Dr Dimitrios Letsios
Department of Informatics
King's College London

Semester 2
Monday 10 February 2025

7CCSMDM1 Data Mining

Programming Coursework

Due: Thursday 20 March 2025 (4pm UK time)

The aim of this coursework assignment is to demonstrate understanding of and obtain experience with classification and cluster analysis which are among the most important data mining tasks. It requires efficiently manipulating data sets, building data mining models, and obtaining insights with different types of data. The coursework is worth 20% of the overall module mark and will be marked out of 100 points. The distribution of points is 30 points for the first part on **decision trees with categorical attributes**, 30 points for the second part on **cluster analysis**, and 40 points for the third part on **text mining**. The data sets required for this coursework are provided in the KEATS page of the module. Do not download them from their original sources for your coursework implementation. The links to their origins are provided for referencing purposes. **Instructions** are given at the end of this document. It is important that you carefully follow all coursework instructions because the coursework will be automatically marked.

1 Decision Trees with Categorical Attributes

This part uses the **adult data set** (`https://archive.ics.uci.edu/ml/datasets/Adult`) from the *UCI Machine Learning Repository* to predict whether the income of an individual exceeds 50K per year based on 14 attributes. The attribute *fnlwgt* should be dropped and the following attributes should be taken into consideration:

Attribute	**Description**
age	age group
workclass	type of employment
education	level of education reached
education-num	number of education years
marital-status	type of maritals status
occupation	occupation domain
relationship	type of relationship involved
race	social category
sex	male or female
capital-gain	class of capital gains
capital-loss	class of capital losses
hours-per-week	category of working hours
native-country	country of birth

1. *[10 points]* Load the data set and compute (a) the number of instances, (b) a list with the attribute names, (c) the number of missing attribute values, (d) a list of the attribute names

with at least one missing value, and (e) the percentage of instances corresponding to individuals whose education level is Bachelors or Masters.

2. *[10 points]* Drop all instances with missing values. Convert all input attributes to numeric using one-hot encoding. Name the new columns using attribute values from the original data set. Next, convert the class values to numeric with label encoding.

3. *[10 points]* Build a decision tree and classify each instance to one of the $<= 50K$ and $> 50K$ categories. Compute the training error rate of the resulting tree.

4. *[Optional, Not marked]* After you have finished with the above questions, you may optionally proceed with some additional steps for a more accurate error rate estimation. First, shuffle the rows of the data set. Next, perform N-fold cross-validation, for different values of N. What is a reasonable choice for N?

5. *[Optional, Not marked]* Evaluate the effect of erroneous values in the training data. Perturb a portion $p\%$ of the attribute values in the training set. For this, you may replace an attribute value with another which is randomly selected from the available values for the same attribute. Such perturbations typically modify the underlying data distribution. What is the effect on the classification accuracy?

2 Cluster Analysis

This part uses the wholesale customers data set (`https://archive.ics.uci.edu/ml/datasets/wholesale+customers`) from the *UCI Machine Learning Repository* to identify similar groups of customers based on 8 attributes. The attributes *Channel* and *Region* should be dropped. Only the following 6 numeric attributes should be considered:

Attribute	**Description**
Fresh	Annual expenses on fresh products.
Milk	Annual expenses on milk products.
Grocery	Annual expenses on grocery products.
Frozen	Annual expenses on frozen products.
Detergent	Annual expenses on detergent products.
Delicatessen	Annual expenses on delicatessen products.

1. *[10 points]* Compute the mean, standard deviation, minimum, and maximum value for each attribute. Round the mean and standard deviation to the closest integers.

2. *[20 points]* Divide the data points into k clusters, for $k \in \{3, 5, 10\}$, using kmeans and agglomerative hierarchical clustering. Because the performance of kmeans, namely the computed solution quality and number of iterations, is significantly affected by the initial cluster center selection, repeat 10 executions of kmeans for each k value. Next, standardize each attribute value by subtracting with the mean and then dividing with the standard deviation for that attribute. Repeat the previous kmeans and agglomerative hierarchical clustering executions with the standardized data set. Identify which run resulted in the best set of clusters using the Silhouette score as your evaluation metric. Visualize the best set of clusters computed in the previous question. For this, construct a scatterplot for each pair of attributes using Pyplot. Therefore, 15 scatter plots should be constructed in total. Different clusters should appear with different colors in each scatter plot. Note that these plots can be used to manually assess cluster separation.

3. *[Optional, Not marked]* The kmeans++ algorithm attempts to improve the kmeans initialization process by randomly sampling initial cluster centers which tend to be spread out. Evaluate how the quality of the resulting set of clusters is affected if kmeans++ is employed.

3 Text Mining

This part uses the **Coronavirus Tweets NLP** data set from Kaggle `https://www.kaggle.com/datatattle/covid-19-nlp-text-classification` to predict the sentiment of Tweets relevant to Covid. The data set (**Corona_NLP_test.csv** file) contains 6 attributes:

Attribute	**Description**
UserName	Anonymized attribute.
ScreenName	Anonymized attribute.
Location	Location of the person having made the tweet.
TweetAt	Date.
OriginalTweet	Textual content of the tweet.
Sentiment	Emotion of the tweet.

Because this data set is quite big, use vectorized (pandas) operations to effectively perform the various tasks with a typical personal computer. In this way, you will be able to run your code in few seconds. Otherwise, running your code might require a significant amount of time, e.g. in the case where *for loops* are used for accessing all elements of the data set. Further, use raw Python string functions for text processing operations.

1. *[13 points]* Compute the possible sentiments that a tweet may have, the second most popular sentiment in the tweets, and the date with the greatest number of extremely positive tweets. Next, convert the messages to lower case, replace non-alphabetical characters with whitespaces and ensure that the words of a message are separated by a single whitespace.

2. *[14 points]* Tokenize the tweets (i.e. convert each into a list of words), count the total number of all words (including repetitions), the number of all distinct words and the 10 most frequent words in the corpus. Remove stop words, words with ≤ 2 characters, and reduce each word to its stem. Recompute the 10 most frequent words in the modified corpus. What do you observe?

3. *[Optional, Not marked]* The previous steps allow analyzing the data set for extracting useful information from the tweets. Plot a histogram with word frequencies, where the horizontal axis corresponds to words, while the vertical axis indicates the fraction of documents in a which a word appears. The words should be sorted in increasing order of their frequencies. Because the data set size is quite big, use a line chart instead of a histogram. In what way this plot can be useful for deciding the size of the term document matrix? How many terms would you add in a term-document matrix for this data set?

4. *[13 points]* This task can be done individually from the previous three. Store the *coronavirus_tweets.csv* corpus in a numpy array and produce a sparse representation of the term-document matrix with a CountVectorizer. Next, fit a Multinomial Naive Bayes classifier to the data. What is the classifier's training accuracy? A CountVectorizer allows limiting the range of frequencies and number of words included in the term-document matrix. Appropriately tune these parameters to achieve the highest classification accuracy you can.

Instructions

- For your project implementation, use the template files *adult.py*, *wholesale_customers.py*, and *coronavirus_tweets.py*. Each of these files corresponds to one part of the coursework and contains a list of empty function bodies that you are expected to fill.
- Do **not** modify the template file names or the names of the functions therein.
- Do **not** convert the template files into another format, e.g. .ipynb (iPython Notebook).
- Do **not** add any source code outside the given function bodies except for import statements.
- The input and output of each function should be clear after reading this coursework description and the template file comments.
- In your function implementations, do **not** modify the data sets in ways that are not indicated in the coursework instructions provided in this document and the python files.
- Your coursework submission should be a zipped folder containing only the three template python files with your function implementations. It should be entitled using your (a) first name, (b) last name, (c) student number in the form *firstname_lastname_studentnumber.zip*. The student number should consist of only numeric digits.
- Do **not** include any other files in your submission, e.g. csv files or plots.
- Your coursework will be marked using a Python script that will import the functions in your submitted Python files. It is important that you follow the guidelines in your function implementations, e.g. the specified inputs and outputs.
- The marking process will apply your function implementations to variants of the provided data sets. These variants will have the same attributes and attribute values with the original data, but the instances will be different.
- Manipulate the data using pandas data frames and pandas series (unless specified otherwise). Build data mining models using scikit-learn.
- Implement this coursework on your own.