Data Mining Report

Syed Hassan (shass054) & Shah Ali (Sali059)

Our coursework consisted of creating three machine learning classification Algorithms, which were the following;

* K- nearest neighbour implanted in python and java programming language this was done to experiment the algorithm performance of both languages.
* K- nearest neighbour implanted in java programming language.
* 5Fold cross validation which was implemented in Java.
* Confusion matrix which was also implemented in Java as the programming language.

We used Adult Income Data which was published in 1996, this data set came with test data which had 45222 instances, which consist of 3620 unknown random missing data which was denoted by a “?”. This data set consisted of 15 features:

|  |  |
| --- | --- |
| * Age | * Sex |
| * fnlwgt | * Capital Gain |
| * Education | * Hours per week |
| * Education Num | * Capital Loss |
| * Marital Status | * Native Country |
| * Occupation | * Earns |
| * Relationship | * Working Class |
| * Race |  |

Moreover, the data has only two classes, under the feature name Earns ,which represents whether an individual earns more than £50,000 or less than £50,000 a year; in order to predict whether the income was indeed higher or lower than 50k, we used classification algorithms to predict, using the data point to see if a person earns more than 50k or not, whereby machine learning algorithms tools such as: K Nearest Neighbours, confusion matrix, 5-Fold Cross Validation were implemented.

**Cleaning data**

One of the first important task was to clean and declutter the missing data and any another data point that consisted of no values. For me to clean the data, I firstly removed 3620 instances that contained the symbol “?”, this is because the missing data would have interrupted the prediction rating.

Deleting data point isn’t good practice because every data is valuable Although the missing data only consisted of 6% of the overall data points, deleting it would have not made a massive difference and thus would not cause any disruption to the algorithm outcome. The data was further analysed; and many of the features that were irrelevant to the prediction of an individual earning less or more than £50K were removed. The features that I felt needed removing were;

* **Fnlwgt:** These features held large unnecessary numerical values that played no contribution to an individual’s prediction.
* **Education:** This feature consists of the numerical values that represent the education level that the individuals received. This data was unnecessary as it wouldn’t add any benefit in the prediction value of finding out if individual earning is below or above 50K. There is already an education Num feature which represents the same data, so this was kept and used in the algorithm.
* **Capital Loss and Capital Gain:** This feature was deleted because most of the data contained zero, which ultimately made it useless data and thus no longer required.

**KNN**

For me to implement the K Nearest Neighbour algorithms, I had to convert the features data points that contained categorical data to numerical. The new data set was then saved onto a new CSV file called data.new.csv. I imported Panda Library to import my CSV datasets The CSV data was then converted into a list whereby I was able to split the list into 80% Test Data and 20% Train Data. The two classes were classified as 1; being over 50K, and 0; which represented being under 50K. After this, I created a function called K Neared Neighbour, which takes three parameters; dataset, the prediction and the value of K. I than converted my list into an array using the library Numpy array, once this was done, I used the Euclidean distance formula to calculate the distance as the square root of the sum of the squared differences between the two Numpy arrays of numbers. Which then returns the k most similar neighbours from the training set for a given test instance. Once we have calculated the closest nearest neighbour for a test instance, Ones found be than use the predicted response based on those found neighbours. We we’re able to achieve this by allowing each neighbour to vote for their class attribute and take the majority vote as the prediction.

In the Java program, I had to first read the CSV file using a scanner, then turn it into an array, turn every value including the strings into integers (e.g. female = 0, male = 1), calculate the Euclidean distance between the test set and the training set, sort the distances in ascending order, print all k nearest neighbours and all predicted classes through the use of HashMaps. The HashMap will show the class and the number of its occurrences in the k nearest neighbours. The total number of occurrences should be equal to k.

Below, is the accuracy of predictions in the KNN algorithm.

Python K-NN version:

* K = 5
* Accuracy: 0.77

Java K-NN version:

* K = 1
* Accuracy = 0.76

**5-Fold Cross Validation**

5 Cross Validation is a technique that would be implemented to assess the performance of the machine learning model. The procedure consisted of a single parameter called the “n” which denotes to the number of groups or sections that given data set is to be split into. In this occurrence, the data sample was split into 5 groups, thus why term 5-fold cross validation was used.

In the Java program, I first ensured that the test set would be present in each of the folds after every iteration (e.g. the test set would be in first fold for the first iteration, then go to second fold in second iteration etc.) Then, I computed the accuracy for the test set using confusion matrix with the following formula (TP + TN/TN+FN+TP+FP.) After, it got slightly tricky as I had to sort each of the rows in the training set to the correct fold specified in the ‘folds’ column. Once I had done this, I added the training sets to the correct fold and computed the accuracy via confusion matrix. Then, I computed the average accuracy for each iteration. Finally, I averaged the accuracy for each iteration again, giving me the overall accuracy. I then experimented with different k values (e.g. [1,3,5,7,9,11,…39]), to find the best k. the full results can be found in the grid.results.txt file under src in the project directory.

The method is known to be comprehendible and produces less biased or less optimistic estimate of the model skill.

**Matrix**

In order to test the 5- fold model in its efficiency to perform, the confusion matrices were used to find our accuracy. The original 5-fold results will iterate 5 times. Each time their will be a new “test set” with a different or possibly the same confusion matrix. Same thing applies with “train set” in which each row has its own specific fold. This will continue until the program is able to compute the overall accuracy. Once I found the best k and accuracy in the 5-cross validation, I created a confusion matrix on the test dataset. From this matrix, I was able to compute the sensitivity, precision and specificity of the data. Results can be seen below.

**Accuracy results from a grid of K values (Java):**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| K | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 | 29 | 31 | 33 | 35 | 37 | 39 |
| Accuracy | 0.76 | 0.63 | 0.67 | 0.65 | 0.66 | 0.66 | 0.60 | 0.63 | 0.64 | 0.65 | 0.61 | 0.61 | 0.59 | 0.62 | 0.63 | 0.61 | 0.63 | 0.62 | 0.62 | 0.63 |

Best K results: K= 1 with the accuracy of 0.76

**Confusion Matrix results (Java):**

Test Set Confusion Matrix: [[151, 0], [49, 0]]

Sensitivity: 1.0

precision: 0.76

Specificity: 0.0

Reference:

Website

UCI MACHINE LEARNING REPOSITORY: ADULT DATA SET

In-text: (Archive.ics.uci.edu, 2018)

Your Bibliography: Archive.ics.uci.edu. (2018). UCI Machine Learning Repository: Adult Data Set. [online] Available at: https://archive.ics.uci.edu/ml/datasets/adult [Accessed 2 Oct. 2018].

Website

BROWNLEE, J.

A Gentle Introduction to k-fold Cross-Validation

In-text: (Brownlee, 2018)

Your Bibliography: Brownlee, J. (2018). A Gentle Introduction to k-fold Cross-Validation. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/k-fold-cross-validation/ [Accessed 3 Nov. 2018].

BROWNLEE, J. Website

A Gentle Introduction to k-fold Cross-Validation

In-text: (Brownlee, 2018)

Your Bibliography: Brownlee, J. (2018). A Gentle Introduction to k-fold Cross-Validation. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/k-fold-cross-validation/ [Accessed 3 Nov. 2018].