Report on Investment Value Prediction using Multilayer Perceptron

I. Introduction

This report describes a machine learning model developed for predicting investment values based on historical foreign direct investment (FDI) data. The model is a multilayer perceptron (MLP) implemented using the TensorFlow library in Python. The dataset consists of annual investment data across various countries, and years.

II. Performance

The model prediction performance was evaluated using the root mean squared error (RMSE) metric. RMSE was chosen as the primary performance metric for several advantages:

1. **Interpretability**: RMSE is expressed in the same units with target variable (in this case millions of USD) which make it easier to interpret to the meaning of the model's performance [1]. RMSE indicate the mean deviation between the predicted values and the true value on the dataset.

2. **Sensitivity to large errors**: The larger the error, the higher the weight RMSE will place on it due to the squared term in its calculation [2]. This is quite critical in the presence of investment value prediction, when big errors may result in big financial problems. Through RMSE, we focus on a large prediction error reduction in the first place.

3. **Common usage**: RMSE is the most popular metric to be used in regression model evaluation [1]. The high-profile nature of the domain allows one to benchmark the model's performances with those of other models and benchmarks in the same domain.

RMSE is calculated as follows [3]:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

where $\boldsymbol{n}$ is the number of instances, $\boldsymbol{y_i}$ is the true value, and $\boldsymbol{\hat{y}_i}$ is the predicted value for instance $\boldsymbol{i}$.

There were 12,276 instances in the dataset. The dataset was separated between 80% (9280) of the train set and 20% (2465) of the test set using random sampling with a fixed seed to ensure reproducibility.

The first experimentation obtained the test data RMSE 2995 after 100 epochs and batch size of 50. This tells us that, on average, the predicted values depart from the true value by roughly 2995.

In order to increase the model's accuracy, a grid search strategy was performed to fine-tune the hyperparameters. The following hyperparameters were tuned:

- **Number of epochs**: [100, 200, 300]
- **Dropout rate**: [0.2, 0.3, 0.5]
- **Learning rate**: [0.001, 0.01, 0.1]
- **Activation function**: ['relu', 'leaky_relu', 'elu']

The grid search evaluated all possible combinations of these hyperparameters, and the best configuration was selected based on the lowest validation loss. The optimal hyperparameters found were:

- **Number of epochs**: 200
- **Dropout rate**: 0.3
- **Learning rate**: 0.001
- **Activation function**: 'relu'

The fine tuning hyper parameters were that the model got a test RMSE of 0.053181 after 200 epochs with batch size of 32. This indicates that, on average, the true values deviate from the model outputs by approximately 0.053181 million USD.

Figure 1: Training and Validation Loss Curves

The model performance was also evaluated by observing training and validation loss curves. The Figure 1 is a demonstration that both training and validation losses are decreasing along the epochs with the validation loss steadily on the heels of the training loss. Such a performance implies that the model takes in enough information from the data and the generalizes well to new cases.

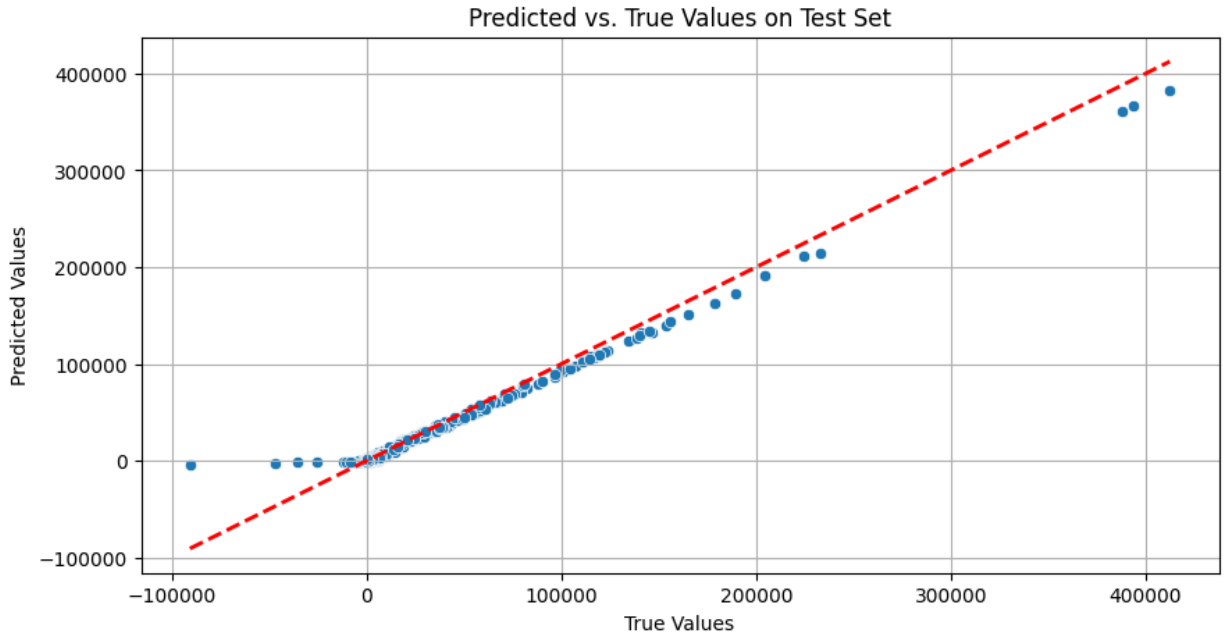

Figure 2: Predicted vs. True Values on Test Set

Together with the loss curves, a scatter plot of the predicted vs actual for the test set had also been plotted. The Figure 2 shows a very high positive correlation between predicted and true values. Therefore, it is implied that the model is predicting accurately.

Finally, the selection of RMSE as the main indicators combining the process of hyperparameter tuning and the analysis of the loss curves and prediction scatter plot illustrate that the investment value prediction model gets a good performance and has the ability to learn patterns from the historical investment data.

III. MLP Model

This MLP model that predicts the investment value contains an input layer, two hidden layers with ReLU activation, and an output layer with a single unit and linear activation. The model architecture can be summarized as follows:

- **Input layer**: Variable number of units based on the amount of input features.
- **Hidden layer 1**: 64 neurons units with ReLU activation, proceeding with 30% dropout.
- **Hidden layer 2**: 32 neurons units with ReLU activation, proceeding with 30% dropout.
- **Output layer**: 1 unit with linear activation.

In the hidden layers we use the ReLU (Rectified Linear Unit) activation function. It is defined as [4]:

$$f(x) = \max(0, x)$$

where $\boldsymbol{x}$ is the input argument to the activation function. ReLU is often one of the top options for nonlinearity in hidden layers. It helps solve the vanishing gradient issue and boosts the sparse representations [5].

The output layer employs a linear activation function:

$$f(x) = x$$

This allows the model to output continuous values for the investment value predictions.

The model is trained to minimize the MSE(mean squared error) loss, which is most commonly used loss function for regression problems [6].MSE is calculated as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where $\boldsymbol{n}$ is the number of instances, $\boldsymbol{y_i}$ is the true value, and $\boldsymbol{\hat{y}_i}$ is the predicted value for instance $\boldsymbol{i}$.

To prevent overfitting, the following regularization techniques were applied:

1. **L2 regularization**: L2 regularization is also Tikhonov regularization or ridge regression which adds a term of penalty in the loss function that is square to the weights [7]. Therefore, it induces the model to learn very small weights, which further inhibits overfitting. The L2 regularization term is calculated as:

   $$L2 = \lambda\sum_{i=1}^{k} w_i^2$$

   where $\boldsymbol{\lambda}$ is the regularization strength and $\boldsymbol{w_i}$ are the model weights.

   L2 regularization is used because it has several benefits:
   - It helps reduce the model's sensitivity to individual features, making it more robust to noise and outliers [8].
   - It is capable of enhancing the model's generalization by balancing the complexity of the learned function [9].
   - It is computationally efficient and easy to implement in gradient-based optimization algorithms [7].

   The regularization strength ($\boldsymbol{\lambda}$) is a hyperparameter that controls the trade-off between fitting the training data and keeping the weights small. It was determined through hyperparameter tuning.

2. **Dropout**: Dropout is a method of regularization of which at each update a fraction of input units are randomly set to zero during training [10]. This enables

the model to not absorb certain features and inspires it to learn in a better way. In this structure, dropout layers with dropout rate of 0.3 (30%) were put after each hidden layer.

Dropout acts as a form of ensemble learning, where multiple subnetworks are trained on different subsets of the data [11]. This increases the model's generalizability and lowers the risk of overfitting.

By combining L2 regularization and dropout, the MLP model is able to learn a robust and generalizable mapping from the input features to the investment value predictions while mitigating the risk of overfitting.

IV. Features & Labels

The target label for the investment value prediction task is the **Value** column from the dataset, which represents the FDI value in millions of USD.

The feature selection strategy included analyze of correlation, and PCA. Initially, the following features were considered based on their potential relevance to investment value prediction:

1. Area Code (M49)
2. Element Code
3. Item Code
4. Year Code

	Area Code (M49)	Element Code	Item Code	Year Code	Value
Area Code (M49)	1	2.5×10^{-16}	5.4×10^{-16}	8.4×10^{-17}	0.69
Element Code	2.5×10^{-16}	1	-2.8×10^{-16}	-7.7×10^{-16}	-0.067
Item Code	5.4×10^{-16}	-2.8×10^{-16}	1	-8.3×10^{-17}	0.35
Year Code	8.4×10^{-17}	-7.7×10^{-16}	-8.3×10^{-17}	1	0.62
Value	0.69	-0.067	0.35	0.62	1

Table 1 – Correlation Matrix

To further refine the feature selection, as you see in Table 1, a correlation matrix was computed to evaluate the pairwise correlations between the relevant features and the variable of target. The correlation matrix sheds light on the degree of the linear relationships and the direction of each variable [12].

Additionally, PCA was applied to the standardized features to identify the most informative principal components [13]. The PCA is a dimensionality-reduction approach that convert the original features to a new set of orthogonal features known as principal components that account for the most variance in the dataset [14].

The PCA analysis indicated that the first four principal components accounted for 95% of the variation in the dataset. The loadings of the original feature on these principal components were identified to understand their importance.

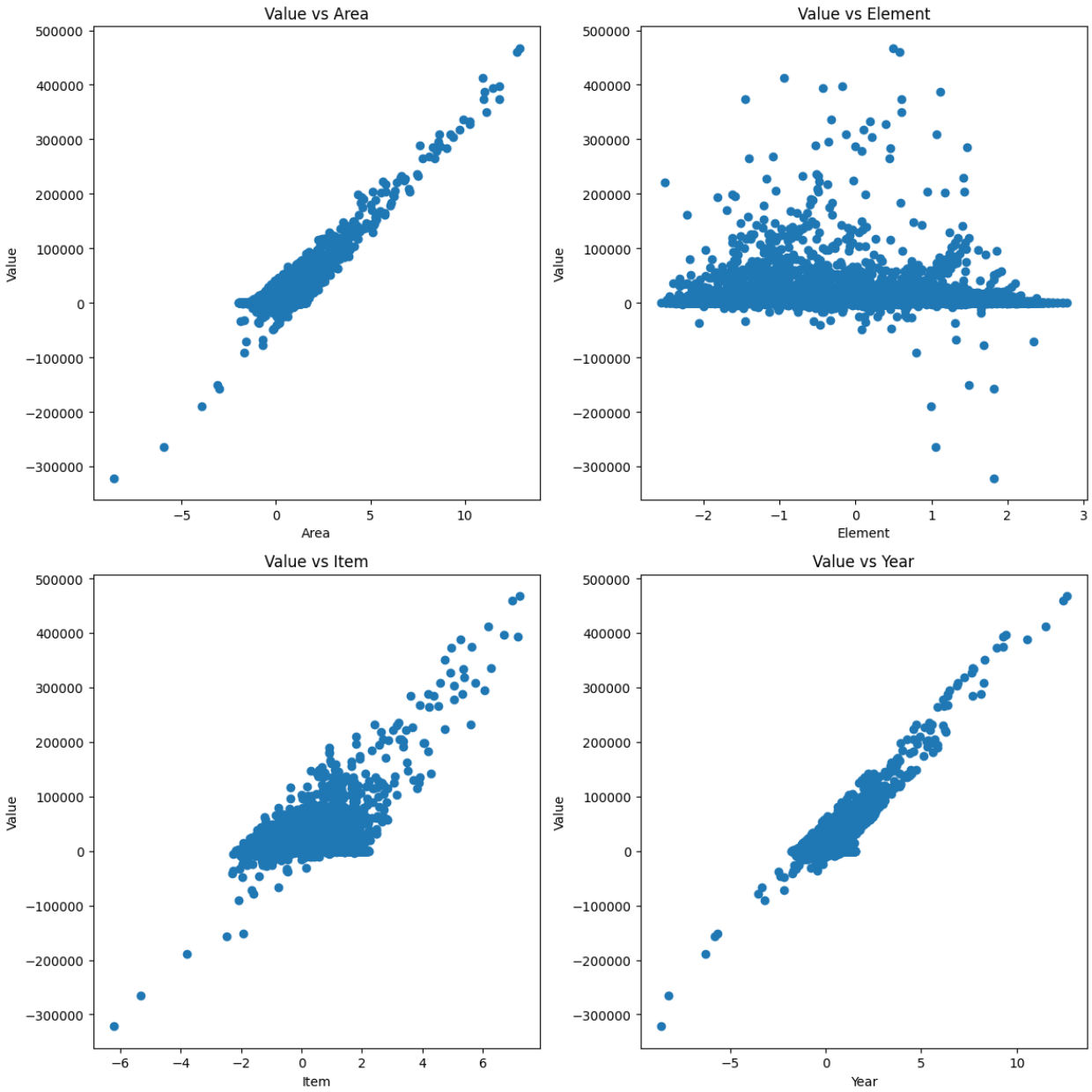

Figure 3: Value vs Principal Components Correlation

Based on the corellation matrix (Table 1) and visualisation of correlation (Figure 3), the following observations were made:

- The first and second principal component (PC1, PC2) had high positive loadings for **Area Code (M49)** and **Year Code**, indicating that these features were strongly associated with the maximum variance direction in the dataset.

- The third principal component (PC3) had a high positive loading for **Item Code**, suggesting that Item Code captured a significant portion of the remaining variance.

- The fourth principal component (PC4) had a relatively low loading for **Element Code**, indicating that Element Code did not contribute significantly to the variance explained by the first three principal components.

Combining the insights from the correlation analysis and PCA, the following features were selected for the investment value prediction model:

1. Area Code (M49)
2. Item Code
3. Year

These features were chosen because they demonstrated strong relationships with the target variable based on the correlation matrix and had high loadings on the most informative principal components identified by PCA.

Element Code was excluded from the final feature set due to its weak correlation with the target variable and low contribution to the variance explained by the principal components.

In summary, a total of 3 input features were used in the model: Area Code (M49), Item Code, and Year. The feature selection process leveraged correlation analysis, and PCA to identify the most relevant and informative features for investment value prediction.

V. Preprocessing

The preprocessing stage is crucial in preparing the data for building the investment value prediction model. It involves several steps to handle missing values, analyze the statistical properties of the features, and perform necessary transformations.

1. **Analyze of Missing Data**

   First step is pre-processing to check for missing values in the dataset. This is important because missing values can impact the model's performance and lead to biased results [15]. The missing values were analyzed using the ***isnull()*** function in pandas, which returns a boolean mask indicating the presence of missing values for each feature.

   Based on the missing value analysis, it was found that there were no missing values in any of the features or the target variable. This eliminates the need for imputation or removal of instances with missing values.

2. **Statistical Analysis**

Statistical analysis was used to understand the trend and distribution of the numerical features. The ***describe()*** function in Pandas was employed to calculate the summary statistics, such as count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values.

The statistical summary provided an overview of the central tendency, dispersion, and range of the numerical features. This information is valuable for understanding the scale and variability of the features and can help identify potential outliers or unusual patterns.

3. **Categorical Analysis**

Categorical features were analyzed to understand the distribution and frequency of different categories. The ***value_counts()*** function in Pandas was used to count the occurrences of each unique category for the relevant categorical features.

The categorical analysis revealed the number of instances for each category and highlighted any class imbalances. This information is important for understanding the representation of different categories in the dataset and can guide decisions on handling categorical features, such as one-hot encoding or label encoding [16].

4. **Feature Scaling**

When independent variables or features have different variance in the data, the feature scaling is a preprocessing technique used to normalize the range [17]. It is particularly important in cases, where features vary in their scales or units. Scale-invariant regularization guarantees all features are equally considered during training.

In this preprocessing step, numerical features were standardized via scaling. Normalization re-scales input variables to zero mean and unit variance. The standardization formula is given by:

$$z = \frac{x - \mu}{\sigma}$$

where $\boldsymbol{x}$ represents the original feature value, $\boldsymbol{\mu}$ is the mean of the feature, and $\boldsymbol{\sigma}$ is the standard deviation of the feature.

Standardization was applied to the selected features: Area Code (M49), Item Codes (IC), and Year. A `StandardScaler` object from the Scikit-learn toolkit was used to do standardization. It was fitted to training data somehow then used to transform training and testing sets to make all the data be scaled equally.

Scaling the features tries to boost the quality of convergence of optimization algorithms and possibly makes the model pay the same attention to each feature during the training process [18].

5. **Train-Test Split**

After preprocessing, the **train_test_split** function in the Scikit-learn library was used to split the data into training and testing sets. The split was made with a test size of 0.2, implying that 20% of the data was set aside for testing and the remaining 80% was used for training. The random seed was set at a fixed value in order to guarantee reproducibility of the split.

Dividing the data into sets for both training and testing allows to evaluate the model's performance on previously unknown data and detect overfitting [19].

In summary, the preprocessing step involved missing data analysis, statistical analysis of numerical features, categorical analysis, feature scaling using standardization, and splitting the data into training and testing sets. These preprocessing techniques help to ensure the quality and compatibility of the data for building the investment value prediction model.

VI. Conclusion

To summarize, this is a report on an MLP model which was developed to predict FDI values. The model gave 0.053181 RMSE in test using 3 input features such as area code, item code and year. It used L2 regularization and dropout techniques to avoid overfitting. The input feature was chosen on the basis of correlation analysis, and PCA to select the relevant and meaningful features for investment value prediction.

By performing missing data analysis, feature analysis for numerical features and categorical features, feature scaling using standardization, and separating the data into training and testing sets were the preprocessing steps I performed. Application of these data preprocessing techniques attempts to end up with better quality and standardized data for developing an investment value prediction model.

The selection of RMSE over other performance metrics, the implementation of hyperparameter tuning and a plot of the loss curves and prediction scatter, are indicators that the investment value prediction model manages to learn patterns from historical investment data and, it performs well.

The present model is constrained by a lack of variety in the selection of input features. Adding certain other key variables like economic indicators or geopolitical factors could greatly improve the model's predictive accuracy.

Future studies could dive into more advanced neural network architectures (CNN or RNN) that are better at uncovering more complex patterns in the investment data. Other

ensemble methods including transfer learning techniques may also be examined to further improve the model's generalization ability.

Developed investment value prediction model gives a base for decision-makers involved in FDI which neither offers the right opportunities nor predictions. Through the application of the statistical data and machine learning approach, the model allows for the issuance of befitting prediction of investment values, hence empowering investment planning and risk evaluation. Such model has real-life applicability as an aid for investors, policy-makers and researchers in the process of studying and predicting evolving investment trends.

Overall, the model of machine learning based investment value prediction proposed for FDIs in the given report shows the capability of machine learning in the FDI domain. With enhanced processing and the incorporation of additional suggested features, the model will be able to make more accurate and data-driven decisions that are based on considerable data inputs.

References:

[1] Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, *7*(3), 1247–1250. https://doi.org/10.5194/gmd-7-1247-2014

[2] Hyndman, R., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

[3] Barnston, A. G. (1992). Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score. *Weather and Forecasting, 7*(4), 699-709. https://doi.org/10.1175/1520-0434(1992)007<0699:CATCRA>2.0.CO;2

[4] Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines Vinod Nair. *ResearchGate*. https://www.researchgate.net/publication/221345737_Rectified_Linear_Units_Improve_Restricted_Boltzmann_Machines_Vinod_Nair

[5] Glorot, X., Bordes, A. & Bengio, Y.. (2011). Deep Sparse Rectifier Neural Networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, in *Proceedings of Machine Learning Research* 15:315-323 https://www.researchgate.net/publication/215616967_Deep_Sparse_Rectifier_Neural_Networks

[6] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer, Verlag. ISBN: 978-0-387-31073-2

[7] Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67. https://doi.org/10.1080/00401706.1970.10488634

[8] Bühlmann, P., & Van De Geer, S. (2011). *Statistics for High-Dimensional data: Methods, Theory and Applications*. Springer. ISBN: 978-3-642-20191-2

[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press. ISBN: 978-0262035613

[10] Srivastava, N., Hinton, G. E., Krizhevsky, A., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *ResearchGate*. https://www.researchgate.net/publication/286794765_Dropout_A_Simple_Way_to_Prevent_Neural_Networks_from_Overfitting

[11] Baldi, P., & Sadowski, P. J. (2013). Understanding dropout. *Advances in neural information processing systems*, 26, 2814-2822. https://proceedings.neurips.cc/paper_files/paper/2013/file/71f6278d140af599e06ad9bf1ba03cb0-Paper.pdf

[12] Benesty, J., Chen, J., Huang, Y., & Cohen, I. (2009). Pearson Correlation Coefficient. In *Springer topics in signal processing* (pp. 1–4). https://doi.org/10.1007/978-3-642-00296-0_5

[13] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical Transactions - Royal Society. Mathematical, Physical and Engineering Sciences/Philosophical Transactions - Royal Society. Mathematical, Physical and Engineering Sciences*, *374*(2065), 20150202. https://doi.org/10.1098/rsta.2015.0202

[14] Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews. Computational Statistics*, *2*(4), 433–459. https://doi.org/10.1002/wics.101

[15] Little, R. J. A., & Rubin, D. B. (2019). Statistical Analysis with Missing Data, Third Edition. In *Wiley series in probability and statistics*. https://doi.org/10.1002/9781119482260

[16] Potdar, K., Pardawala, T. S., & Pai, C. D. (2017). A comparative study of categorical variable encoding techniques for neural network classifiers. *International Journal of Computer Applications*, *175*(4), 7–9. https://doi.org/10.5120/ijca2017915495

[17] Aksoy, S., & Haralick, R. M. (2001). Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recognition Letters*, *22*(5), 563–582. https://doi.org/10.1016/s0167-8655(00)00112-4

[18] Juszczak, P., Tax, D., & Duin, R. (2002). Feature Scaling in Support Vector Data Description. *ResearchGate*. https://www.researchgate.net/publication/2535451_Feature_Scaling_in_Support_Vector_Data_Description

[19] Reitermanova, Z. (2010). Data splitting. In *WDS'10 proceedings of contributed papers* (Vol. 1, pp. 31-36). https://physics.mff.cuni.cz/wds/proc/pdf10/WDS10_105_i1_Reitermanova.pdf