# Machine Learning Assessed Exercise Report

Apoorva Tamaskar 2349061T
Tsan-Mu,Lin 2339752L
Tianyu Zhang 2291884Z

March 14, 2018

## Abstract

The purpose of this assessment is to build two machine learning models for each regression and classification task. There is a training data set and testing data set for each task. This report will present the best potential models. The best models in the regression section are multiple linear regression and decision tree models, while in the classification section, they are the naive bayes and the gradient boosting. Finally, some details about the experiments, the general ideas and mathematics of these models are illustrated and three literature review will be included in the end of this paper.

## 1 Introduction

The regression task is related to CPU performance. Throughout the years, numerous methods have used various hardware specification of the computer to come up with a proper attempt to evaluate the CPU utility. The dataset in the regression task contains the hardware values of Cache memory size, minimum and maximum number of I/O channels (CHMIN and CHMAX), machine cycle time, and minimum and maximum main memory (MMIN and MMAX) of 168 different computers to train on. Two machine learning models were created by learning from the train dataset and used to predict the test dataset which contains the data of 41 computers. Then, the predictions on the test dataset were submitted to the Kaggle.

The other task is about the classification of human cells. In the field of pathology, tissue microarray (TMA) is a recent innovation to unveil the information of cells. For analysing further information on this, it is better to build a classifier which could distinguish cells between epithelial (where cancer cells live) and stromal (where immune cells and other normal cells live) regions. The training dataset in this classification task contains 600 data points and 112 features. Each data point is constructed from small regions of coherent appearance known as superpixel. Two machine learning models were built by learning from the train dataset and used to predict the test dataset. After that the predictions on the test dataset are submitted to the Kaggle.

## 2 Experiments

For conducting the experiments for this report, libraries such as numpy, scipy and scikit-learn, were used to implement models and to check the score in these particular models.
Then, these scores which were obtained on Kaggle were utilized to finalize our model choice and implement in these models.

### 2.1 Models with lower scores

#### 2.1.1 Classification task

Most models which we used for the classification task and did not give the desired results,some examples are below:

- Random prediction: A random number generator was used to predict the class of the data points. The maximum accuracy was around 60%. Then, to obtain the better results, the means of normalising the columns of each dataset was used and the the performance decreased to 55% accuracy.
- Maximum cardinality: The cardinality of the number of points of each class was calculated and the maximum of them belong to class 1. A blind prediction on this predicts that every test data point belongs to the class 1. This attempt approached an accuracy of around 40%. It implied that a blind guess of every item belonging to class 2 would get approximately 60% accuracy. In short, this may indicate that the test dataset is not divided properly into the public and private leaderboard and may also indicate the training and test data are not uniformly distributed.
- SVM(support vector machine):Since the dataset has high dimensionality, it could be better to use

the SVM method to reduce the dimension of the data. Initially, by training SVM on the entire data set and without dimension reduction, the results revealed around 70% accuracy for the test data. Then, the PCA(principal component analysis) reduction along with SVM was implemented. This decreased the score down to around 65%. Thus, this result would represent that this approach would be over-fitting the training data. For tackling this issue, a 10-fold cross validation was imported. This means was tried to split the training data into 10 sets. It always kept 1 set as the test dataset and the other 9 sets were used as a training dataset. Then, the best results of the 10 were used for a prediction. However, this did not improve the result by much. In the end, the best result in this model was around 72%.

- Decision Tree:In the classification task, since the dataset has 112 features, the decision tree approach could be used to analyse this issue. By using this, it obtained accuracy of around 73% by implementing from the library "scikit-learn" in python, a considerable improvement compared to the score from SVM model. Next, for getting further improvement, normalization of the data was used to modify the dataset and train the model again. This reduced the score to around 60%. Because this is a significant blow to the score, it could imply that the model may be over-fitted. To counter this problem, a 10-fold cross validation was used with the decision trees method. By adapting this method, it showed a considerable improvement to around 78%.

- Random Forest:Due to the improvement in the score by using decision trees as compared to SVM, the random forest model, using scikit-learn library in python to attain a better predictions was the next logical thing to attempt. However, it scored around 74%. Better results could be obtained by using modifications, such as normalization of columns and cross validation as these reduced the performance to around 73%.

- K Nearest Neighbors: As the above models did not provide a desired results. The KNN algorithm with $K = 5$, an algorithm taught in the course, was adapted. However, it only revealed a score around 79%. According to the algorithm, using cross validation would cause the results more fluctuated and make the obtained score unreliable. Hence, this method may not a good choice.

#### 2.1.2 Regression task

Moving to the regression task, some classification algorithms, such as decision tree or gradient boosting, could be used to evaluate the performance of the computers. For this, the following two assumptions are considered:-
First, the given data was assumed to be distributed properly into training and testing dataset.
Second, the data is real data, so for similar training values it is possible that the performance is similar. There is no sudden jump and small variation in values.
As the data was assumed to be distributed fairly, by training the model on the given data we can expect that the classified value of the data won't be too far off from the actual value, thus giving our model a low error rate.

The models which gave higher errors than others are below:

- Gradient Boosting: Simply training the model with the training data gave a error of around 130 on the test data given on kaggle. This by itself is worse performance compared to linear regression model. As it might be possible to achieve the given margin of error with this algorithm, dimensionality reduction using PCA was utilized in this model.This increased the error to around 1034, hence the assumption of using dimensionality reduction while using this approach was given up. The next attempt was using normalizing the data column wise and using the average array of this matrix to convert the values into 0 and 1. If the value is greater than average for that column it gets a value of 1, 0 otherwise. This degraded the error even more, hence did not apply this as well. Lastly, the 10 fold cross validation was simply added in order to get a better model. This reduced the error to 80. In addition to the utilization of models, there were also some modifications for the data values for each point. For further improvement, the variables of MMIN and MMAX are replaced with their sum and took $log_10$. Then, the variables of CHMIN and CHMAX are replaced with their sum and took $log_2$ of the sum. Furthermore, instead of using the performance directly, its $log_2$ value is adapted. This modification achieved an error of around 27.

- Naive Bayes:Multi-classifier Naive Bayes was adapted in the classification task. This did not perform well and had an error of the order $10^{10}$. Hence this model was not used for further experiments.

- SVM:The multi-classifier Naive Bayes was included in the experiment as well and this performed not well as well, the error was around order $10^{13}$. Hence this model was not used for further experiments

## 2.2 Models of choice

In the above experiments, the models are used as the following. These models represent the best results and could be used to better predict in the unseen test dataset.

### 2.2.1 Classification task

- Gradient Boosting: As the amount of data given was not enough to train the models to a reasonable extent, the gradient boosting algorithms were resorted to tackle this problem. This model works well as it uses various different models. Using the gradient boosting algorithm without any modifications, it obtained an accuracy around 81%. This accuracy is much higher compared to other algorithms used in this task.

  This finding conveyed that this model can outperform other models, but this also raised a doubt that the data could be over-fitted. By conducting a 10-fold cross validation on the dataset it showed that the data was not over fitting by much. Hence, manipulation of the features could get a better score. This method could obtain a desirable result with normalizing the data column wise and classifying each element of the matrix as 0 or 1 by checking if the element is less than equal to the average for the column or not respectively. This simple modification boosted the performance to around 85%. To achieve a better results, PCA ,a dimensionality reduction method, was used to check if the performance is boosted. However, it could not find a subset from data which displayed this property and got a worse performance.

- Naive Bayes: According to some references[6], the Naive Bayes method could provide a access on Spam (binary) classification. So, it is conducted to the classification tasks. Initially, training Naive Bayes without any modifications on the entire training data gave a score of around 78%.

  For further improvement, the column wise $5^{th}$ mean of the training dataset was computed. Each element of the matrix was compared to the mean of the corresponding column and replaced the element with 0 if it was less than the average, otherwise it as replaced with 1. After this transformation, it obtained a score of around 82%. Due to the considerable improvement, the cross validation was utilised to check out results and it showed a score of around 81%. In addition, PCA was used to find a subset of features and train the model better. But it was not able to find such a subset, as it revealed a worse performance. Hence, the earlier model is a better choice in this experiment. So, the potential best model would be the naive bayes approach with modification to the data.

### 2.2.2 Regression task

- Multiple Linear Regression:
  As the first attempt for the regression task, the linear regression model was used to examine the data and showed an error around 47 for the dataset. After that, another approach, the variables of MMIN and MMAX were replaced with their sum and divided by 1000, and took $log_2$. Then, the variables of CHMIN and CHMAX were replaced with their sum and took $log_2$ of the sum and variable of CPU performance was replaced by it's $log_2$ value. In particular, the function of the horizontal stack was used to stack the sin, the square and the cube of the data to the matrix. This modification obtained an error around 35 on the test dataset. In addition, the weights obtained from 10 fold cross validation does not provide a significant improvement on the test dataset(error around 36). Hence, the model without validation is considered.

- Decision Tree:
  Training the decision tree model on the given training dataset, it provided an error around 28. For further improvement, the variables of MMIN and MMAX were replaced with their sum and took $log_{10}$. Then, the variables of CHMIN and CHMAX were replaced with their sum and took $log_2$ of the sum. This modification further improved the result to around 24. A cross validation test was used to check for over-fitted or not. Then it showed an insignificant variation in the result. Then, in the end, the model without cross validation would be a better choice.

## 2.3 Comparison between the best performance models

### 2.3.1 Regression Task

The multiple linear regression model showed an error of around 35 and the decision tree method represents an error of around 24, while in normal case this is unexpected. It should be noted that it could be a result of the strong assumption about the training data given to us being real life data. If this condition was violated, it could not obtain such a good result.

### 2.3.2 Classification Task

The gradient boosting algorithm performance is better than Naive Bayes algorithm, around 85% and around 82% respectively. This could be resulted from the amount of training data is not sufficient and gradient

boosting takes into account various different models to get a better prediction.

# 3 Details of the potential best performance models

The previous section identified the models which could obtain the best scores with certain modifications.
Next, the mathematics and algorithms about the above approaches will be detailed later in this section.

## 3.1 Classification task

For the Classification task, the result from Gradient Boosting is the best and the result form Naive Bayes is $2^{nd}$.

### 3.1.1 Naive Bayes

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. The Bayes Theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event. It can be presented mathematically like this: $P(A\,|B) = \frac{(B\,|A)P(A)}{P(B)}$ In this equation, A and B are events and the probability of B that is P(B) cannot equal to 0. The independent probability P(A) and P(B) and the conditional probability $P(B\,|A)$ are already known, and according to which the conditional probability $P(A\,|B)$ can be obtained.[2] The naive Bayes classifiers treat all features independently, and they adapt maximum likelihood method to estimate the necessary parameters for classification, which only require a small number of training data.[1] The function of Naive Bayes classifier is like this:
$p\,(\,x_new\,|t_new = k, X, t) =$
$\prod_{d=1}^{D} p\,(\;x^new_d\,|t_new = k, X, t)$
Where the components of $x_new$ are independent for a particular class, D is the number of dimensions and $x^new_d$ is the value of the dth one. When it comes to the probability computation, one can only work with the naive Bayes model without associating the Bayesian probability or adapting any other Bayesian methods. Although it is a relatively simple probabilistic model, it can achieve considerably high performance which even could be better than some advanced models under some occasion with some pre-processing, which is already proved theoretically by Zhang's research in 2004.[3] Because of the advantage of simplicity and competitive performance, Naive Bayes classifier is widely used in automatic medical diagnosis.[4]

### 3.1.2 Gradient Boosting

Gradient boosting usually produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It combines the weak models into a single strong model in one iteration fashion. Similar to other boosting models, it can be built in stage-wide fashion by optimizing the arbitrary differential loss function. The goal of gradient boosting is to find the minimum value of the differentiable loss function so that the value of F(x) will be the most optimized one. The loss function can be represented as:

$$L(y, F(x))$$

And the equation of gradient boosting is shown below:

$$F(x) = \sum_{i=1}^{M} \gamma_i h_i(x) + \text{const}$$

The gradient boosting method assumes a real-valued y and seeks an approximation $F(x)$ by weighting the sum of $h_i(x)$ $\mathcal{H}$ which are called weak learners. [5]

## 3.2 Regression task

For the Regression task, the decision tree method could be the best approach and the Linear Regression method is $2^{nd}$ among the experiments.

### 3.2.1 Multiple Linear Regression model

Multiple linear regression is a modeling approach whereby one fits a linear relationship between a continuous response variable and a series of explanatory variables. The linear regression model is given by the following formula:

$$y_i = \beta_0 + \beta_1 X_{i1} + ..... + \beta_p X_{ip} + \varepsilon_i$$

For i=1....n Where $\beta_0$ is the intercept term and $\beta_1, ..., \beta_p$ are the coefficients of the explanatory variables.

The linear regression assumptions are given as follows:
1.There is a linear relationship between the response variable and the explanatory variables
2.The values $X_1, ..., X_i$ are fixed values
3.The explanatory variables are independent (i.e. no presence of multicollinearity)
4.Error terms are independent and $N(0, \varrho^2)$ distributed with zero mean and variance $\varrho^2$.
The Strengths of Linear regression are its straightforward and easy to understand and interpret. In addition, it can be updated easily with new dataset. Yet, because there are some assumptions of the linear regression model, it would perform poorly when the dataset is non-linear relationship. Specifically, it may not to capture the complex patterns, and difficult to find the right interaction terms or polynomials[6].

### 3.2.2 Decision Tree Model

The decision tree approach constructs regression models as a tree structure. The goal of this method is to build a model that predicts the value of a dependent variable based on several independent variables. It breaks down the original dataset into a series of smaller subsets and an associated decision tree is increasingly developed. The cornerstones of decision tree are decision nodes and leaf nodes. The former represents the value for the attributed tested while the later shows the different decision on the numerical targets. In addition, the top decision node is called root node which implies the best predictor (Figure 1).

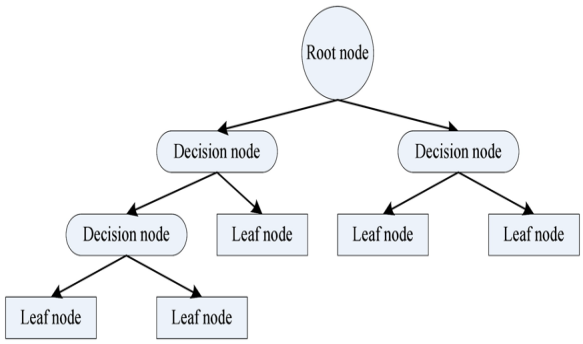


**Figure 1. The sketch of the Tree Model**

Other key features of the tree model approach are that decision tree model is a non- parametric, non-linear model and under no underlying assumptions[7]. In addition, it could deal with both categorical and numerical data and does not need too much data preparation. To sum up, the advantages of tree model are below. It could handle complex relationships between the predictor variables and the dependent variable in many different circumstances and deal with both categorical and numerical data. However, sometimes it could create extremely complex model.

# 4 Literature Review

- Generative Adversarial Nets:
  Goodfellow and his colleagues proposed a framework to estimate generative models via an adversarial process, which could solve the problems that approximating intractable probabilistic computations and utilizing the piecewise linear units in the generative context. The experiment adapted two models to train the data. One is the generative model which can catch data distribution, the other is the discriminative model which can decide whether a sample is from model distribution or data distribution. And authors explored the adversarial nets. The current researchers developed generative machine to generate samples from needed distribution. And using Noise-contrastive estimation (NCE) and predictability minimization and adversarial examples were also introduced to improve the performance. According to the results of the experiment, the new method do not need Markov chains, all the problems are solved and the computation ability is improved as well.

- Distributed Representations of Words and Phrases and their Compositionality:
  To capture numerous relationships of syntactic and semantic is complex. Mikolov et.al. propose that "Skip-gram model" could extract the information from abundant unstructured messages in an efficient method. In this paper, authors reveal several prolongations of the original Skip-gram model. "Hierarchical Softmax", Negative sampling algorithm ", "Negative Sampling" and "Subsampling of Frequent Words" are illustrated, respectively. This essay adopts the analogical reasoning task to examine these approaches mentioned above and reveals that these methods have obvious improvement in examining the learned words or phrases, even for the uncommon entities. In particular, the subsampling method could provide a faster training and considerably representations of unusual words, while Negative sampling algorithm might markedly improve to examine frequent words. In the end, this essay concludes that different tasks have different optimization parameters.

- Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells:
  In order to solve the problems about the existing potential confounding factors which could affect the expression of the heterogeneity of gene and the ability about identify subpopulations, Buettner and et al. proposed a computational approach to handle the hidden factors. The previous works adapted imaging techniques such as RNA-FiSH (fluorescence in situ hybridization) to measure singlr-cell gene expression. But the amount of gene it can measured is small. Compared with the previous method, authors used single-cell latent variable model (scLVM) to examine the hidden factors and test it through a population of staged mouse embryonic stem cells (mESCs). After proving scLVM can help to identify the meaningful subpopulation of cells, this model is applied to explore the cell differentiation of T helper 2 (TH2). The result shows that scLVM model is able to eliminate variation due to the cell cycle and potential confounding factors before analysis step.

# References

[1] Narasimha Murty, M.; Susheela Devi, V. (2011). Pattern Recognition: An Algorithmic Approach. ISBN 0857294946.
[2]*Stuart*, *A.*; *Ord*, *K.*(1994), *Kendall'sAdvancedTheory ofStatistics* : *VolumeIDistributionTheory*, *EdwardArnold*, 8.7.[3]*ZhangH.*(2004).*TheOptimality ofNaiveBayes.Faculty ofComputerScience*, *UniversityofNewBrunswick. http* : *//www.cs.unb.ca/ hzhang/publications/ FLAIRS*04*ZhangH.pdf*
[4]*RishI.*(2001).*Anempiricalstudyofthenaive Bayes classifier.IBMResearchDivision*, *RC*22230(W0111-014), *ComputingScience.http* : *//www.research.ibm.com/people/r/rish/papers /RC*22230*.pdf*
[5]*ChengLi.*"*AGentleIntroductiontoGradientBoosting*"
[6]*IanH.WittenandEibeFrank*(2005), *DataMining PracticalMachineLearningToolsandTechniques SecondEdition*, *SanFrancisco*, *MorganKaufmann*
[7]*AbhijitGhatak*(2017), *MachineLearningwithR*, *Singapore*, *Springer*