**Report for Assessment 1: Design Regression Models**

**Task 1. Load the dataset, do basic data pre-processing, and split the dataset**

The first step is to load the dataset using “pandas”, print the length of the dataset and display the first 5 rows of the dataset. There are a total of 2928 rows in the dataset. Only the column “Status” is categorical and all other columns are numerical.

The shape of the dataset is (2928, 20), which means that there a total of 2928 rows in the dataset and that each row in the dataset has 20 columns.

Then, check if there are any missing values in the dataset. Since there are only very small number of rows with missing values in the dataset, those rows are just removed for easy processing. After removing those rows with missing values, there are a total of 2909 rows in the dataset, which means only 19 rows are removed.

Next, check if there are any duplicated rows and it is found that there are no duplicated rows.

After that, get a summary of numerical columns, and check value count for the categorical variable "Status".

The next step is to transform the nominal variable "Status" into dummy variables “Status\_Developing” and “Status\_Developed”. The "Status\_Developed" column is dropped because “Status\_Developing” and “Status\_Developed” are mutually exclusive and thus only one of them (either one) is required.

After the above processing, display the first 5 rows of the dataset. The column “Status” is replaced by the column “Status\_Developing”. Now, all columns are numerical. The shape of the dataset is (2909, 20), which means that there a total of 2909 rows in the dataset and that each row in the dataset has 20 columns.

Define the input variables and the target variable according to the following:

* The first column (with column index 0) is the target variable "Life expectancy".
* All other columns (with column index from 1 to the end) are the input variables.

Split the dataset into training and testing (with 10% of the dataset for testing).

Apply normalisation on the input variables X (both the training and test set) according to the following sequence:

1. Fit scaler on training data only.
2. Transform training data.
3. Transform testing data.

Do not fit scaler on both training and testing data because testing data are unseen during training.

**Task 2. Train and evaluate the two regression models on the training set with a cross-validation method, optimise the models, and evaluate models on the test set**

According to Freedman (2009), linear regression is approach for modelling the relationship between a dependent variable and one or more independent variables. When there is only one independent variable, it is called simple linear regression. When there are more than one independent variable, it is called multiple linear regression. According to Cortes and Vapnik (1995), support vector machines (SVMs) are supervised learning models, which are able to analyse data for both regression and classification.

Run a 5-fold cross validation for both the Linear Regression and SVM models with default parameter settings. The average r-squared scores based on the cross validation results are shown in the table below:

|  | Linear Regression model | SVM model |
| --- | --- | --- |
| Average r-squared | 0.8376013975750766 | 0.8546740513419694 |

According to the above table, the SVM model has a slightly higher average r-squared score than the Linear Regression model.

For the Linear Regression model, the following parameter setting is applied in finetuning the model:

* 'fit\_intercept': [True, False]
* 'positive': [True, False]

For the SVM model, the following parameter setting is applied in finetuning the model:

* 'kernel': ('linear', 'poly', 'rbf', 'sigmoid')
* 'C': [10, 50]
* 'degree': [2, 3]
* 'coef0': [0.001, 0.005]
* 'gamma': ('auto', 'scale')

Before the above parameter setting for the SVM model is finally chosen, many other values for ‘C’ and ‘coef0’ have been tried. According to the results obtained, the above parameter setting may generate better performance. Therefore, it is chosen as the final parameter setting for the SVM model. Values for “degree’ have been intentionally limited to no greater than 3 to reduce running time of the grid search.

The average r-squared scores for each model (before and after finetuning each model) are shown in the table below:

|  | Linear Regression model | SVM model |
| --- | --- | --- |
| Average r-squared  before finetuning | 0.8376013975750766 | 0.8546740513419694 |
| Average r-squared after finetuning | 0.8376013975750766 | 0.9267662864920666 |

According to the above table, the performance of the Linear Regression model is not improved after finetuning. How ever, the performance of the SVM model is improved significantly after finetuning.

Finally, the average r-squared scores for the two optimised models on the testing dataset are shown in the table below:

|  | Linear Regression model | SVM model |
| --- | --- | --- |
| Average r-squared using the best model on the testing dataset | 0.8323515207054641 | 0.9336508093960161 |

According to the above table, the performance of the optimised SVM model on the testing dataset is significantly better than that of the optimised Linear Regression model.

**References**

Cortes, Corinna; Vapnik, Vladimir (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297.

David A. Freedman (2009). Statistical Models: Theory and Practice. Cambridge University Press.