solution may not always be ideal and depends on the initial setting of the centroids. Therefore, in practice the K-means algorithm is usually run a few times with different random initialisations. One way to choose between these different solutions from different random initialisations
is to choose the one with the lowest cost function value (distortion).

V.A.1 Finding closest centroids

In the “cluster assignment” phase of the K-means algorithm, the algorithm assigns every training example x(i) to its closest centroid, given the current positions of centroids. Specifically, for every example i we set

$$c^{(i)} := j \quad \text{that minimizes} \quad ||x^{(i)} - \mu_j||^2,$$

where c
(i) is the index of the centroid that is closest to x(i), and µj is the position (value) of the j’th centroid. Note that c(i) corresponds to idx(i) in the starter code.

V.A.2 Computing centroid means

Given assignments of every point to a centroid, the second phase of the
algorithm recomputes, for each centroid, the mean of the points that were assigned to it. Specifically, for every centroid k we set

$$\mu_k := \frac{1}{|C_k|} \sum_{i \in C_k} x^{(i)}$$

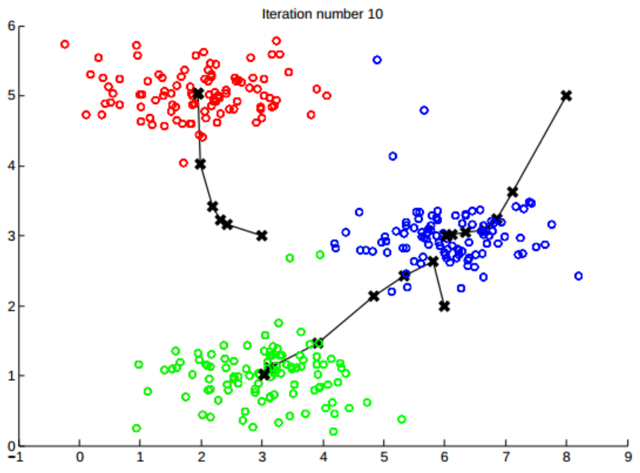


Fig. 11 K-means on example dataset

V.A.3 Random initialisation

The initial assignments of centroids for the example dataset were
designed so that you will see the same figure as in Figure 1. In practice, a good strategy for initialising the centroids is to select random examples from
the training set.

VI. Conclusions

Gradient descent is a flexible and powerful tool for machine learning, but the high computational complexity. And k-means clustering is a method of vector quantisation originally from signal processing, that is popular for cluster analysis in data mining. In this paper, we propose a new approach for fast computation of user pattern recognition and crusting user documents with a focus on large spatial data sets.

We had collect user data about Internet browsing and try to identify pattens of the visiting web sits. And also we are going to analyses usage of media files (.avi, .wmv, .mkv) and analysis user pattens and categorise media file and suggest items to user with new experiences. My approach with this problem is quoit strait for word. In this research paper we discuss about what features we are going to pick as most suitable futures, how combine k- mean with reg ration for achieve high accurate recommendation for media file recommendation.

According to my research I have selected best set of features and set author parameters for good performances and also use k mean for categorise to identify user categorise Vs selected features

As father work we have to identify method to automatically identify user media files such as (.avi, .wmv, .mkv) and categorize them in to movies , songs , tutorials , etc. and then categorize them in to watch and not yet, Sinhalese,English , etc . Then give suggest respected to that categories also.

Then we have use features like New Year season , Christmas season to suggest user interests. And also special dates such as children day. And so may things to consider to develop this research to give good user experiences with PUROHITHA.

Project web site	:	http://goo.gl/1u9Yjs
Technical Blog	:	http://goo.gl/qhVcKL
Demo video	:	http://goo.gl/jp4prT

Acknowledgment

This research paper is made possible through the help and support from everyone, including: parents, teachers, family, friends, and in essence, all sentient beings. Especially, please allow me to dedicate my acknowledgement of gratitude toward the following significant advisors and contributors:

First and foremost, I would like to thank Dr. A. Shehan Perera, Mr. Nisansa de Silva , Mr. Adeesha Wijayasiri and all staff members of CSE most support and encouragement. He kindly read my paper and offered invaluable detailed advices on grammar, organisation, and the theme of the paper.

Second, I would like to thank Dr. Andrew Ng to givring best tutorials abut mashing learning tutorial.

Finally, I sincerely thank to my parents, family, and friends, who provide the advice and financial support. The product of this research paper would not be possible without all of them.

References

[1] S. M. Metev and V. P. Veiko, Regularised linear regression, R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.

[2] J. Breckling, Ed., Least angleregression (with discussion) Berlin, Germany: Springer, 1989, vol. 61.

[3] S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, “A novel ultrathin elevated channel low-temperature poly-Si TFT,” *IEEE Electron Device Lett.*, vol. 20, pp. 569–571, Nov. 1999.

[4] M. Wegmuller, J. P. von der Weid, P. Oberson, and N. Gisin, “"An improved K-means 10. Dong, J. and M. Qi, 2009. "K-means Optimizationclustering algorithm” in *Proc. ECOC’00*, 2000, paper 11.3.4, p. 109.