L-Università ta' Malta

Machine Learning 2022 Documentation

Faculty of Information & Communication Technology

ICS3206 - Machine Learning, Expert Systems, and Fuzzy Logic

Matteo Sammut 0206002L

Algorithms

Decision Tree Classifier

Decision trees are a type of supervised learning algorithm that can be used for both classification and regression tasks, because the data passed is statistical this is a regression tree. The algorithm starts with a single root node, which represents the entire dataset. From this root node, the algorithm repeatedly splits the data into subsets based on certain conditions or "decision points" until it reaches a leaf node, which represents a prediction or a class label, these decision points may be data points such as the pitch in a voice. Each internal node of the tree represents a feature or an attribute of the data, and each branch represents the possible values of that feature. The tree is constructed by iteratively selecting the most informative feature to split the data on at each node, this is known as "splitting" and is based on some criterion. We tested "gini","log_loss" and "entropy", different criterion that measure the quality of the split.

We can also vary between splitting on the "best" or "random[ly]" the best split refers to the split that results in the highest reduction in impurity for the given feature and threshold. A random split is not completely random, on the other hand it refers to a split selected randomly among the best splits. This is used to increase randomness and reduce the risk of overfitting in the decision tree. The "random_state" parameter dictates the randomness when a feature is permutated, all features are randomly permutated in a decision tree. Setting this variable should not affect the results as it only determines the seed of the random number generator to be used, it still made it a valid contester for testing.

This makes Decision Trees useful for their interpretability and ability to handle both categorical and numerical data.

Artificial Neural Network

For this section the Keras package was used for this part of the assignment. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow. It allows for easy and fast prototyping through user friendliness, modularity, and extensibility. It supports both convolutional networks and recurrent networks, as well as combinations of the two.

In the code, Keras is used to build an artificial neural network (ANN) for predicting the sex of speakers based on the voice data provided in the "voice.csv" file. The ANN model is created using the Sequential class from Keras, which allows for the creation of a linear stack of layers. The layers used in this model include a Dense layer, each neuron in the dense layer receives input from all the neurons in the previous layer, and a 'relu' activation function. The final Dense layer is 1 unit, this final layer is used for binary classification, where the output will be a probability representing the likelihood that the input belongs to either sex.

The model is then trained using the fit method, which takes the training data and trains the model for a set number of epochs using a batch size. Once the model is trained, it can then be used to make predictions on the test data using the predict method. The output of this method is thresholded to either male or female depending on the binary probability it returns.

The model's performance is evaluated using several metrics such as accuracy_score, classification_report, and confusion_matrix. There are also several parameters that can be adjusted to improve its performance and fine-tune its behaviour.

The number of layers and neurons in an ANN can improve its ability to learn complex patterns in the data, however, increasing these parameters also increases the risk of overfitting. To prevent overfitting, one can make use of dropout, dropout is a technique used to prevent overfitting by randomly dropping out some neurons during training. One disadvantage is that this methodology could lead to very high diminishing returns, so depending on the graphs a slightly less accurate ANN way be more ideal to one that uses way more resources. Epochs represent the number of times the ANN will be exposed to the dataset, it too can result in overfitting, it would be more likely that we are able to find the sweet spot with such a parameter.

Another parameter that defines how the neural network function is the activation function. Activation functions are mathematical operations applied to the output of a neuron in an artificial neural network, these are applied on at a per layer level to the ANN. There were three activation functions that were tested in the code, ReLu (Rectified Linear Unit), Sigmoid and Softmax. ReLu is used in the hidden layers of the network and helps to introduce non-linearity in the model and is quite computationally efficient. Sigmoid is another common activation function that maps its input to values between 0 and 1, which is useful for binary classification. Softmax is also an activation function that is used in the output layer of a multi-class classification problem and returns a probability distribution over the classes. It normalizes the output of the last layer into a probability distribution over all the classes. By all means ReLu should be the activation function of choice as it is the most tailor made for hidden layers.

The Adam optimizer is a commonly used optimization algorithm in deep learning that adjusts the learning rate adaptively for each parameter during training. The learning rate is a hyperparameter included in such optimizers that determines the step size at which the optimizer makes updates to the model parameters. A smaller learning rate may result in slower convergence but a more accurate model, while a larger learning rate may converge quickly but result in a less accurate model. The Adam optimizer and the learning rate are both important factors in the training of a neural network, as they determine how quickly and accurately the model learns from the training data. However, despite other optimizers existing (SGD, RMSProp, Adamax, Nadam, etc.) the focus was put on the learning rate, mainly to cut down on the number of hyperparameters to account for.

The batch size was another parameter that could have been accounted for but was omitted, the batch size is a hyperparameter that controls the number of samples that are processed before the ANN's weights are updated. It can affect the ANN's performance and training time, a larger batch size can lead to faster training, but it can also cause the model to converge to a suboptimal solution. A smaller batch size can lead to more accurate model, but it can also slow down the training process. It's a trade-off between speed and accuracy, but is more important when training more resource heavy ANNs such as object detectors. Therefore, the optimizing of this parameter was deemed less important than the ones mentioned above. It is important to note that the more hyperparameters that we optimize for, the more trials would need to take place to account for the increase in possible parameter combinations.

Nevertheless, when graphing the above parameters, it was important that rigorous scientific testing be pursued. With the final aim of this testing is to find a set of ideal parameters for the dataset, it was important to not test for every hyperparameter as this would exponentially increase the number of trials that would need to be performed. Rather an emphasis was put on select parameters that could yield the highest performance increase.

Logistic Regression

Logistic regression is a simple model, it uses a logistic function to model the probability of a given voice sample belonging to one of the two sexes. The hyperparameters of a logistic regression classifier, such as the regularization strength and the algorithm, can be adjusted to improve its performance.

One of the main hyperparameters of is the "solver" parameter (algorithm). This parameter defines the algorithm used to find the optimal solution of the logistic regression model, it can make use of a number of algorithms such as "newton-cg", "saga" and "sag". Each solver has its own advantages and disadvantages, so varying between them is key to finding the optimal hyperparameters.

K-Nearest Neighbours Classifier

A k-nearest neighbours (KNN) classifier is a type of algorithm that can be used for classification tasks. KNN classifiers make predictions based on the similarity of new data points to the points in the training dataset, from k neighbours of a datapoint are female then an educated guess can be made that it too is female. The number of nearest neighbours, "k", is a hyperparameter that can be tested and adjusted.

When a new voice data point is encountered, the KNN classifier would find the k closest points in the training dataset to the new point, based on a chosen distance metric, “uniform” or “distance”, those k points would be used as the prediction for the new point. Uniform means that all points factor equally when determining a new datapoint, while distance gives more weight to neighbours close to the datapoint. “p” determines whether Manhattan distance or Euclidean distance is used.

One advantage of using a KNN classifier for this problem is that it is a simple and interpretable model, which makes it easier for us to understand how it is making predictions. It is well suited for datasets such as these where the clusters are evenly split.

Support Vector Machine Classifier

The main idea behind a SVM classifier is to find the best boundary (or "hyperplane") that separates the different classes in the data. This boundary is chosen such that it maximizes the margin between the different classes, which helps to reduce the chances of misclassification.

Once the SVM classifier is trained, it can be used to make predictions on new data. The prediction process is similar to the ANN model, where the classifier would return the probability of the input belonging to a particular class. There are several different SVM kernel functions that can be used during training, such as linear, polynomial, radial basis function (RBF) and sigmoid. We can expect functions that are linear to function marginally better due to the nature of the dataset.

Evaluation

Results

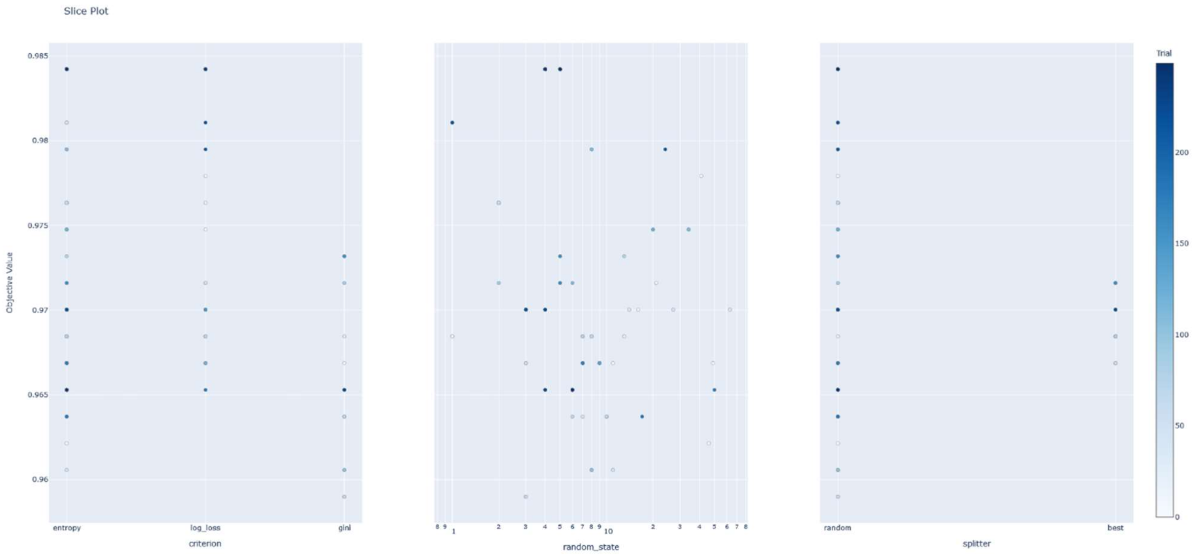


Figure 1:Decision Tree

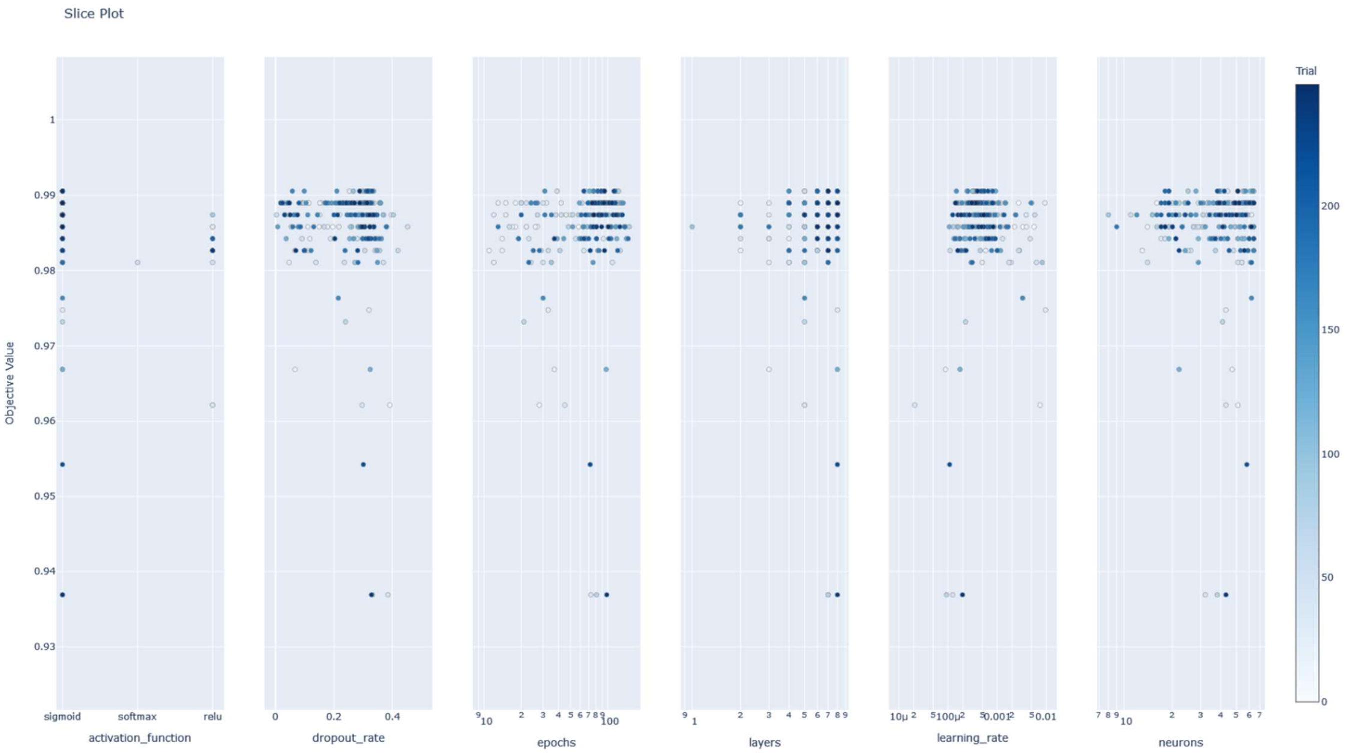


Figure 2: ANN

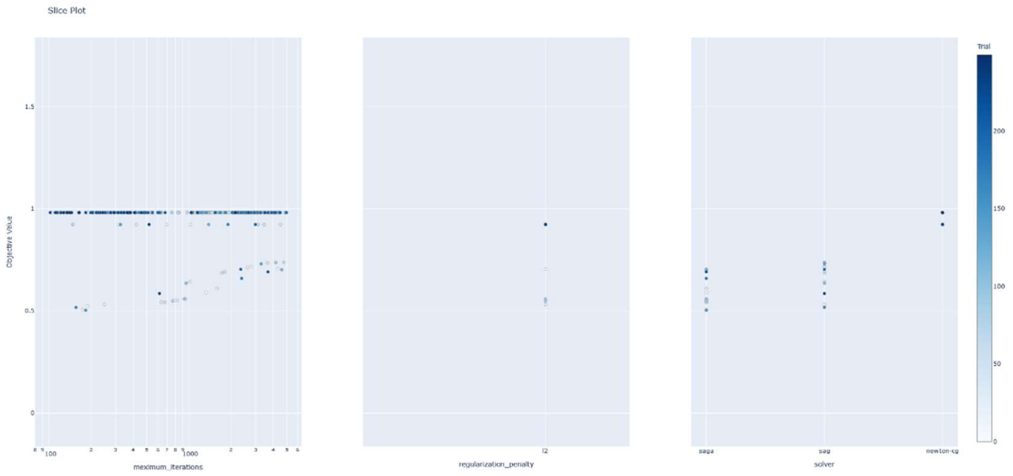


Figure 1:Log Regression

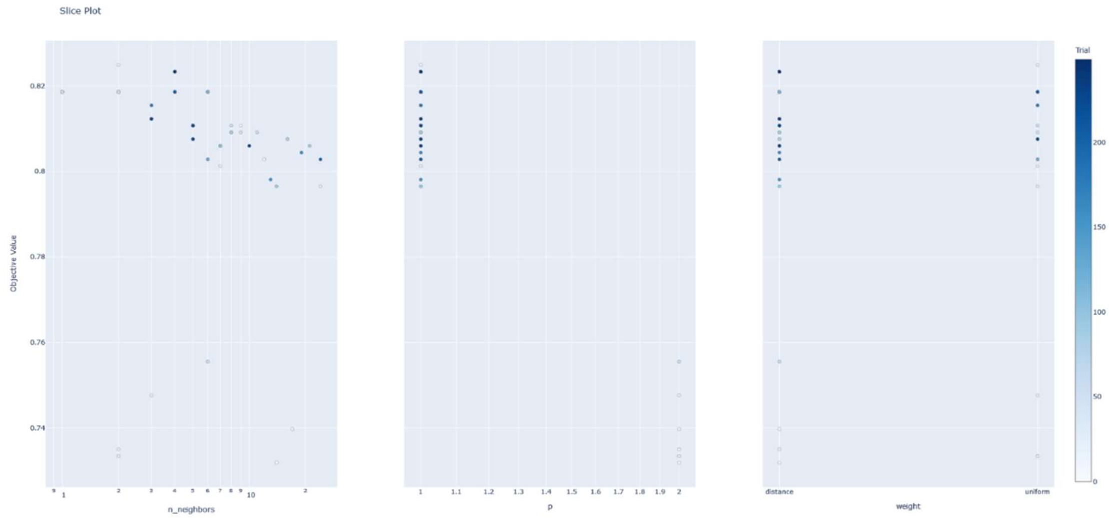


Figure 2:K Nearest Neighbour

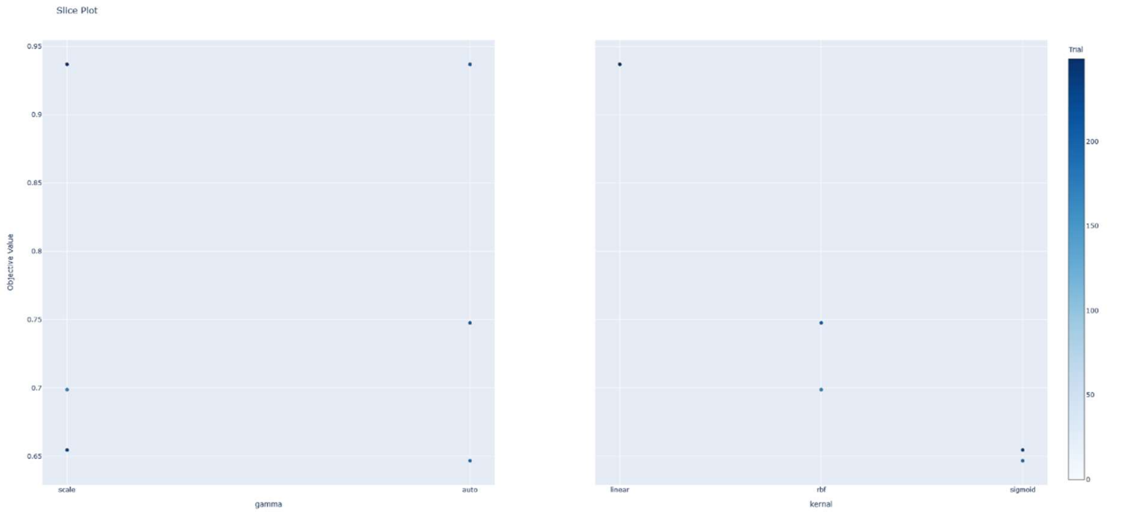
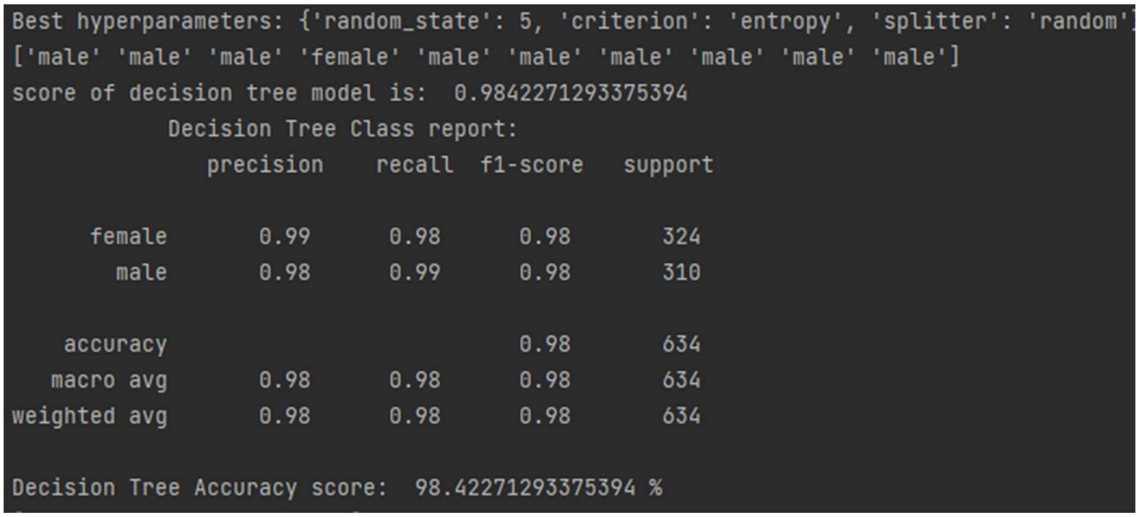


Figure 3:Support Vector Machine



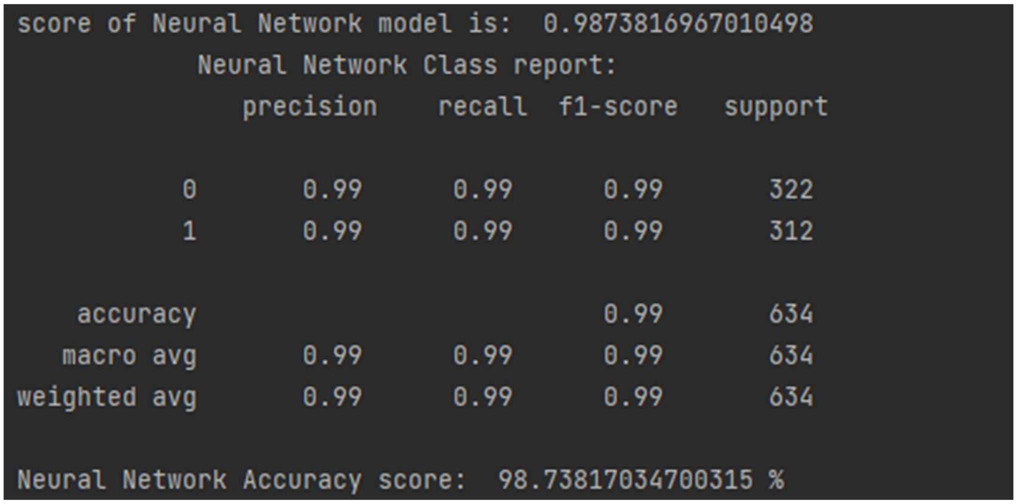
```
Best hyperparameters: {'random_state': 5, 'criterion': 'entropy', 'splitter': 'random'
['male' 'male' 'male' 'female' 'male' 'male' 'male' 'male' 'male' 'male']
score of decision tree model is:  0.9842271293375394
            Decision Tree Class report:
               precision    recall  f1-score   support

      female       0.99      0.98      0.98       324
        male       0.98      0.99      0.98       310

    accuracy                           0.98       634
   macro avg       0.98      0.98      0.98       634
weighted avg       0.98      0.98      0.98       634

Decision Tree Accuracy score:  98.42271293375394 %
```

Figure 4: DecTree Results



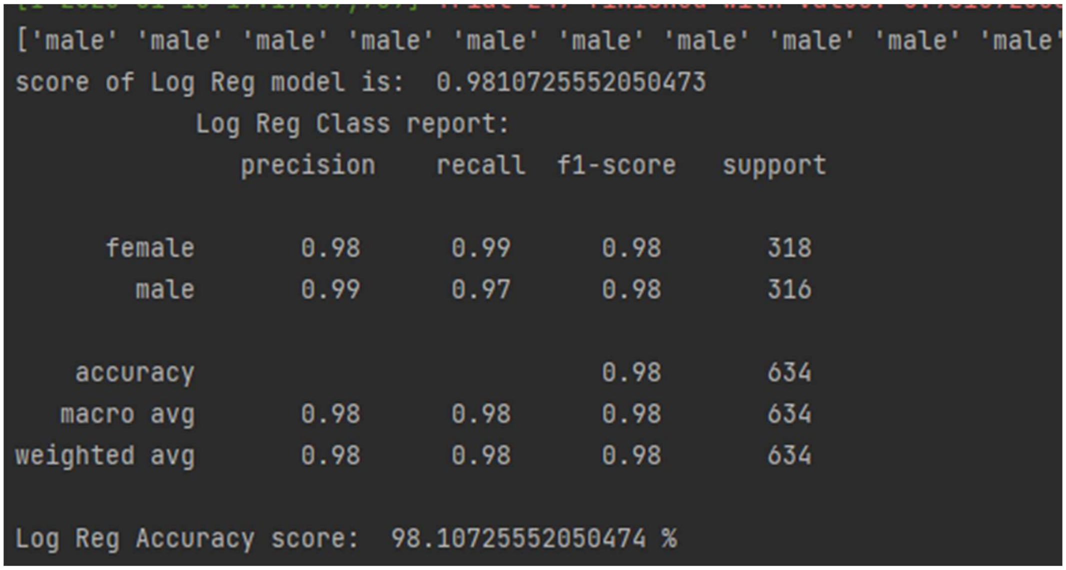
```
score of Neural Network model is:  0.9873816967010498
            Neural Network Class report:
               precision    recall  f1-score   support

           0       0.99      0.99      0.99       322
           1       0.99      0.99      0.99       312

    accuracy                           0.99       634
   macro avg       0.99      0.99      0.99       634
weighted avg       0.99      0.99      0.99       634

Neural Network Accuracy score:  98.73817034700315 %
```

Figure 5:ANN Results



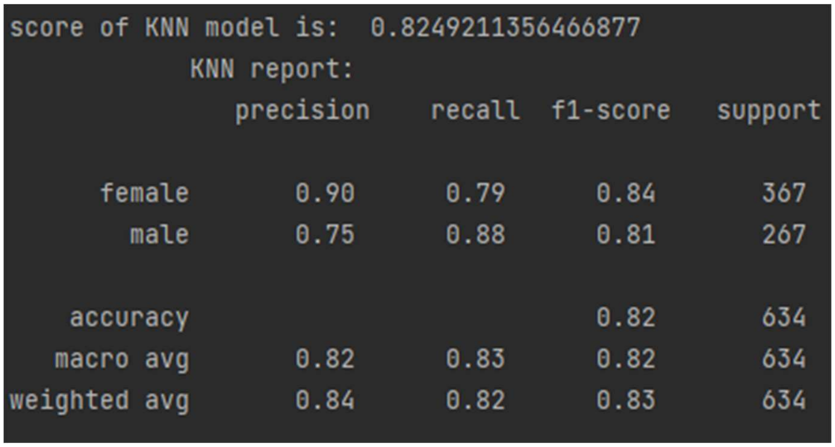
```
['male' 'male' 'male' 'male' 'male' 'male' 'male' 'male' 'male' 'male'
score of Log Reg model is:  0.9810725552050473
            Log Reg Class report:
               precision    recall  f1-score   support

      female       0.98      0.99      0.98       318
        male       0.99      0.97      0.98       316

    accuracy                           0.98       634
   macro avg       0.98      0.98      0.98       634
weighted avg       0.98      0.98      0.98       634

Log Reg Accuracy score:  98.10725552050474 %
```

Figure 6:Log Reg results

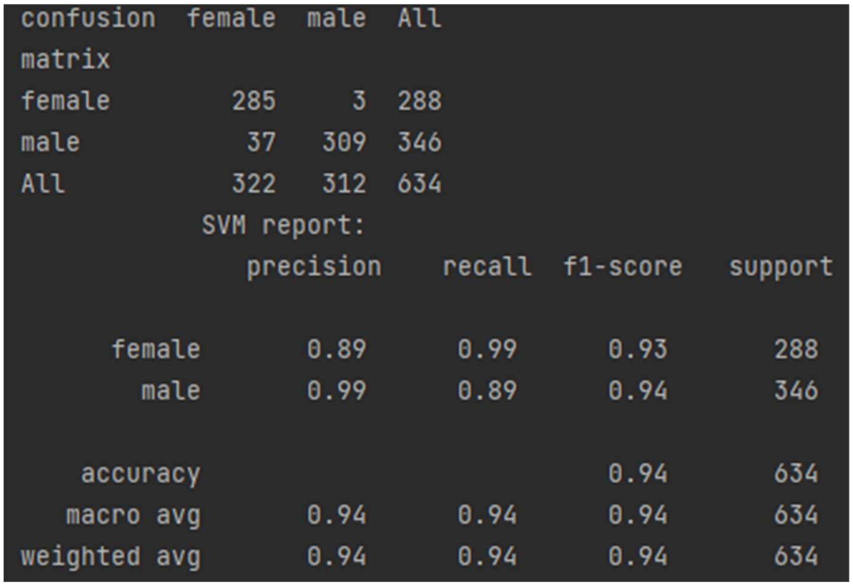


```
score of KNN model is:  0.8249211356466877
            KNN report:
               precision    recall  f1-score   support

      female       0.90      0.79      0.84       367
        male       0.75      0.88      0.81       267

    accuracy                           0.82       634
   macro avg       0.82      0.83      0.82       634
weighted avg       0.84      0.82      0.83       634
```

Figure 7:KNN Results

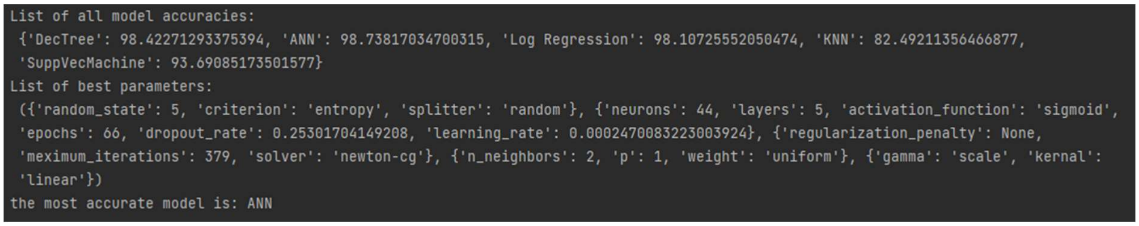


```
confusion  female  male  All
matrix
female        285     3  288
male           37   309  346
All           322   312  634
            SVM report:
               precision    recall  f1-score   support

      female       0.89      0.99      0.93       288
        male       0.99      0.89      0.94       346

    accuracy                           0.94       634
   macro avg       0.94      0.94      0.94       634
weighted avg       0.94      0.94      0.94       634
```

Figure 8:SVM Results



```
List of all model accuracies:
 {'DecTree': 98.42271293375394, 'ANN': 98.73817034700315, 'Log Regression': 98.10725552050474, 'KNN': 82.49211356466877,
 'SuppVecMachine': 93.69085173501577}
List of best parameters:
 ({'random_state': 5, 'criterion': 'entropy', 'splitter': 'random'}, {'neurons': 44, 'layers': 5, 'activation_function': 'sigmoid',
 'epochs': 66, 'dropout_rate': 0.25301704149208, 'learning_rate': 0.0002470083223003924}, {'regularization_penalty': None,
 'meximum_iterations': 379, 'solver': 'newton-cg'}, {'n_neighbors': 2, 'p': 1, 'weight': 'uniform'}, {'gamma': 'scale', 'kernal':
 'linear'})
the most accurate model is: ANN
```

Figure 9:Final Results

Limitations

For Logistic Regression, the penalty refused to work for ‘l1’ and ‘elasticnet’, mainly due to a mismatch of solvers being used.

For SVM, polynomial was excluded from testing due to the exaggerated testing times.

The trails outputted the best performing parameters, this tended to skewer data towards parameters that incited some chance to randomly perform better than more stable hyper parameters combinations.

Discussion

Decision Tree

A scatter plot in the “random_state” graph confirms that the hyper parameter has no impact on the algorithm, this was to be expected as this variable only affects the seed used for the random number generator.

The criterion graph shows that the information gain criterion,”entropy” and “log_loss”, are superior for this type of implementation rather than the Gini impurity. These dictate the best splits that the decision tree could make. On best splits, “random” achieved higher highs and lower lows than the much more consistent “best”, this was to be expected, it can be said that the lows are due to it randomly picking the weakest of the splits and its highs are due to its benefits of reducing overfitting.

Artificial Neural Network

ReLu is usually the choice for hidden layers given it is more computationally efficient and effective, it was quite surprising to see it behind the sigmoid activation function. This simply show the importance of varying and scientifically testing hyperparameters as, in this case, the sigmoid function is better suited for the specific dataset and task.

The number of hidden layers and neurons are the essential building blocks in an ANN, they dictate the ANNs ability to learn complex patterns in the dataset. Therefore, during testing, it was quite evident that having less than 5 layers significantly hindered the ANN. The sweet spot for neurons varied, mainly due to multiple hyperparameter combinations being valid, many neurons with a high dropout may be equivalent to low neurons with low dropout.

The range of acceptable learning rate values was quite apparent, this was to be expected as a high learning rate increases the chance of overshooting, the lower learning rates proved more consistent for this reason. However, one should keep in mind that a smaller learning rate increases training time, so taking the larger value without diminishing returns is ideal.

The dropout rate was found to perform well with a rate of 0.15 and 0.35 before experiencing a sharp drop in performance. With any values less resulting in overfitting and more resulting in too many neurons being dropped, preventing complex patterns in the data to be learnt.

Log Regression

The limit placed on the classifier by the maximum iterations hyperparameter did not significantly affect the algorithm, we may however observe a lower trail throughout the graph most likely caused when the classifier failed to converge leading to a drop in performance.

Regularization penalty had numerous problems due to most criteria being deprecated, most algorithms in the solver hyper parameter do not work with most penalties, hence the results of the graph are neglectable.

Saga and sag lagged behind newton-cg. Saga and sag are utilised more for larger datasets due to their fast speeds. Newton-Cholesky is much more powerful, despite its larger memory usage and training times, it clearly performed the best overall.

K-Nearest Neighbours

The results of the KNN trails were the most opposing to perceived predictions, some observations may be made. The number of neighbours “k” performed better with lower values (2-4), when there are only two categories (male or female) taking more datapoints may just increase the likelihood of selecting bad neighbours, especially if uniform distance is used.

The distance metric used skewed heavily for Manhattan distance, this could be due to the number of fields causing noise in the algorithm, hence Manhattan distance would be more sensitive to large differences in feature values. Additionally, it is less affected by outliers present in the data.

The trials heavily preferred a k of 4 and a “distance” due to its stability, however using a k of 2 the classifier was randomly able to get a better accuracy by using a k of 2.

Support Vector Machine

The Support Vector Machine did not have as varied hyper parameters. The graphs show a distinct preference to the linear kernel, it should be noted that the default parameter is rbf. Linear is inherently the best due to both sexes having a concrete distinction, simply put the data set is not complex enough to require a non-linear kernel

In terms of the gamma hyperparameter, there was no significant difference in performance when using "auto" or "scale". The gamma parameter controls the width of the radial basis function only in the RBF kernel; hence it did not affect linear kernel instances.

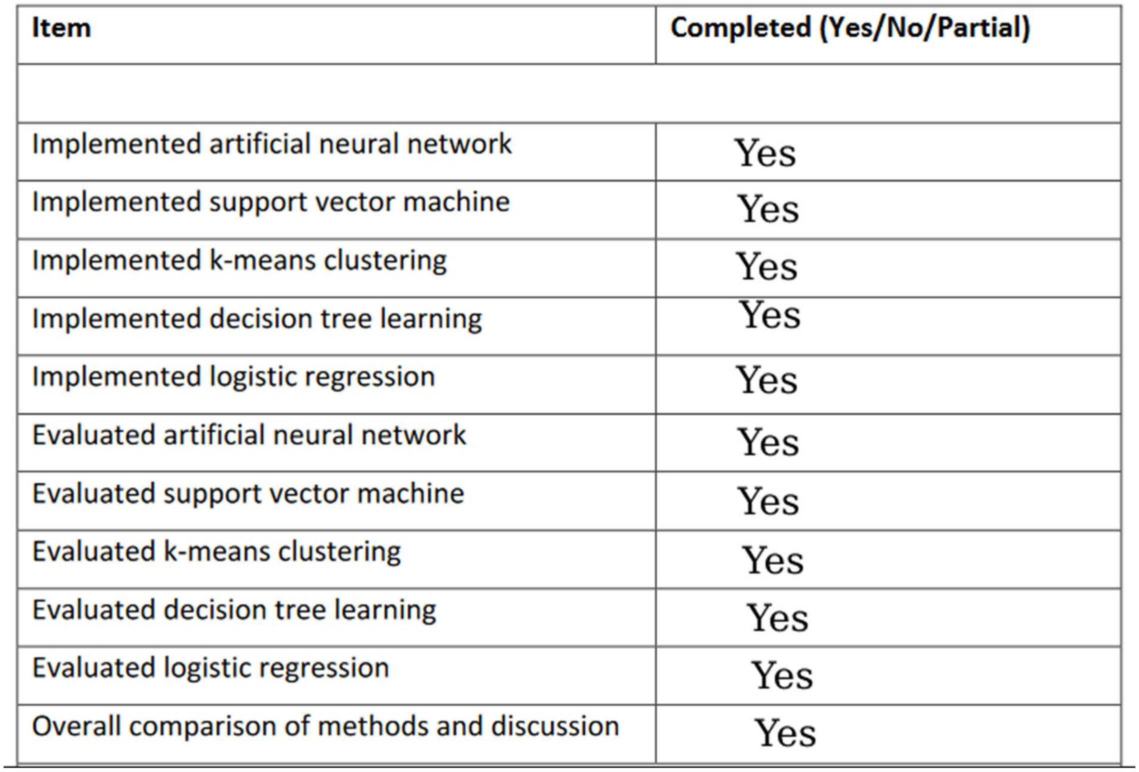
Comparison of methods

What was interesting is that although Euclidean distance was usually preferred, the KNN classifier performed noticeably better with Manhattan distance. Manhattan distance is less computationally expensive and is more sensitive to the overall difference in values.

Although the ANN reached a higher accuracy, Decision Tree reached a similar level of accuracy in a fraction of the time compared to the ANN. This efficiency makes it the preferrable for this situation. Additionally, Decision Trees are also easier to implement and interpret compared to the black box nature of an ANN.

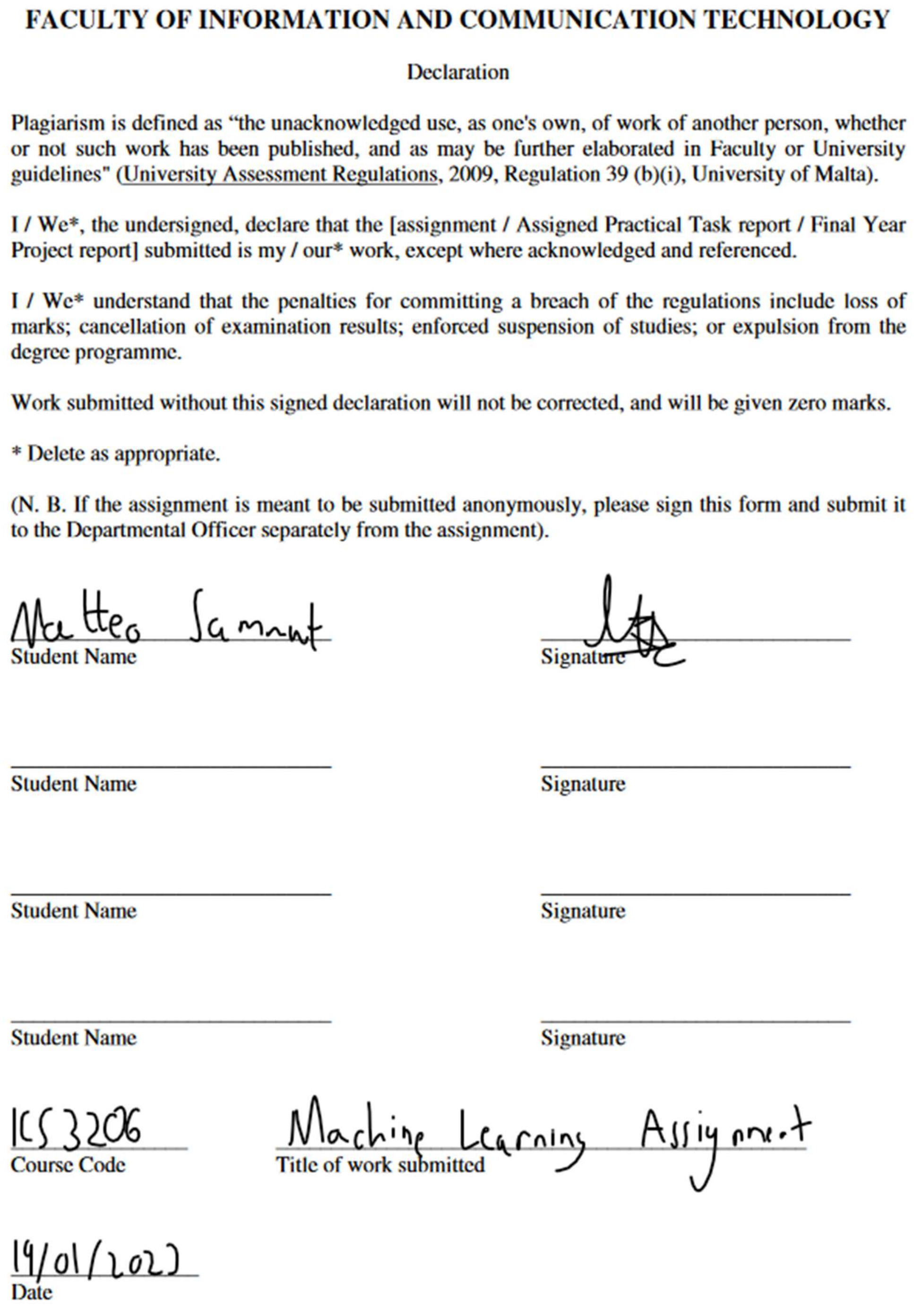
KNN had the worst showing, due to its simplicity although being relatively fast. Despite its relatively fast computation time, the KNN classifier's low accuracy makes it the least preferable option.

Statement of Completion



Item	Completed (Yes/No/Partial)
Implemented artificial neural network	Yes
Implemented support vector machine	Yes
Implemented k-means clustering	Yes
Implemented decision tree learning	Yes
Implemented logistic regression	Yes
Evaluated artificial neural network	Yes
Evaluated support vector machine	Yes
Evaluated k-means clustering	Yes
Evaluated decision tree learning	Yes
Evaluated logistic regression	Yes
Overall comparison of methods and discussion	Yes

Plagiarism Form



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Declaration

Plagiarism is defined as "the unacknowledged use, as one's own, of work of another person, whether or not such work has been published, and as may be further elaborated in Faculty or University guidelines" (University Assessment Regulations, 2009, Regulation 39 (b)(i), University of Malta).

I / We*, the undersigned, declare that the [assignment / Assigned Practical Task report / Final Year Project report] submitted is my / our* work, except where acknowledged and referenced.

I / We* understand that the penalties for committing a breach of the regulations include loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.

Work submitted without this signed declaration will not be corrected, and will be given zero marks.

* Delete as appropriate.

(N. B. If the assignment is meant to be submitted anonymously, please sign this form and submit it to the Departmental Officer separately from the assignment).

Matteo Sammut
Student Name

Signature

Student Name

Signature

Student Name

Signature

Student Name

Signature

ICS3206
Course Code

Machine Learning Assignment
Title of work submitted

19/01/2022
Date