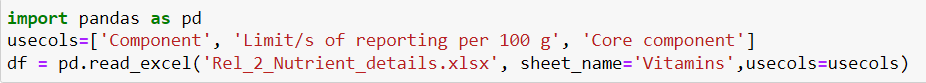
I am doing classification problem on the vitamins part in Australian Food Composition Database - Nutrient information. The input feature variables are component, Limit/s of reporting per 100 g and output feature variable is core component. The target variable would be that a component is core component or not.

For determining the correlation between the input feature variables (component and Limit/s of reporting per 100 g) and the output feature variable (core component), I am using point-biserial correlation as a correlation measure. To cite the standard for the strength of the correlation, I referred to common guidelines for interpreting correlation coefficients. One widely used guideline mentioned in Neter, J., Kutner, M. H., Nachtsheim, C. J., & Wasserman, W. (1996). Applied linear statistical models (4th ed.). Irwin suggests the following thresholds for correlation strength:

* 0 to 0.3: Weak correlation
* 0.3 to 0.7: Moderate correlation
* 0.7 to 1.0: Strong correlation

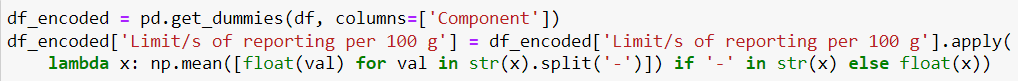
**Data Loading and Column selecting**

I read the selected columns, are component, Limit/s of reporting per 100 g and core component on Jupyter Notebook by using pandas library of python.



**Data Pre-processing**

Now, I am doing data preprocessing on the dataframe to do one hot encoding of the component column and handling range values in Limits of reporting per 100 g column.



Preparting the input features X and target variable y,

A picture containing text, font, screenshot, line

Description automatically generated

Splitting the dataset to training and testing, Training dataset consists of 80% of the dataset,  


Filling the missing values in input features of training and testing dataset by using mean of all values. Updating the target variable accordingly. Gave zero value to the core component empty values and 1 to the **ü**

A picture containing text, font, screenshot, white

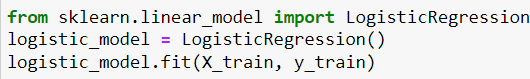
Description automatically generated

A picture containing text, font, screenshot, line

Description automatically generated

**Applying ML models,**

Initializing and training the logistic regression model,



Making predictions on the test set using logistic regression,

A picture containing font, text, screenshot, line

Description automatically generated

Evaluating the logistic regression model,

A picture containing text, font, screenshot, line

Description automatically generated

Printing the precision, recall, F1-score and support by using classification report of the sklearn library,

A screenshot of a computer

Description automatically generated with medium confidence

Computing and plotting the confusion matrix,  
A screenshot of a graph

Description automatically generated with low confidence

Now applying Multi-Layer Perceptron on it,

Defining the network architecture using Tensorflow and keras library of python, hidden layers are 2 with 100 nodes and ReLU activation function. Output layer will have 1 node and sigmoid is applied as activation function as it is binary classification.

A screenshot of a computer code

Description automatically generated with medium confidence

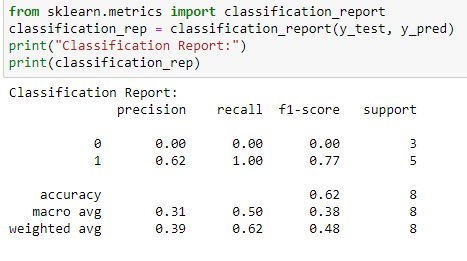
Compiling the model by using binary cross entropy as the loss as it is binary classification problem, Adam optimizer is used as it showed best accuracy. Then the model is trained by using batch size of 2 and 10 epochs. The accuracy during training is also plotted below.

A picture containing text, diagram, screenshot, line

Description automatically generatedA screenshot of a computer program

Description automatically generated with low confidence

Making predictions on the test set using Multilayer perceptron model and evaluating the MLP model. Printing the precision, recall, F1-score and support by using classification report of the sklearn library,



Now, Finding and plotting the confusion matrix of the evaluation,

