THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP 5318

Assignment name: Assignment 2

Tutorial time: Mon 20:00 -21:00 **Tutor name:** Harrison Nguyen, Awani Kumar

DECLARATION

We the undersigned declare that we have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, an, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Academic Dishonesty and Plagiarism in Coursework Policy* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1. Sheng Yuan	430058318	Yes / No	Yes/No	
2. QinQing Peng	470476783	Yes / No	Yes / No	Qinqing Peng
3. Haoran Liu	470162749	Yes / No	Yes / No	
4.		Yes / No	Yes / No	
5.		Yes / No	Yes / No	
6.		Yes / No	Yes / No	
7.		Yes / No	Yes / No	
8.		Yes / No	Yes / No	
9.		Yes / No	Yes / No	
10.		Yes / No	Yes / No	

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

2018

Assignment 2 -Group 3

QINQING PENG (QPEN0175), SHENG YUAN (SYUA5465), HAORAN LIU (HLIU4246)

TUTORS: HARRISON NGUYEN, ASWANI KUMAR

Table of Contents

Abstract

In this report, the cover type dataset was selected by our group. We applied 3 classification methods as KNN, Random Forest and Logistic Regression to predict 7 forest types. Our focus on both improving the accuracies of the classifiers and comparing the performance of them. Corresponding to the data pre-processing methods, one hot encoding, normalization and Principle Component Analysis (PCA) were selected to apply control variable method to analyse how each method influences on the overall results of each classifier.

Introduction

- What is the problem you intend to solve?

This assignment is to solve a real-world problem by applying machine learning and data mining methods. The target is to select 3 types of classifiers to classify a chosen dataset and compare their performance. Cover type dataset was selected by our group. This is a textual dataset in the UCI Machine Learning Repository which recorded the forestay data from 4 wild areas in Roosevelt National Forest in northern Colorado, USA. It contains 581012 instances, each one has 12 attributes, separated in 7 classes. After literature reviewing, Logistic Regression, Random Forest and KNN were selected. To apply these classifiers, Sklearn library will be utilized. Meanwhile, various parameter adjusting progress and data pre-processing methods are applied.

- Why is this problem important?

The problem is to classify the different forest cover types by analysing their features. This research has great potential in forestry and geography. In the research of Thomas and Aravind [1], they indicated to improve the performance of classifier on this kind of problems can help to increase the efficiency of investigating in wild environment in the field of forestry and relevant organizations such as USFS (US Forest Service). [1] For example, a fine-grained classifier for this dataset can do transfer learning to other forest datasets in the other locations. In addition, some institutions, such as Kaggle, will have competitions to collect ideas from participants for future researches.

Previous work (Literature review)

The cover type Dataset is a famous and traditional dataset which was collected in 1998 in the UCI machine learning repository. Many researchers concentrating in machine learning chose this dataset to test the performance of the classifiers they designed. Therefore, referring to previous work, before designing the new classifiers, would be helpful for the future researchers. This part will introduce a study on this dataset by Thomas, Aravind. In addition, some algorithms from Hugo Sjoqvist's study will also be introduced.

Thomas and Aravind, who are researchers from NYC Data Science Academy, did a study on a competition hosted by Kaggle [1], which selected a training set of 15120 observations and a test set of 565892 observations. Such a large data size made this classification task very challenging. In this study, the researchers decided to use packages of machine learning and visualization in R language. This study covers not only the processes of designing classifiers and comparison of the results, but also a professional investigation of all important attributes including elevation, aspect, slope and others of the dataset. For example, by the visualization of the distribution of all classes in the scale of elevation, the elevation distribution diagram was generated:

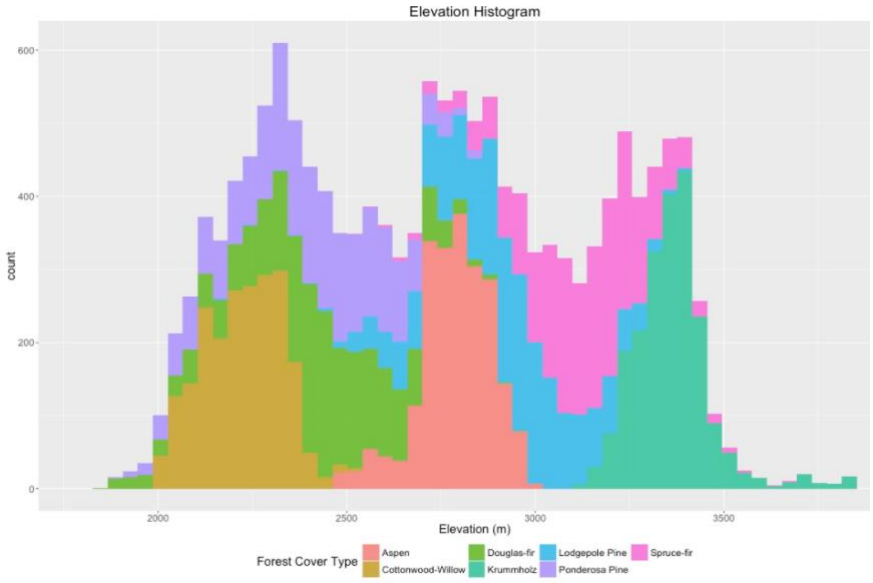


Figure 1. Elevation distribution diagram [1]

In this diagram it can approximately find that the train data was equally distributed among the 7 classes. [1] Besides, it can prove the elevation is a reliable parameter to classify the cover type. For example, the Cottonwood Willow plants can only exist in the range of about 2000 to 2500

meters elevation in this training set. Furthermore, Thomas and Aravind analyzed all attributes existing in the dataset and plotted figures for them.

Apart from this, they also combined the influence of attributes to synthetically analyse the characters of each class such as in Figure 2, Hillshade_9am, slope and elevation have been analysed together by building a 3-dimension Cartesian coordinate.

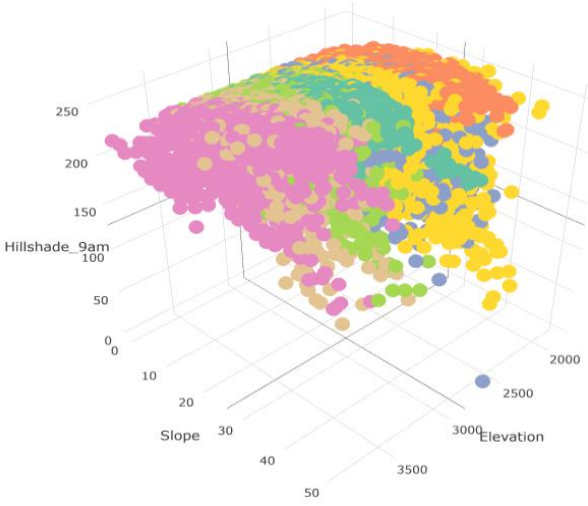


Figure 2. Elevation distribution diagram 2 [1]

In the modelling part, they applied 3 methods: Multinomial Regression, Neural Networks and Tree based methods (Random Forest and etc.). The Multinomial Logistic Regression based on the original dataset. However, the others are based on both the original dataset and the Feature Engineering. Feature Engineering is a method to pre-process on the original attributes. For example, the values of the elevations were transformed to the logarithm forms as 'log_elevation'. This will improve the accuracy of Neural Networks and Tree based methods.

In the part of Logistic Regression, Lasso regularization and Ridge regularization performed as the data pre-processing. To concurrently use both can produce an elastic-net regularization. Finally, the best accuracy will attain 59.59%. [1] For neural network, a 54-100-7 node neural nets on training data achieved 52.11% accuracy for 300 max iterations and 52.43% for 500 max iterations. The other methods were not considered to be reference in this assignment.

For the research of Hugo Sjoqvist, it is much simpler in data analysis and visualization, but the methods are quite different. Support Vector Machine, Naiver Bayes and Decision Tree. The data

pre-processing methods are a little bit different. Random Forest selection, Lasso and PCA were picked. [2]

For overall accuracy, SVM got 89.1% under RF selection, but only 81.8 by Lasso; NB got 66.8% after Lasso, but only 48.0% after PCA; finally, for Random Forest, after PCA it owned 94.7% accuracy, but 75.2% after Lasso. [2]

These 2 articles are a part of literatures referred by this assignment. The detailed methods could be very different for optimizing the accuracy and comparing the performance.

Methods

1. Pre-processing

1.1 One hot encoding

One hot encoding is a process that convert categorical variables with N-state into a N-bits representation form while only one bit is 1, which makes machine learning classifier perform a better job in prediction [3]. For example, in the Covtype data set, the Wilderness Area and Soil Type are already in one hot encoding format. Moreover, the Cover type, containing 7 types, can be also represented in one hot encoding format rather than in form of 1 to 7. For example, for the first cover type, we would transfer it to one hot encoding as [1, 0, 0, 0, 0, 0, 0], for the third cover type, we would transfer it to one hot encoding as [0, 0, 1, 0, 0, 0, 0] and etc.

1.2 PCA

Principle component analysis (PCA) is a popular method to dimensionality reduction and to overcome the curse of dimensionality. The curse of dimensionality is that if the number of a data set's features is significantly exceeding the number of samples, the classification model will be over complicated, and prediction might be overfitting [4]. By applying PCA, the features of the data set can be maintained. The following section introduces the steps of the PCA algorithm [5].

Step 1: Let X be the matrix of the data set.

Step 2: Subtract the mean $\mathrm{X_{mean}} = \mathrm{X} - \overline{\mathrm{X}}$

Step 3: Calculate the covariance matrix $\mathrm{A} = \mathrm{X_{mean}} * \mathrm{X_{mean}^{T}}$

Step 4: Calculate the eignvectors and eigenvalues of A

Step 5: Choose the components n and form a feature Vector

$= (\mathrm{eig_1}\ \mathrm{eig_2}\ \mathrm{eig_2} \ldots \mathrm{eig_n})$, where $\mathrm{eig_n}$ is eigenvetor with top n eigenvlaue

Step 6: Derive the new data set $\mathrm{X_{new}} = \mathrm{FeaturesVector} * \mathrm{X}$

The explained variance is a significant aspect for the PCA performance and can be used to find the optimal component number. The total explained variance is defined as the sum of the variances of each component, which shows the feature reservation percentage of the original data set [6]. Hence, for K principal components analysis, the total explained variance is:

$$\text{Total explained variance} = \frac{\sum_i^k \lambda_i}{\sum_j^m \lambda_j}$$

$$\text{where } \lambda_i \text{ is the eigenvlaues of component and n is the orignal dimension}$$

1.3 Normalization

Data normalization is a feature scaling method, which can rescale the attributes to the range from 0 to 1 in general or other range with given minimum and maximum number. It is beneficial when the data set has a wide range of values. For instance, if one of the features has enormous value compared with others, it will definitely dominate the prediction of classifiers, which implement the Euclidean distance, such as KNN. Moreover, it is reported that the gradient descent converges perform faster with normalization [7]. The process of the normalization is:

$$x_{\text{nomarlized}} = \frac{x - \min(x)}{\max(x) - \min(x)}, \text{where x is the feature column}$$

2. Classification Methods

2.1 Logistics regression

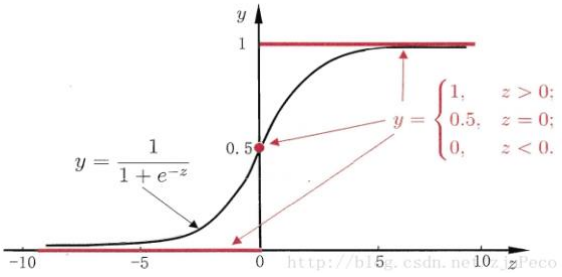
Logistic Regression is a traditional method for classification. Initially, this method was applied to binomial classification problems. It requires that the output label y will only be $y \in \{0, 1\}$ by the input prediction value $z = w^T x + b$.

To achieve this goal, an activate function – sigmoid function was taken into use. The form of sigmoid function is:

$$h_\theta(x) = \frac{1}{1 + e^{-x}}$$

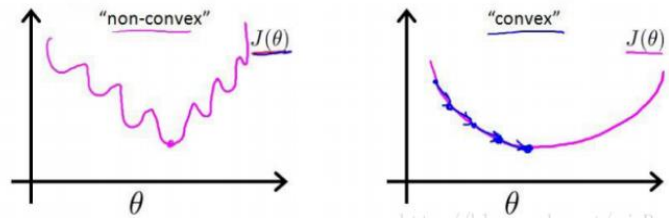
This activate function has 2 main advantages. Firstly, it can limit the output values of y in the range of {0, 1}, which can be equally divided buy the y axis and the point (0, 0.5). It can be clearly observed in Figure 3. When y is under 0.5, it will be label 1; when y is over 0.5, it will be label 2. Meanwhile, without applying sigmoid function, the cost function will always be non-convex (Figure 4). This is will be very hard to utilize the gradient decent to reach the global

minimum. The sigmoid function can transform the function to be convex, which is easy for iteratively gradient decent.



(Ref: https://blog.csdn.net/zjuPeco/article/details/77165974).

Figure 3. The figure of sigmoid function



(Ref: https://blog.csdn.net/zjuPeco/article/details/77165974)

Figure 4. The non-convex and convex function diagrams

The cost function is a critical concept in Logistic Regression. The basic theory is to fit the curve represented by the model to the values of training dataset by updating weights. The cost function is to reduce the loss between the original training dataset and the curve. The cost function:

$$\mathrm{Cost}(h_\theta(x), y) = -y\log\left(h_\theta(x)\right) - (1-y)\log(1-h_\theta(x))$$

To reduce the cost, gradient decent method is commonly used. The general process is locating the start points on the cost function curve and initializing a learning rate, then process the gradient decent iteratively and refresh the input theta values (weights):

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)}\log h_\theta\left(x^{(i)}\right) + (1-y^{(i)})\log(1-h_\theta\left(x^{(i)}\right))]$$

For the iteration, it repeats

$$\theta_j: \theta_j - \alpha\frac{\vartheta}{\vartheta\theta_j}J(\theta) = \theta_j - \frac{1}{m}\sum_{i=1}^{m}(h_\theta\left(x^{(i)}\right) - y^{(i)})x_j{}^{(i)}$$

to get the $\min_\theta J(\theta)$.

2.2 KNN

As for classifier selection, we decided to perform k-nearest neighbours, which is also called KNN algorithm. In the KNN classifiers, there are three required elements: store records, distance measure and a value of k.

In the conventional procedure of KNN, the nearest k neighbours should be found so that it would be classified to the corresponding record. Then a rule (e.g. by majority decision rule) would be used in order to classify the record. In this process, the record is classified as one of the majority

class of the k neighbours. Among all the steps, the most crucial step for KNN algorithm is to measure the distance. Euclidean Distance is the most popular method to be performed and the predictors should be standardised or normalised in this procedure [8]. The Euclidean Distance can be calculated as below (Suppose record #1 $(t_1, t_2, \ldots, t_p)$ and record #2$(u_1, u_2, \ldots u_p)$):

$$\text{Euclidean Distance} = \sqrt{(t_1 - u_1)^2 + (t_2 - u_2)^2 + \cdots + (t_p - u_p)^2}$$

Choosing value of k would also have obvious impact to the classification. Higher values of k can help the model with alleviating overfitting. When k = n-1, the classifier performs the same rule as naïve rule. When k=1, which is the simplest case for KNN algorithm, it would still be very powerful classification tool, especially when we have large number of record. Overall, the selection for value of k should be based on which k has the best performance of classification.

2.3 Random forest

Random forest is one of the most powerful machine learning classification algorithms, which was firstly introduced by Ho in1995 and finally developed by Breiman in 2001[9]. As its name random forest, it is an integrated model with a number of decision trees to create the forest.

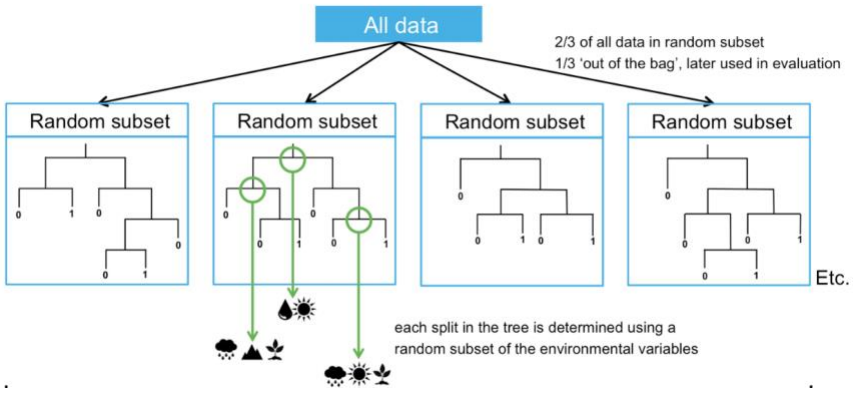
Decision trees is a top-down approach, where the tree structure is constructed by different types of nodes. Starting with the root node, the process generates a binary split in each node based on the attributes and it is executed until it meets the terminal criteria. Finally, the end node represents the predicted outcome, or the predicted class. The node splitting is the significant part of the decision trees, which can be achieved by implementing methods such as the Gini impurity [10].

$$\text{Gini impurity } G(K) = \sum_{i}^{J} P(i) * \big(1 - P(i)\big)$$

where $P(i)$is the probability of a certain classification i

The random forest algorithm applied a technique called bagging [11]. Firstly, it separates the dataset into K numbers bootstrap samples. Each random subset creates a decision tree. Hence, for each test data, it has several possible outcomes corresponding to each decision trees. To make a prediction, the class with the highest number is selected. In general cases, the higher number

of trees will provide a higher accuracy results [12]. The diagram of random forest classifier is shown in the Figure 5.



Ref: (https://support.bccvl.org.au/support/solutions/articles/6000083217-random-forest)

Figure 5. Random forest classifier diagram

3. Validation Method – K-fold cross validation

In K-fold cross validation, the original data set is split randomly into K subset. In each time, one of them is used as the test data set and the remaining K-1 subset are used as the training data set. This process will repeat K times. The main benefit of this method is that the whole data set are used for both training and validation purpose, while each observation is only used for validation once.

Experiments and Discussion

1. Execution configuration

Hardware

- MacBook Pro (13-inch, 2017, Two thunderbolt 3 ports)
- Processor 2.3 GHz Intel Core i5
- Memory 8 GB 2133 MHz LPDDR3
- Graphics Intel Iris Plus Graphics 640 1536MB

Software

- Jupyter notebook (Version 5.4)

Runtime

Implement the python library time

2. Procedures description

Data inspection

(1) Find the total number of samples for each cover type

(2) Find the summary statistics for each feature

Pre-processing

(1) Observe the impact of different principal components number for each classifier
(2) Find suitable principal component number for each classifier
(3) Observe the impact of different normalization range

Logistics regression

(1) Observe the impact of amount of sampling data
(2) Observe the impact of whether using normalization or not

KNN

(1) Observe the impact of different K nearest number
(2) Find optimal K nearest number

Random forest

(1) Observe the impact of different tree number
(2) Find optimal Tree number

Comparison with confusion matrix and classification report

(1) Compare the performance between KNN, LR, and RF.

3. Results and Comparison

3.1 Results

3.1.1 Data Inspection

The total number of samples for each cover type is shown in the Figure 6. According to the data description [13], the summary statistics for each feature is displayed in the Table 2. The total record of samples is 581012. Among them, type 2 has the largest quantity of 283301records, followed by type 1, which has 21840 records. However, the amount of type 4 has only 2747 records, which is approximately one percent of the amount of type 1. Therefore, in our data set splitting section, the types with fewer record such as type 4 would have few or even no record in the subset, which could result in a poor prediction for such types. In other words, the types with large quantity, such as type 1 and type 2 may govern the prediction.

Since the features are measured in different units, their ranges are widely varied. With higher standard deviation, the Euclidean distance is larger. For example, the features, horizontal distance to roadway and horizontal distance to fire points, will dominant the distance calculation in KNN classifier without any pre-processing. In our work, the features are treated as independent variable. Hence, the rescale function, such as normalization or standardization, can be applied in the pre-processing.

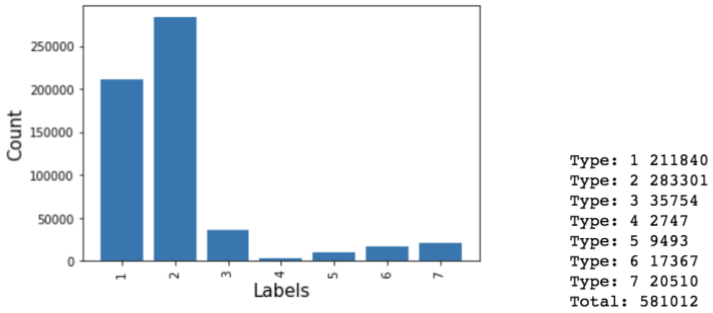


Figure 6. Total number of samples for each cover type

Feature's name	Measurement in units	Mean	Standard deviation
Elevation	Meters	2959.36	279.98
Aspect	Azimuth	155.65	111.91
Slop	Degrees	14.1	7.49
Horizontal distance to Hydrology	Meters	1269.43	212.55
Vertical distance to Hydrology	Meters	46.42	58.30
Horizontal distance to Roadways	Meters	2350.15	1559.25
Hill shade 9am	0 to 255 index	212.15	26.77
Hill shade noon	0 to 255 index	223.32	19.77
Hill shade 3pm	0 to 255 index	142.53	28.27
Horizontal distance to Fire Points	meters	1980.29	1324.19
Wilderness Area	0 (absence) or 1 (presence)	/	/
Soil types	0 (absence) or 1 (presence)	/	/

Table 2. Summary statistics

3.1.2 Preprocessing

3.1.2.1 PCA

The Figure 7 shows the influence of principal components number for each classifier on running time and accuracy. And the total explained variance for each principal component number is shown in the Figure 8. The range reduction of reduction number is from 2 to 10. In this section, only the PCA performance is analysed. Therefore, the K nearest number for KNN classifier is simply selected as well as the tree number for random forest classifier, which are 15 and 5 respectively.

For the running time, the increase of reduction number has a significant impact on the running time of random forest classifier, but rarely on the other two. For measuring accuracy, it is obvious that the accuracy for all of methods increases till their limitation, while the dimension is

growing. It can be explained by total explained variance in the Figure 8. As the total explained variance becomes larger, more features are reserved, which will result in a better prediction.

In conclusion, considering the aspects including high performance and low execution time, we chose the optimal principal component number as 5 for KNN and random forest in the rest parts of analysis. And for logistics regression classifier, the prediction performance is not powerful enough with a small principal component number (E.g. Only around 0.6 in the range 2 to 5). Hence, we chose a larger principal component number for logistic regression, which is 10.

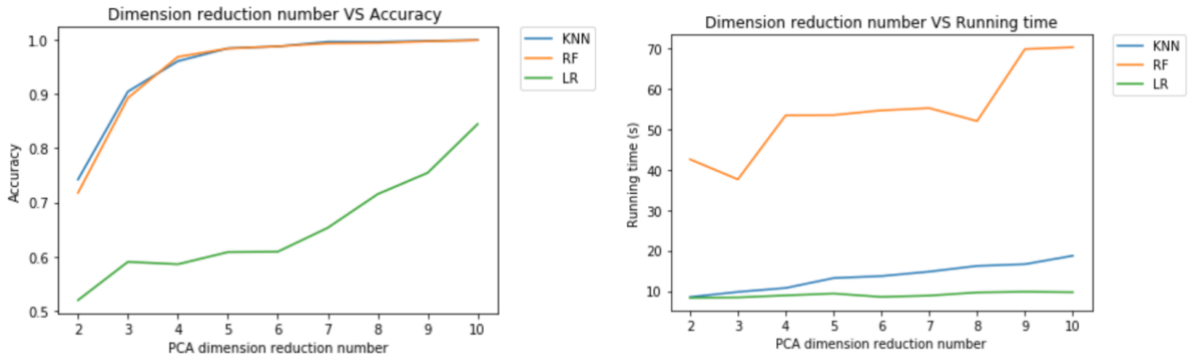


Figure 7. Dimension reduction number vs Accuracy and Running time

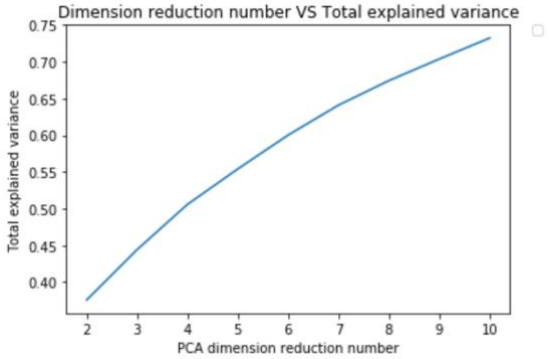


Figure 8. Dimension reduction number vs Total explained variance

3.1.2.2 Normalization Range

The Figures 9 displays the influence of rescale range on the accuracy of the classifier. The range is from (0,1) to (0,100) and only logistic is tested. Actually, even the range turned from 1 to 100, the accuracy fluctuates slightly at around 0.848. And the difference is around 0.004, which can be ignored. We can conclude that normalization range will not affect the prediction accuracy. Hence, for the rest parts, the default range (0, 1) is chosen.

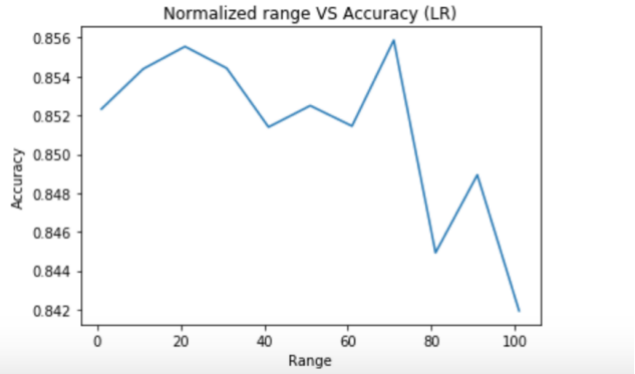


Figure 9 Impact of normalized range on accuracy

3.1.3 Logistic Regression

In Figure 10 shows the result with out normalization as a contrast. If only apply PCA on the original dataset, the accuracy would not be very high, which is approximately 60%. The serious problem is that only the three largest class can be predicted. The classes which only contain small numbers of records will be ignored by the classifier. One possible reason is shown in the warning saying that the coefficient did not converge because the maximum iteration was reached.

```
/Users/shengyuan/anaconda3/lib/python3.6/site-packages/sklearn/linear_model/sag.py:326: ConvergenceWarning: The max_i
ter was reached which means the coef_ did not converge
  "the coef_ did not converge", ConvergenceWarning)
```

	precision	recall	f1-score	support
1	0.62	0.80	0.70	84633
2	0.76	0.58	0.66	113390
3	0.38	0.79	0.52	14269
4	0.00	0.00	0.00	1065
5	0.08	0.01	0.02	3906
6	0.06	0.04	0.05	6995
7	0.03	0.01	0.01	8147
avg / total	0.63	0.63	0.61	232405

Figure 10. The performance of the model before applying normalization

Referring to the solutions of the previous work, the numbers of each class were changed to be the exactly same – 2500 samples. This step was aim at guaranteeing each class can be trained in an equal probability. However, the result of improvement was even worse, only about 50%, but all the types were predicted. Figure 11 displays the performance of this method.

	precision	recall	f1-score	support
1	0.43	0.22	0.29	1002
2	0.49	0.26	0.34	994
3	0.49	0.34	0.40	997
4	0.50	0.89	0.64	1020
5	0.79	0.52	0.63	965
6	0.54	0.44	0.48	1029
7	0.49	0.98	0.65	993
avg / total	0.53	0.52	0.49	7000

Figure 11. The performance of the model by sampling the data

To make logistic regression classifier achieve high prediction accuracy, we try different solver. saga is good for multiclass problems and can handle L1 penalty. Apart from adjusting the parameters in the Logistic Regression model itself, it is also important to investigate how the data pre-processing could influence on the model. As mentioned in the method section, one advantage of normalization is that the gradient decent converges perform faster with normalization. Hence, our best solution to is using the normalization to rescale the data set into range 0 to 1 and then execute the PCA. The results are that not only the the warning is disapear, but also the accuracy is above 80%.

	precision	recall	f1-score	support
1	0.82	0.77	0.79	84633
2	0.83	0.87	0.85	113390
3	0.95	0.99	0.97	14269
4	1.00	0.28	0.43	1065
5	0.97	0.94	0.95	3906
6	0.98	0.98	0.98	6995
7	0.97	0.99	0.98	8147
avg / total	0.85	0.85	0.85	232405

Figure 12. The results after using normalizaion

3.1.4 K-Nearest Neighbor

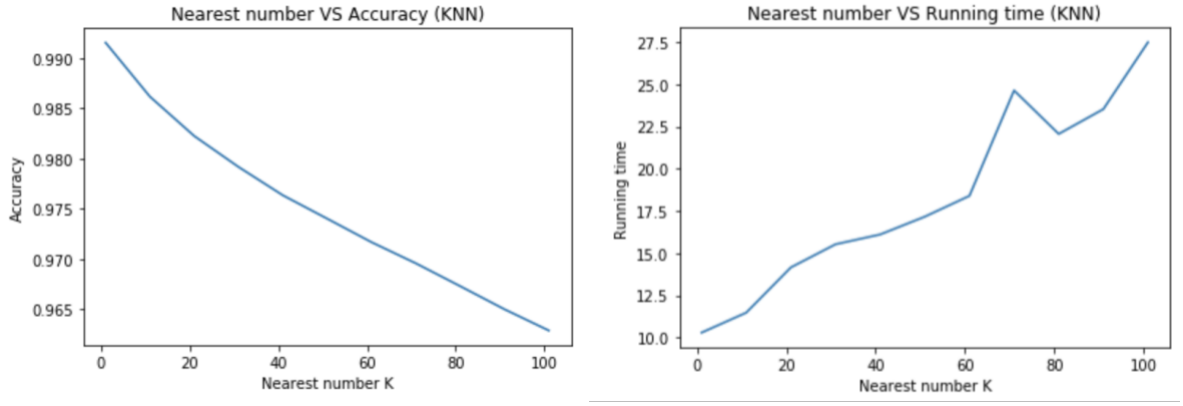


Figure 13. The performance of KNN corresponding to the K nearest number

In order to understand how different value of k impact on the accuracy of the classifier and the overall run time, we plotted the above two line-charts to show the correlation between those variables. As we could see on the left chart, when we choose the value of k ranges from 1 to 100, as the value of k increases, the accuracy of the classifier decreases. Then, we look at the run time of the program, as we can see, as the value of k goes up, the running time would also increase. The reason for such relationship is that when value of k increases, the sorting process becomes more complicated and it would cause more time.

In this classifier, we used optimal k of 1 for our KNN classifier to achieve high accuracy and efficient runtime. According to the above table, the overall accuracy of KNN classifier when k = 1 is 99.01%.

3.1.5 Random Forest

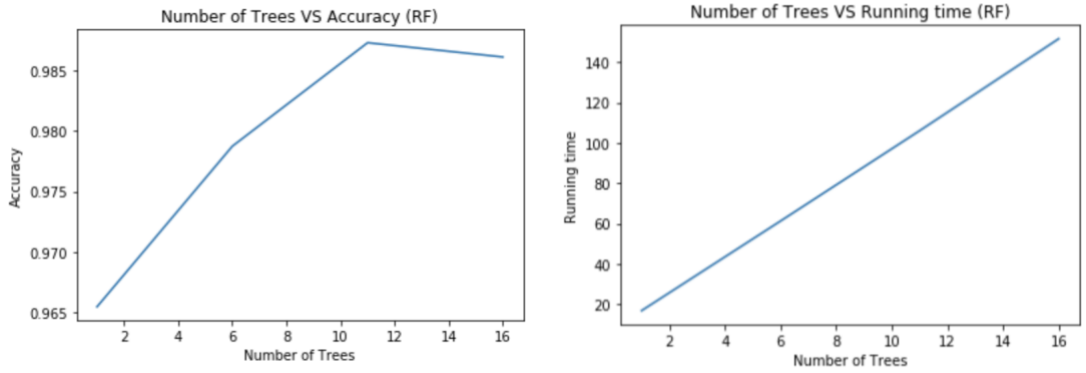


Figure 14. The performance of Random Forest corresponding to the number of trees

As for random forest classification we also performed normalization to the dataset. The above two line-plots shows the relationship between number of trees to accuracy and to running time.

From the result shown above, in the range from 1 to 10, the accuracy is positive correlation with the number of trees. when the number of tree is around 10 to 12, the accuracy is stay around 0.983. According to the Figure 14, the running time increases as the number of trees increases. When number of decision trees is increasing, more conditions are considered, and the process becomes much more complicated, which provides a better result and consumes more time.

In order to achieve a relatively good outcome and efficient performance, random forest classification was performed with five trees. The performance of the classifier reaches almost the accuracy of 98%.

3.2 Comparison and Discussion

Running time: 9.1847

	precision	recall	f1-score	support
1	0.82	0.77	0.79	84633
2	0.83	0.87	0.85	113390
3	0.95	0.99	0.97	14269
4	1.00	0.28	0.43	1065
5	0.97	0.94	0.95	3906
6	0.98	0.98	0.98	6995
7	0.97	0.99	0.98	8147
avg / total	0.85	0.85	0.85	232405

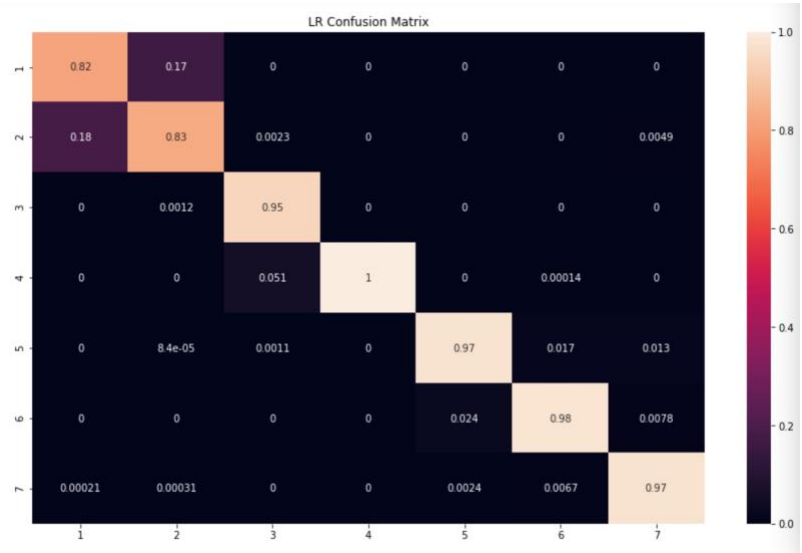


Figure 15. The result table and confusion matrix of Logistic Regression–with normalization

Running time: 7.259

	precision	recall	f1-score	support
1	0.99	0.99	0.99	84633
2	0.99	0.99	0.99	113390
3	0.97	0.98	0.98	14269
4	0.82	0.75	0.78	1065
5	0.99	0.98	0.99	3906
6	1.00	1.00	1.00	6995
7	0.99	1.00	0.99	8147
avg / total	0.99	0.99	0.99	232405

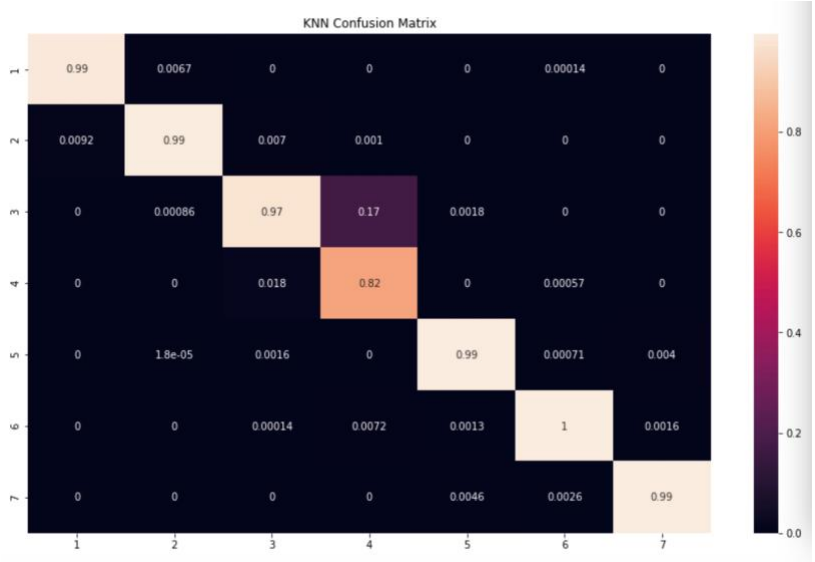


Figure 16. The result table and confusion matrix of KNN– with normalization

Running time: 26.0366

	precision	recall	f1-score	support
0	0.97	0.99	0.98	84633
1	0.99	0.98	0.99	113390
2	0.97	0.97	0.97	14269
3	0.89	0.63	0.74	1065
4	0.99	0.95	0.97	3906
5	0.99	0.99	0.99	6995
6	0.99	0.99	0.99	8147
avg / total	0.98	0.98	0.98	232405

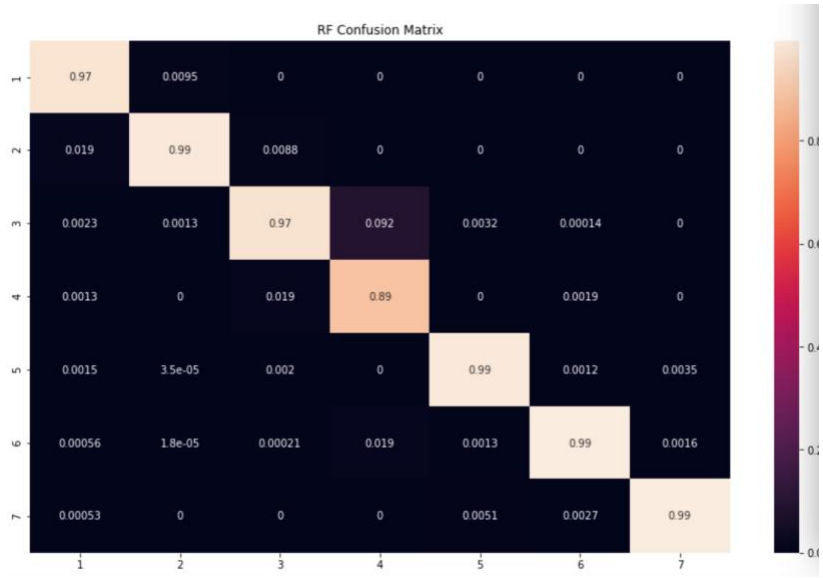


Figure 17. The result table and confusion matrix of Random Forest– with normalization

The confusion matrixes of the three classifiers used are presented above. As we could see that among the three classification methods, KNN has the best accuracy performance by looking at the confusion matrix, which can also be referring to reported result in previous section. As for logistic regression, it has a relatively low accuracy rate compare to the other two methods and it also uses a higher number of PCA.

There is also interesting to find that for the fourth class it has a lower accuracy rate comparing to other classes. However, when we look closer to the data we found that the reason for causing this issue is that for fourth class – Cottonwood/Willow, it has the smallest number of record – 2747, with the total of 581012 records. Another reason might be there is a close similarity between the third class – Ponderosa Pine and the fourth class – Cottonwood/Willow. Due to the similarities between those two and the third class has significantly large number of record of 35754, few of fourth class item were classified as in third class for KNN and Random Forest methods. On the other hand, logistic regression model performs well for classify the type 4 class, but it has relatively low accuracy on classifying between type1 and 2 classes.

After running and comparing the three classification methods, we believe KNN is a better fit for this dataset. First of all, this dataset contains 581012 records, so it is not a very large dataset, in this case using KNN would fulfil the goal of classifying different classes. Secondly, by comparing the result for the three classification methods, KNN has the highest accuracy rate and

with relatively low explained variance ratio, so that it need less information to make accurate classification. Thirdly, running time is also an important measurement, as we mentioned in our result section, KNN method has 7.2 seconds of running time, random forest has 26.0 seconds of running time and logistic regression classifier has 9.1 seconds of running time. Overall, according to the observations and performances, KNN method seems to be perform better than random forest method and logistic regression method on this dataset.

Conclusion and Feature work

In this assignment, we successfully applied machine learning and data mining techniques to generate three types of classifiers to classify the classes in Cover type dataset. We tested logistic regression method, KNN algorithm method and random forest method. In order to improve the performance, we performed data preprocessing as normalization and PCA. Overall, all three of classifiers can have accuracy rate above 85%. Among the three classifiers, KNN seems to be the most accurate and efficient approach to make classification in this dataset.

There are few future work that one could perform on this dataset. Since KNN has such a high accuracy in this assignment when we use k=1, then it may have a chance of become overfitting. In this case, regularization can be performed on KNN classifier to prevent the likelihood of overfitting. A graph to help interpret the process of regularization is shown below:

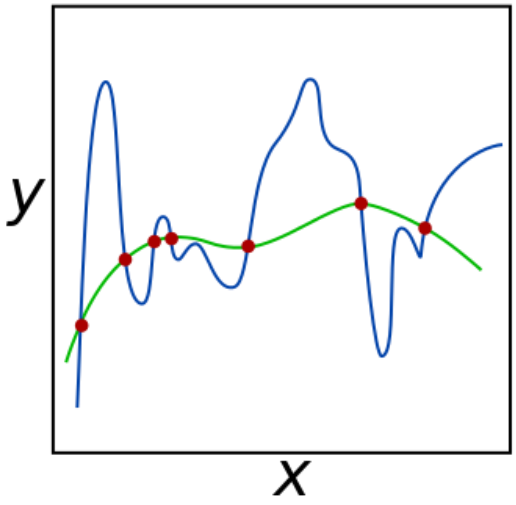


Figure 18. An example diagram of overfitting

For future work, we also suggest performing neural network to this dataset to minimize error and perform a more efficient classification by providing less amount of information.

Reference

[1] T. Kolasa and A. Raja, "Forest Cover Type Classification Study", Rstudio-pubs-static.s3.amazonaws.com, 2018. [Online]. Available: https://rstudio-pubs-static.s3.amazonaws.com/160297_f7bcb8d140b74bd19b758eb328344908.html. [Accessed: 01- Jun- 2018].

[2] H. Hugo, Classifying Forest Cover type with cartographic variables via the Sup- port Vector Machine, Naive Bayes and Random Forest classifiers., 1st ed. Örebro: Orebro University, 1992.

[3] R. Vasudev, "What is One Hot Encoding? Why And When do you have to use it?", Hacker Noon, 2017. [Online]. Available: https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f. [Accessed: 04- Jun- 2018].

[4] V. Spruyt, "The Curse of Dimensionality in Classification", Computer vision for dummies, 2014. [Online]. Available: http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/. [Accessed: 04- Jun- 2018].

[5] L. Smith, "A tutorial on Principal Components Analysis", Cs.otago.ac.nz, 2002. [Online]. Available: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf. [Accessed: 04- Jun- 2018].

[6] S. Holland, "PRINCIPAL COMPONENTS ANALYSIS (PCA)", Strata.uga.edu, 2008. [Online]. Available: https://strata.uga.edu/software/pdf/pcaTutorial.pdf. [Accessed: 04- Jun- 2018].

[7] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", Arxiv.org, 2018. [Online]. Available: https://arxiv.org/abs/1502.03167. [Accessed: 04- Jun- 2018].

[8] D. Abbott, Applied Predictive Analytics. Wiley, 2014, pp. 254-270.

[9] D. Benyamin, "A Gentle Introduction to Random Forests, Ensembles, and Performance Metrics in a Commercial System", Bing.com, 2012. [Online]. Available: http://www.bing.com/cr?IG=70006CA78D16421E87D8EEDB2DE67C09&CID=3D34AC24625068B23E09A7DA63AD69E7&rd=1&h=b5m98MAbg2aNXNhMnBGl3l0f_SpMlCtg5TJ2qmUmK1M&v=1&r=http%3a%2f%2fblog.citizennet.com%2fblog%2f2012%2f11%2f10%2frandom-forests-ensembles-and-performance-metrics&p=DevEx.LB.1,5504.1. [Accessed: 29- May- 2018].

[10] S. Polamuri, "How the random forest algorithm works in machine learning", Dataaspirant, 2017. [Online]. Available: http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/. [Accessed: 29- May- 2018].

[11] J. Brownlee, "Bagging and Random Forest Ensemble Algorithms for Machine Learning", Machine Learning Mastery, 2016. [Online]. Available: https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/. [Accessed: 04- Jun- 2018].

[12] R. Eulogio, "Introduction to Random Forests", Datascience.com, 2017. [Online]. Available: https://www.datascience.com/resources/notebooks/random-forest-intro. [Accessed: 29- May- 2018].

[13] "Covertype Data Set", Archive.ics.uci.edu, 2018. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/covertype. [Accessed: 04- Jun- 2018].

Appendix

Code Instruction

1. Open Jupyter notebook
2. Open the "Assignment2.ipynb"
3. Run the first eight cells, which contain all the necessary external libraries and all the defined functions
4. Load the data

```
# Read the data
data = pd.read_csv("covtype.data",header=None, index_col=False)
```

5. Simply click the run bottom of the rest of the cells to get the results. E.g. After running the cells with comment "KNN classifier" at the beginning, it will provide the KNN classifier analysis
6. To get all the analysis, the simplest way is click Kernel > Restart & Run all

External Libraries

- Scikit-learn
- Pandas

Data set URL

https://archive.ics.uci.edu/ml/datasets/covertype