INFS7203 Project Proposal

Jaskeerat Singh
School of Information Technology and Electrical Engineering
The University of Queensland, Qld., 4072, Australia

1 Introduction

The project aims to use different data mining techniques on an imperfect dataset of E Coli bacteria present in CSV format. The overall objective is to design a classifier with a good level of generalisation to differentiate whether the test data genes have cell communication.

2 Dataset

This project will utilise the E. coli dataset, which consists of 1500 tuples of gene data. There are a total of 107 columns of data in the data. The first 103 columns are numerical features, and the following three are nominal features that define the expression level of each record. The last column is the target column, with the label indicating the cell communication. The positive class is denoted as 1, and the negative class is 0.

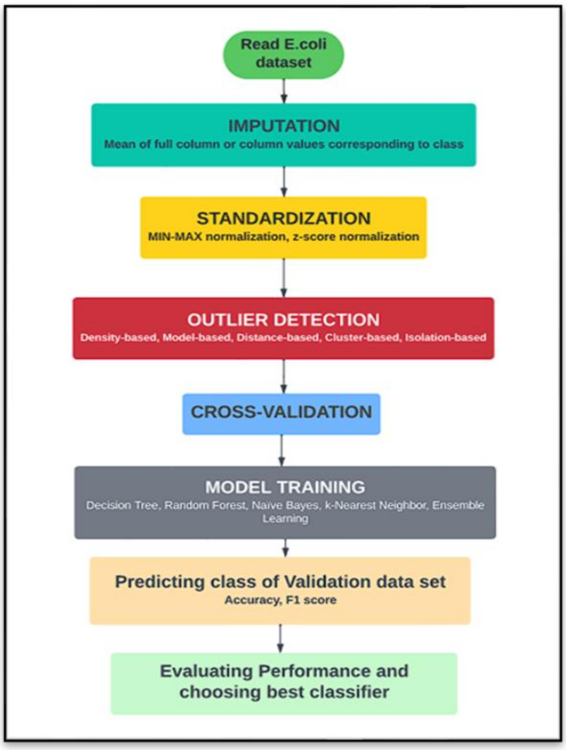

Read E.coli dataset
IMPUTATION
Mean of full column or column values corresponding to class
STANDARDIZATION
MIN-MAX normalization, z-score normalization
OUTLIER DETECTION
Density-based, Model-based, Distance-based, Cluster-based, Isolation-based
CROSS-VALIDATION
MODEL TRAINING
Decision Tree, Random Forest, Naive Bayes, k-Nearest Neighbor, Ensemble Learning
Predicting class of Validation data set
Accuracy, F1 score
Evaluating Performance and choosing best classifier

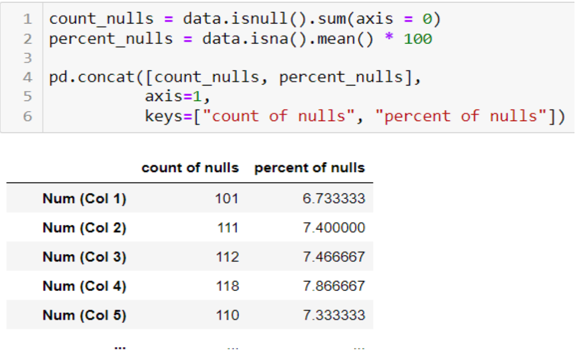
Fig 1. Process of Classification

3 Methodology

The main steps included in designing the methodology for classification include data pre-processing (imputation, normalisation, outlier pruning and feature selection), cross-validation (training and validating), and evaluation of the classifier models based on performance (Accuracy, F1, Recall). The classifier will be chosen based on performance to classify the class label for the test data.

4 Data Pre-processing

Data pre-processing is the process of taking raw data and transforming it into processed information. Unprocessed data often leads to incorrect results. Hence, pre-processing will be important for cleaning the data, removing outliers and feature selection. The dataset provided for classification has nulls in various columns. Furthermore, we can also see some deviation in the data, which may lead to outlier cases.

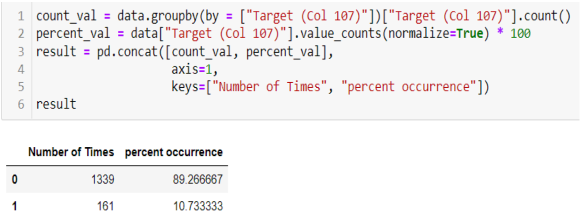


```
count_nulls = data.isnull().sum(axis = 0)
percent_nulls = data.isna().mean() * 100

pd.concat([count_nulls, percent_nulls],
          axis=1,
          keys=["count of nulls", "percent of nulls"])
```

	count of nulls	percent of nulls
Num (Col 1)	101	6.733333
Num (Col 2)	111	7.400000
Num (Col 3)	112	7.466667
Num (Col 4)	118	7.866667
Num (Col 5)	110	7.333333
...	...	...

Fig 2. Count and Percent of Nulls in the data

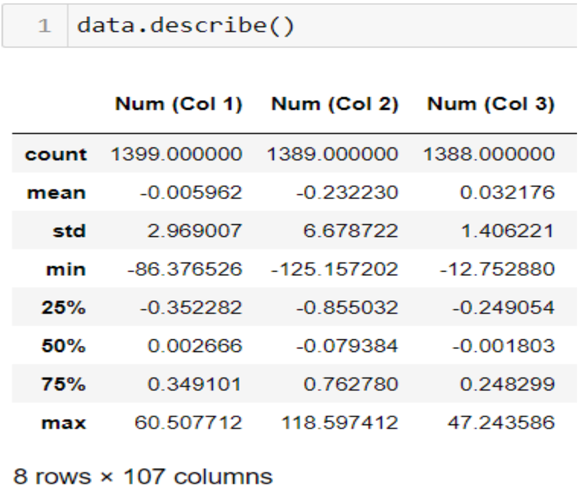


```
count_val = data.groupby(by = ["Target (Col 107)"])["Target (Col 107)"].count()
percent_val = data["Target (Col 107)"].value_counts(normalize=True) * 100
result = pd.concat([count_val, percent_val],
                   axis=1,
                   keys=["Number of Times", "percent occurrence"])
result
```

	Number of Times	percent occurrence
0	1339	89.266667
1	161	10.733333

Fig 3. Percentage of Occurrence of the target class

Imputation for the missing values in each column will be carried out by replacing nulls and blank values by either the mean of the whole column or by the mean of the column based on the label class. Based on the count of each unique value in the target column, we can see that 0 occurs approximately 90% of the time. Hence, the method to impute value based on the class label will be preferred.



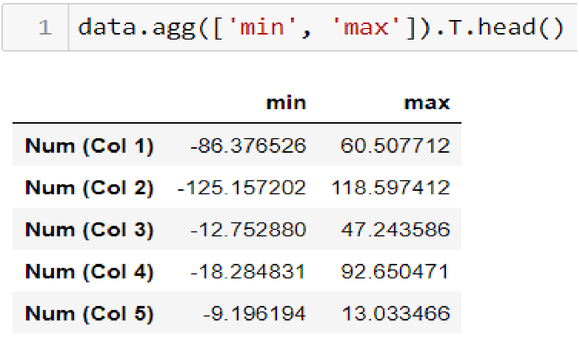
```
data.describe()
```

	Num (Col 1)	Num (Col 2)	Num (Col 3)
count	1399.000000	1389.000000	1388.000000
mean	-0.005962	-0.232230	0.032176
std	2.969007	6.678722	1.406221
min	-86.376526	-125.157202	-12.752880
25%	-0.352282	-0.855032	-0.249054
50%	0.002666	-0.079384	-0.001803
75%	0.349101	0.762780	0.248299
max	60.507712	118.597412	47.243586

8 rows × 107 columns

Fig 4. Example Description for first 3 columns

Anomaly detection will be carried out by comparing different techniques like density-based, statistical model-based, cluster-based techniques etc., on the features of the dataset. Based on the performance of the different techniques, the best method will be chosen. For our classification analysis, we will use the statistical-based method, which assumes that the data is following gaussian distribution and outliers are the attributes that lie outside the standard deviation from the mean.

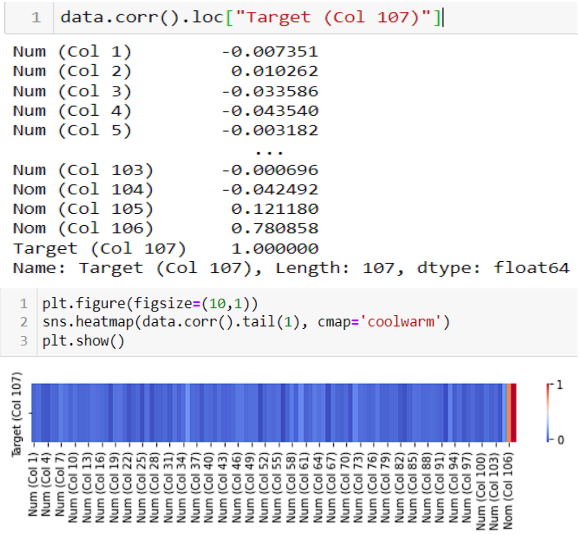


```
data.agg(['min', 'max']).T.head()
```

	min	max
Num (Col 1)	-86.376526	60.507712
Num (Col 2)	-125.157202	118.597412
Num (Col 3)	-12.752880	47.243586
Num (Col 4)	-18.284831	92.650471
Num (Col 5)	-9.196194	13.033466

Fig 5. Range of values for the first five columns

Different Data features have different degrees of magnitude where the scale of values can be different. Depending on the distribution of the data, data normalisation techniques will be applied to various features of the data. Scores of max-min normalisation which bounds the value between [0,1] and applied to data when the distribution of the data is unknown. z-score normalisation does not bound value to [0,1] and the data needs to follow Gaussian distribution. For these reasons, min-max distribution will be preferred.



```
data.corr().loc["Target (Col 107)"]
```

```
Num (Col 1)        -0.007351
Num (Col 2)         0.010262
Num (Col 3)        -0.033586
Num (Col 4)        -0.043540
Num (Col 5)        -0.003182
                      ...
Num (Col 103)      -0.000696
Nom (Col 104)      -0.042492
Nom (Col 105)       0.121180
Nom (Col 106)       0.780858
Target (Col 107)    1.000000
Name: Target (Col 107), Length: 107, dtype: float64
```

```
plt.figure(figsize=(10,1))
sns.heatmap(data.corr().tail(1), cmap='coolwarm')
plt.show()
```

Fig 6. Correlation of target column with other features

The curse of dimensionality states that an increase in the feature set leads to an exponential increase in computational efforts to generalise the given data. Hence, feature selection is used to find the best set of features that shows the most correlation in defining the class of a function and will be only used in the classification process.

5 Cross-Validation

K-fold Cross-validation will be used to assess the score of the classifier model being trained. Here the data will be divided into k-sets, where the k-1 sets will serve as the training data and the last set as the validation set. This training data will be used to map the features of the dataset to their corresponding label class. A validation set will be used to measure the performance of the classifier model used.

We will choose the combination of the imputation method, outlier method and normalisation technique for pre-processing. After processing, the data will be divided and post-classification of the data a score will be produced. Depending on these classification scores of k-sets, an average score will be calculated. Based on these average scores, different methods for pre-processing and classification will be decided. This k-fold method will help in reducing bias in the classification system as every data point will be treated as part of the training and validation set.

A large value of K in K-fold cross-validation will lead to a larger processing time, and a small value will not give an optimal result. Hence, we choose the value of K in which the validation set forms 10% of the total data. Therefore, K = 10.

6 Classification Models

The classification model will be developed on the processed E. coli dataset. Various classifier models including a decision tree, random forest, k-nearest neighbour, Naive Bayes and ensemble classifier will be used to classify the class of the validation set.

A. Decision Tree

The decision tree is a supervised learning model. It produces a flowchart, upside-down tree structure. The structure consists of interconnected nodes where the internal node is also the parent node which is connected to various leaf nodes. The leaf nodes are also called decision nodes.

The hyperparameter required for defining the classifier is the purity level for splitting a node or assigning a class label. To find the purity level, various methods like Information gain (Higher is better), Gain ratio (Higher is better) or Gini index (lower is better) are considered. Here the entropy is considered before splitting a node. To reduce the risk of the classifier performing overfitting [3], pruning methods like pre-pruning (during construction) or post-pruning (after construction) will be performed using different criteria like the max-depth, weighted split, size of node leaf etc.

As the number of features is huge, we cannot build a normal decision tree. Instead, cross-validation is used where CV chooses to train-test split the data and use iterations to produce the best evaluation score under the user-defined conditions like max-depth, max-leaf nodes, min-entropy etc.

B. Random Forest

In random forest, a large number of decision trees are created. Each decision tree will output a class value prediction. Here a majority vote takes place for the predicted class values where the class value with the highest votes becomes the output class value for the random forest. Random forest performs better than decision trees as the combined result of many classifiers is greater than that of one classifier. Like decision tree, purity level using various methods like Information gain, gain ratio or Gini index is considered. The decision trees in the random forest randomly choose a sample of data and calculate the results. The combination of these results is the final output for the random forest. Hyperparameter used in the classifier is the number of decision trees to be used. To determine the optimal number of trees to be used, Various iterations of the k-fold cross-validation are performed. Other parameters like max-depth, and min-samples-split, are considered and the best average score will define the combination of parameters to be used.

C. Naive Bayes

Naive Bayes classifier is a probability-based classifier that is based on Bayes Theorem. It works on the assumption that each predictor feature has made an independent and equal contribution to the outcome. The classifier calculates the probability of a class for a given set of features. Hence it shows the probability of a class value for a combination of different features.

$P(y \mid X) = P(X \mid y) . P(y) / P(X)$
$X = (x_1, x_2, x_3, \ldots\ldots, x_n)$

The variable **y** is the class variable. x_1, $x_{2,}$ x_3 …. x_n represent the feature list on which the class - y is defined.

For the gaussian Naive Bayes classifier, we can see that the features in the dataset are continuous in nature and not discrete. Hence the Gaussian version of Naïve Bayes will be applied for classification.

D. k-Nearest Neighbour

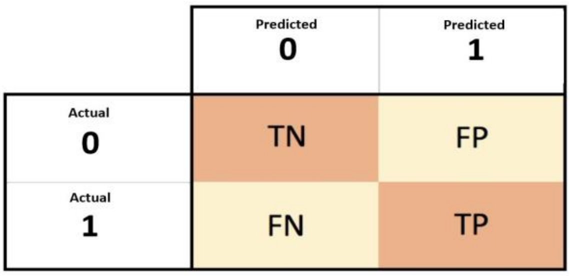
knn works under the assumption that similar things are close to each other. The classifier learns the mapping of the features to their label in the training phase and uses feature similarity (how closely the features of testing data resemble training data) from the test set to predict class values. There are various distance metrics which can be used like Manhattan distance, Euclidean distance and Chebyshev distance to calculate the similarity between the features. The value of hyperparameter k needs to be defined which is a variable that chooses the vote majority class value from its k (1, 3, 5, 7, ... n) nearest neighbours. The value of k is always taken as odd to avoid a tie-in vote majority results. The process of cross-validation is used where the method uses various combinations of increasing k values [1] and different distance metrics to obtain an accuracy score against a validation set.

E. Ensemble learning

Ensemble learning [4] is a classifier that combines the results of various classifier models to predict the class value. Here the result of multiple classifiers will be better in predicting the value in comparison to an individual classifier. The majority class will be voted as the class chosen by the ensemble classifier.

7 Performance Evaluation

Evaluating the performance of the classifier is an important step for validating the quality of the trained classifier model. After the model has been trained with the testing data, we will use the validation set to test the quality of the predicted classes. This will help us in ranking the different classifier models based on their scores.



	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Fig 7. Confusion Matrix

Accuracy = TN+TP / TN+TP+FP+FN

Precision = TP / TP +FP

Recall = TP / TP+FN

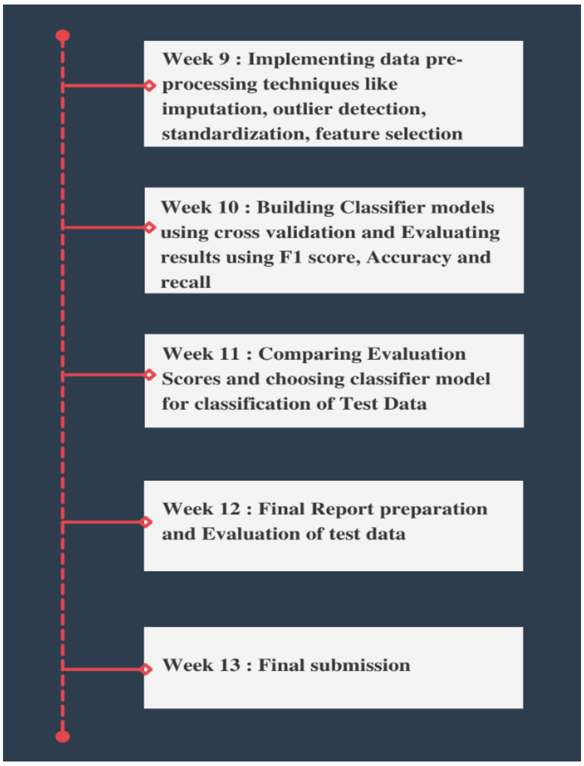
F1 = 2 / precision^{-1} + recall^{-1}

For evaluation, we will check 2 different scores - Accuracy and F1 score [2]. Accuracy is defined as the (number of correct class predictions) / (number of total classes). Accuracy is not the most appropriate metric if some classes have a very low proportion of data compared to other classes. In such a case F1 score needs to be calculated. To calculate the F1 score, a confusion matrix [5] (fig. 7) needs to be developed which consists of 4 different values - True positive (TP), False positive (FP), False Negative (FN), and True Negative (TN). F1 score is defined as the harmonic mean of precision and recall, where precision is defined as the rate of those that are truly positive and recall is defined as the rate of those predicted as positive.

During pre-processing, it was observed that there is an imbalance between the classes in the target column. Hence F1 score will be used to assess the performance of the classifiers. Using Cross-Validation for k-1 sets to train a classifier the last validation set is tested and an F1 score is presented.

The average of these scores will determine the classifier to be used against the test data. Recall can also serve as the metric to decide the final classifier as it shows the number of actual positive cases. Hence, F1 and followed by recall can serve as the metric to determine the best classifier.

Project Implementation Timeline



Week 9 : Implementing data pre-processing techniques like imputation, outlier detection, standardization, feature selection

Week 10 : Building Classifier models using cross validation and Evaluating results using F1 score, Accuracy and recall

Week 11 : Comparing Evaluation Scores and choosing classifier model for classification of Test Data

Week 12 : Final Report preparation and Evaluation of test data

Week 13 : Final submission

Fig 8. Project Implementation Timeline

References

[1] R. Wazirali, "An Improved Intrusion Detection System Based on KNN Hyperparameter Tuning and Cross-Validation," Arab J Sci Eng 45, pp. 10859–10873, 2020.

[2] Reda Yacouby and Dustin Axman, "Probabilistic Extension of Precision, Recall, and F1 Score for More Thorough Evaluation of Classification Models," Association for Computational Linguistics, pp. 79–91, 2020.

[3] Galathiya, A. S., A. P. Ganatra, and C. K. Bhensdadia, "Improved decision tree induction algorithm with feature selection, cross validation, model complexity and reduced error pruning," International Journal of Computer Science and Information Technologies, pp. 3427-3431, 2012.

[4] E. Lutins, "Ensemble methods in machine learning: What are they and why use them?," Medium. https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f, 2017.

[5] M. Zuhaib, "Demystifying the confusion matrix using a business example," Medium. https://towardsdatascience.com/demystifying-confusion-matrix-29f3037b0cfa, 2019.