Implementation of Feature Selection using Correlation Matrix in Python

Ahmad Farhan AlShammari
Department of Computer and Information Systems
College of Business Studies, PAAET
Kuwait

ABSTRACT

The goal of this research is to develop a feature selection program using correlation matrix in Python. Feature selection is used to determine the most important features in data. It helps to reduce the number of features, decrease the complexity of computations, increase the accuracy, and improve the performance of the applied model. Correlation matrix is used to measure the correlation between the input (independent) features and the output (dependent) feature. The input features that are highly correlated with the output feature are identified, filtered, and selected.

The basic steps of feature selection using correlation matrix are explained: preparing data (input and output), creating transpose of input data, creating data matrix, computing correlation matrix, plotting correlation matrix, selecting features (adding relevant features and removing redundant features), and printing selected features.

The developed program was tested on an experimental dataset. The program successfully performed the basic steps of feature selection using correlation matrix and provided the required results.

Keywords

Artificial Intelligence, Machine Learning, Feature Selection, Filtering, Correlation Matrix, Correlation Coefficient, Features, Relevant, Redundant, Python, Programming.

1. INTRODUCTION

In recent years, machine learning has played a major role in the development of computer systems. Machine learning (ML) is a branch of Artificial Intelligence (AI) which is focused on the study of computer algorithms to improve the performance and efficiency of computer programs [1-15].

Feature selection is sharing the knowledge of many related fields: machine learning, programming, data science, mathematics, statistics, and numerical methods [16-21].

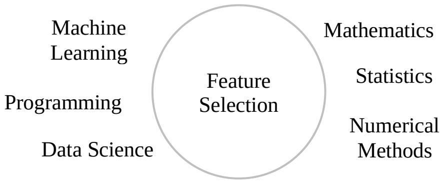


Fig 1: Area of Feature Selection

In this paper, feature selection is applied using correlation matrix to find the most important features in data by computing the correlation between features. Feature selection is mostly used in the applications of machine learning, especially in classification and regression.

2. LITERATURE REVIEW

The review of literature explored the fundamental concepts of feature selection using correlation matrix [22-27].

Feature selection is very important in the field of machine learning. It is used to select the most important features in data. It helps to reduce the amount of processed data, decrease the number of computations, and improve the accuracy of the applied model.

In general, feature selection methods are divided into three main types: filters, wrappers, and embedded.

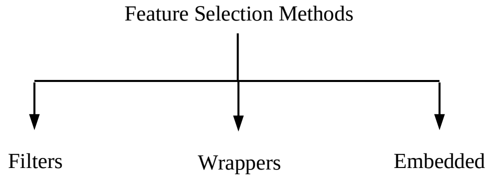


Fig 2: Types of Feature Selection

This research is focused on filters. They are statistical methods used to find the most important features in data. There are different filtering methods such as correlation matrix, Chi-squared, and Analysis of Variance (ANOVA).

Note: In this research, the correlation matrix is applied.

The fundamental concepts of feature selection using correlation matrix are explained in details in the following section.

Feature Selection:

Feature selection is the process of finding the most important features in data that affect the performance of the applied model. The correlation matrix is used to compute the correlation between the input and output features.

The concept of feature selection using correlation matrix is illustrated in the following diagram:

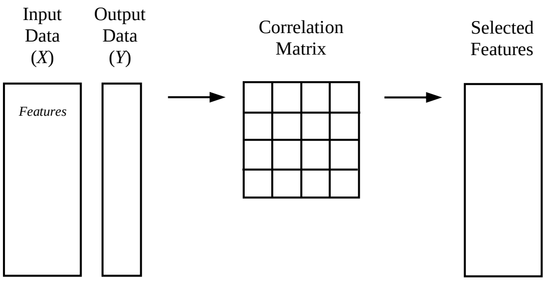


Fig 3: Explanation of Feature Selection

The input data (*X*) consists of the features of the input variables, where variables are represented in columns. It is shown in the following form:

$$X = \begin{bmatrix} x_{0,0} & x_{1,0} & x_{2,0} & \cdots & x_{n-1,0} \\ x_{0,1} & x_{1,1} & x_{2,1} & \cdots & x_{n-1,1} \\ x_{0,2} & x_{1,2} & x_{2,2} & \cdots & x_{n-1,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{0,m-1} & x_{1,m-1} & x_{2,m-1} & \cdots & x_{n-1,m-1} \end{bmatrix}$$

The output data (*Y*) consists of the features of the output variable. It is shown in the following form:

$$Y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{m-1} \end{bmatrix}$$

Then, the transpose of the input data (*X*) is created, where variables are represented in rows. It is shown in the following form:

$$Xt = \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} & \cdots & x_{0,m-1} \\ x_{1,0} & x_{1,1} & x_{1,2} & \cdots & x_{1,m-1} \\ x_{2,0} & x_{2,1} & x_{2,2} & \cdots & x_{2,m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-1,0} & x_{n-1,1} & x_{n-1,2} & \cdots & x_{n-1,m-1} \end{bmatrix}$$

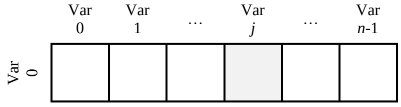
After that, the data matrix (*D*) is the combination of the transpose of the input data (*Xt*) and the output data (*Y*). It is shown in the following form:

$$D = \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} & \cdots & x_{0,m-1} \\ x_{1,0} & x_{1,1} & x_{1,2} & \cdots & x_{1,m-1} \\ x_{2,0} & x_{2,1} & x_{2,2} & \cdots & x_{2,m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-1,0} & x_{n-1,1} & x_{n-1,2} & \cdots & x_{n-1,m-1} \\ y_0 & y_1 & y_2 & \cdots & y_{m-1} \end{bmatrix}$$

Now, the data matrix (*D*) is ready to be used in computing the correlation matrix.

Correlation Matrix:

The correlation matrix (CM) is a statistical table used to measure the correlation between variables. It is represented as a matrix of size (*n*×*n*) as shown in the following form:



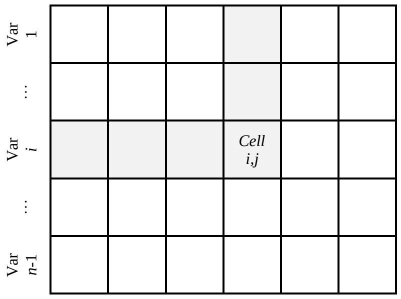


Fig 4: Explanation of Correlation Matrix

where each cell (*i,j*) in the matrix shows the correlation between the variables in row (*i*) and column (*j*).

Correlation Coefficient:

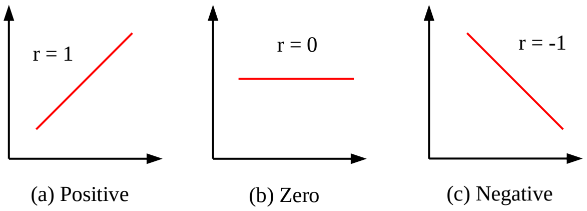
The correlation between two variables (*X* and *Y*) is computed using Pearson's correlation coefficient (*r*) by the following formula:

$$r = \frac{\sum(x - \bar{x})(y - \bar{y})}{\sqrt{\sum(x - \bar{x})^2 \sum(y - \bar{y})^2}} \qquad (1)$$

where: (*x*) and (*y*) are the items of variables (*X*) and (*Y*) respectively, and ($\bar{x}$) and ($\bar{y}$) are their means.

The value of correlation coefficient (*r*) belongs to the range [-1,1], where: (-1) indicates full negative correlation, (0) indicates no correlation, and (1) indicates full positive correlation.

The different types of correlations are illustrated in the following diagram:



(a) Positive (b) Zero (c) Negative

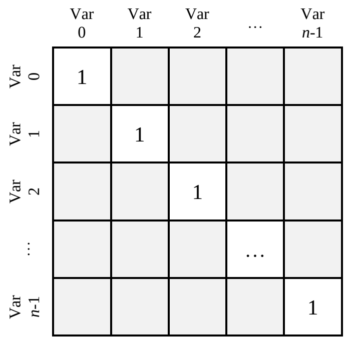
Fig 5: Types of Correlations

The sign of the correlation coefficient (*r*) shows the "direction" of the correlation (positive or negative), and the magnitude of the correlation coefficient (*r*) shows the "strength" of the correlation (weak or strong).

For example, the strength of the correlation coefficient (*r*) can be described by the following scale:

$$|\, r \,| = \begin{cases} 0 \text{ - } 1.99\,, & \text{Very Weak} \\ 0.2 \text{ - } 0.39, & \text{Weak} \\ 0.4 \text{ - } 0.59\,, & \text{Medium} \\ 0.6 \text{ - } 0.79, & \text{Strong} \\ 0.8 \text{ - } 1, & \text{Very Strong} \end{cases}$$

Note: In the correlation matrix, the elements of diagonal are always (1) because they show the correlation between the variable and itself, and the elements of upper triangle are identical to lower tringle, as shown in the following form:



	Var 0	Var 1	Var 2	…	Var *n*-1
Var 0	1				
Var 1		1			
Var 2			1		
…				…	
Var *n*-1					1

Fig 6: Diagonal, Upper, and Lower Triangles

Selecting Features:

The important features are identified, filtered, and selected from the original set of features by the following steps:

1. Adding Relevant Features:

The relevant features are highly correlated with the output feature. If the correlation coefficient (*r*) is above threshold, then it is recommended to add the input feature.

For example, in the following correlation matrix:

	X0	*X1*	*Y*
X0	1		
X1		1	
Y	0.5		1

The input feature (*X0*) is highly correlated with the output feature (*Y*). Therefore, the input feature (*X0*) is added.

2. Removing Redundant Features:

The redundant features are highly correlated with each other. If the correlation coefficient (*r*) is above threshold, then it is recommended to add one of them and remove the other.

For example, in the following correlation matrix:

	X0	*X1*	*Y*
X0	1		
X1	0.8	1	
Y			1

The input features (*X0*) and (*X1*) are highly correlated with each other. Therefore, one of them is added and the other is removed.

Note: The input feature that is more correlated with the output feature is added, and the less correlated is removed.

Feature Selection System:

The feature selection system is explained in the following outline:

Input: Input data (*X*) and output data (*Y*).

Output: Selected Features.

Processing: The input and output data are prepared for processing. First, the transpose of the input data and the data matrix are created. Then, the correlation matrix is computed between the input and output features and plotted. After that, the important features are selected by adding the relevant features and removing the redundant features. Finally, the selected features are printed.

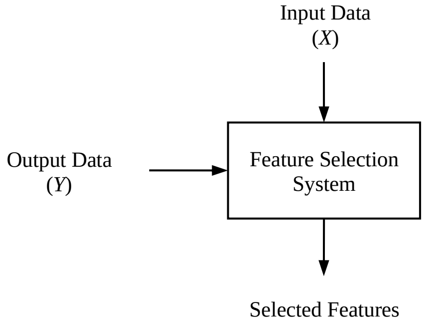


Fig 7: Diagram of Feature Selection System

Python:

Python [28] is a general high-level programming language. It is very simple to code, easy to learn, and powerful. It is the most popular programming language, especially in the development of machine learning applications.

Python provides additional libraries for different purposes, for example: Numpy [29], Pandas [30], Matplotlib [31], NLTK [32], SciPy [33], and SK Learn [34].

Note: In this research, the standard functions of Python are applied without using any additional library.

3. RESEARCH METHODOLOGY

The basic steps of feature selection using correlation matrix are: (1) preparing data (input and output), (2) creating transpose of input data, (3) creating data matrix, (4) computing correlation matrix, (5) plotting correlation matrix, (6) selecting features (adding relevant features and removing redundant features), and (7) printing selected features.

1. Preparing Data:
   1.1. Input Data
   1.2. Output Data
2. Creating Transpose of Input Data
3. Creating Data Matrix
4. Computing Correlation Matrix
5. Plotting Correlation Matrix
6. Selecting Features:
   6.1. Adding Relevant Features
   6.2. Removing Redundant Features
7. Printing Selected Features

Fig 8: Steps of Feature Selection

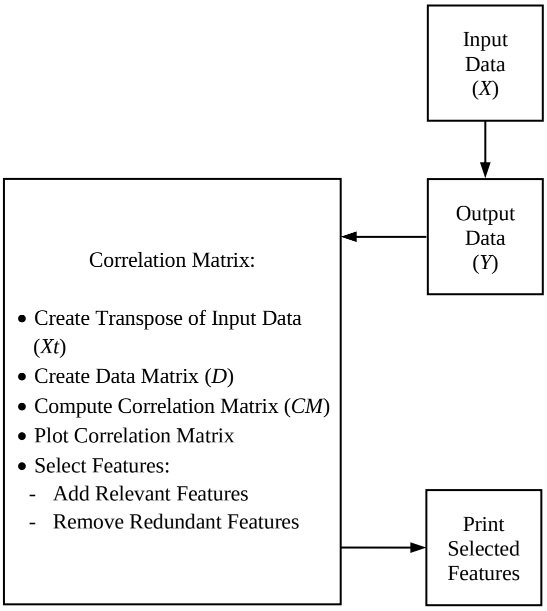


Fig 9: Flowchart of Feature Selection

The basic steps of feature selection using correlation matrix are explained in details in the following section.

1. Preparing Data:

The input and output data are prepared by the following steps:

1.1. Input Data:

The input data (*X*) is obtained from the original source and converted into list in the following form:

```
X = [[x_{0,0}, x_{1,0}, x_{2,0}, ..., x_{n-1,0}],
     [x_{0,1}, x_{1,1}, x_{2,1}, ..., x_{n-1,1}],
     [x_{0,2}, x_{1,2}, x_{2,2}, ..., x_{n-1,2}],
      ...
     [x_{0,m-1}, x_{1,m-1}, x_{2,m-1}, ..., x_{n-1,m-1}]]
```

1.2. Output Data:

The output data (*Y*) is obtained from the original source and converted into list in the following form:

```
Y = [y_0, y_1, y_2, ..., y_{m-1}]
```

2. Creating Transpose of Input Data:

The transpose of the input data (*Xt*) is created by the following code:

```
def transpose(a):
    nr = len(a)
    nc = len(a[0])
    at = []
    for i in range(nc):
        row = []
        for j in range(nr):
            row.append(a[j][i])
        at.append(row)
    return at
```

3. Creating Data Matrix:

The data matrix (*D*) is created as shown in the following form:

```
D = [[x_{0,0}, x_{0,1}, x_{0,2}, ..., x_{0,m-1}],
     [x_{1,0}, x_{1,1}, x_{1,2}, ..., x_{1,m-1}],
     [x_{2,0}, x_{2,1}, x_{2,2}, ..., x_{2,m-1}],
      ...
     [x_{n-1,0}, x_{n-1,1}, x_{n-1,2}, ..., x_{n-1,m-1}],
     [y_0, y_1, y_2, ..., y_{m-1}]]
```

4. Computing Correlation Matrix:

The correlation matrix (*CM*) is computed for all the features in the data matrix (*D*). It is done by the following code:

```
def compute_CM(D):
    n = len(D)
    CM = []
    for i in range(n):
        row = []
        for j in range(n):
            row.append(compute_r(D[i], D[j]))
        CM.append(row)
    return CM
```

The correlation coefficient (*r*) is computed for each two features using formula (1). It is done by the following code:

```
def compute_r(x, y):
    n = len(x)
    m_x = sum(x)/n
    m_y = sum(y)/n
    sum_xy = 0
    sum_x2 = 0
    sum_y2 = 0
    for i in range(n):
        sum_xy += (x[i] - m_x)*(y[i] - m_y)
        sum_x2 += (x[i] - m_x)**2
        sum_y2 += (y[i] - m_y)**2
    return (sum_xy)/math.sqrt(sum_x2*sum_y2)
```

5. Plotting Correlation Matrix:

The correlation matrix is plotted using the matplotlib library. It is done by the following code:

```
import matplotlib.pyplot as plt

threshold = get_max(CM)/2
for i in range(n):
    for j in range(n):
        plt.text(j,i, CM[i][j],
        ha="center",
        va="center",
        color="white" if (CM[i][j] > threshold)
        else "black")
plt.imshow(CM, cmap="Blues")
plt.title("Correlation Matrix")
plt.xticks(range(n))
plt.yticks(range(n))
plt.colorbar()
plt.show()
```

6. Selecting Features:

The important features are identified, filtered, and selected by the following steps:

6.1. Adding Relevant Features:

The relevant features are directly identified from the correlation matrix and added to the selected features.

6.2. Removing Redundant Features:

The redundant features are directly identified from the correlation matrix and removed from the selected features.

7. Printing Selected Features:

The selected features are printed by the following code:

```
print("Selected Features :")
for i in range(len(selected_features)):
    print(selected_features[i])
```

4. RESULTS AND DISCUSSION

The developed program was tested on an experimental dataset from Kaggle [35]. The program performed the basic steps of feature selection using correlation matrix and provided the required results. The program output is explained in details in the following section.

Input and Output Data:

The input data (*X*) and the output data (*Y*) are prepared and printed as shown in the following view:

```
        X0      X1      X2              Y
--------------------------------------------
0:      0.01    18.0    2.31    ...     24.0
1:      0.03    0.0     7.07    ...     21.6
2:      0.03    0.0     7.07    ...     34.7
3:      0.03    0.0     2.18    ...     33.4
4:      0.07    0.0     2.18    ...     36.2
5:      0.03    0.0     2.18    ...     28.7
6:      0.09    12.5    7.87    ...     22.9
7:      0.14    12.5    7.87    ...     27.1
8:      0.21    12.5    7.87    ...     16.5
9:      0.17    12.5    7.87    ...     18.9
...
```

Transpose of Input Data:

The transpose of the input data (*Xt*) is created and printed as shown in the following view:

```
Transpose of X (Xt):
0:      0.01    0.03    0.03    0.03    ...
1:      18.0    0.0     0.0     0.0     ...
2:      2.31    7.07    7.07    2.18    ...
3:      0.54    0.47    0.47    0.46    ...
4:      6.58    6.42    7.18    7.0     ...
5:      65.2    78.9    61.1    45.8    ...
6:      4.09    4.97    4.97    6.06    ...
7:      1.0     2.0     2.0     3.0     ...
8:      296.0   242.0   242.0   222.0   ...
9:      15.3    17.8    17.8    18.7    ...
...
```

Data Matrix:

The data matrix (*D*) is created and printed as shown in the following view:

```
Data Matrix (D):
0:      0.01    0.03    0.03    0.03    ...
1:      18.0    0.0     0.0     0.0     ...
2:      2.31    7.07    7.07    2.18    ...
3:      0.54    0.47    0.47    0.46    ...
4:      6.58    6.42    7.18    7.0     ...
5:      65.2    78.9    61.1    45.8    ...
6:      4.09    4.97    4.97    6.06    ...
7:      1.0     2.0     2.0     3.0     ...
8:      296.0   242.0   242.0   222.0   ...
9:      15.3    17.8    17.8    18.7    ...
...
12:     24.0    21.6    34.7    33.4    ...
```

Correlation Matrix:

The correlation matrix (*CM*) is computed and printed as shown in the following view:

```
Correlation Matrix (CM):
0:      1.0     -0.28   0.57    ...     -0.23
1:      -0.28   1.0     -0.42   ...     0.32
2:      0.57    -0.42   1.0     ...     -0.34
3:      0.79    -0.41   0.72    ...     -0.35
4:      -0.29   0.31    -0.35   ...     0.86
5:      0.5     -0.52   0.57    ...     -0.36
6:      -0.49   0.64    -0.68   ...     0.19
7:      0.24    -0.1    0.14    ...     -0.2
8:      0.43    -0.08   0.48    ...     -0.23
9:      -0.31   -0.3    -0.01   ...     -0.37
...
```

Correlation Matrix Plot:

The correlation matrix is plotted using the matplotlib library and displayed as shown in the following chart:

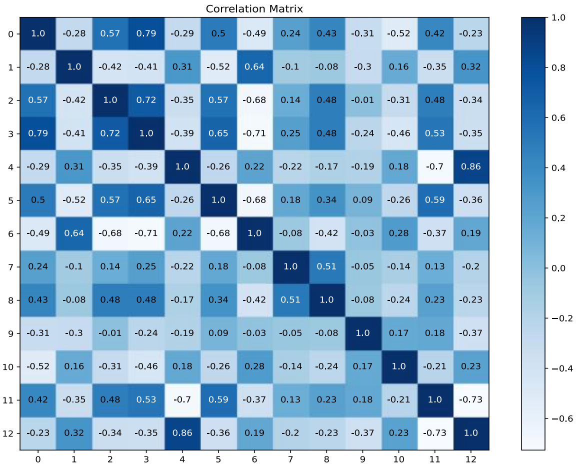


Fig 10: Correlation Matrix Plot

Selected Features:

The important features are identified, filtered, and selected, where the relevant features are added and the redundant features are removed. The selected features are printed as shown in the following view:

```
Selected Features:
1
3
4
5
9
```

In summary, the program has successfully performed the basic steps of feature selection using correlation matrix and provided the required results.

5. CONCLUSION

Machine learning is playing a major role in the development of computer systems. It helps to improve the performance and efficiency of computer programs.

Feature selection is an important process in the field of machine learning. It is used to find the most important features in data. Correlation matrix is used to compute the correlation between the input and output features.

In this research, the author developed a program to perform feature selection using correlation matrix in Python. The developed program performed the basic steps of feature selection using correlation matrix: preparing data (input and output), creating transpose of input data, creating data matrix, computing correlation matrix, plotting correlation matrix, selecting features (adding relevant features and removing redundant features), and printing selected features.

The program was tested on an experimental dataset and provided the required results: data matrix, correlation matrix, correlation matrix plot, and selected features.

In future work, more research is needed to improve the current methods of feature selection using correlation matrix. In addition, they should be more investigated on different fields, domains, and datasets.

6. REFERENCES

[1] Sammut, C., & Webb, G. I. (2011). "Encyclopedia of Machine Learning". Springer Science & Business Media.

[2] Jung, A. (2022). "Machine Learning: The Basics". Singapore: Springer.

[3] Kubat, M. (2021). "An Introduction to Machine Learning". Cham, Switzerland: Springer.

[4] Li, H. (2023). "Machine Learning Methods". Springer Nature.

[5] Dey, A. (2016). "Machine Learning Algorithms: A Review". International Journal of Computer Science and Information Technologies, 7 (3), 1174-1179.

[6] Bonaccorso, G. (2018). "Machine Learning Algorithms: Popular Algorithms for Data Science and Machine Learning". Packt Publishing.

[7] Jo, T. (2021). "Machine Learning Foundations: Supervised, Unsupervised, and Advanced Learning". Springer.

[8] Jordan, M. I., & Mitchell, T. M. (2015). "Machine Learning: Trends, Perspectives, and Prospects". Science, 349(6245), 255-260.

[9] Forsyth, D. (2019). "Applied Machine Learning". Cham, Switzerland: Springer.

[10] Chopra, D., & Khurana, R. (2023). "Introduction to Machine Learning with Python". Bentham Science Publishers.

[11] Müller, A. C., & Guido, S. (2016). "Introduction to Machine Learning with Python: A Guide for Data Scientists". O'Reilly Media.

[12] Zollanvari, A. (2023). "Machine Learning with Python: Theory and Implementation". Springer Nature.

[13] Raschka, S. (2015). "Python Machine Learning". Packt Publishing.

[14] Sarkar, D., Bali, R., & Sharma, T. (2018). "Practical Machine Learning with Python". Apress.

[15] Swamynathan, M. (2019). "Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics using Python". Apress.

[16] Kong, Q., Siauw, T., & Bayen, A. (2020). "Python Programming and Numerical Methods: A Guide for Engineers and Scientists". Academic Press.

[17] Yale, K., Nisbet, R., & Miner, G. D. (2017). "Handbook of Statistical Analysis and Data Mining Applications". Elsevier.

[18] Unpingco, J. (2022). "Python for Probability, Statistics, and Machine Learning". Cham, Switzerland: Springer.

[19] Brandt, S. (2014). "Data Analysis: Statistical and Computational Methods for Scientists and Engineers". Springer.

[20] VanderPlas, J. (2017). "Python Data Science Handbook: Essential Tools for Working with Data". O'Reilly Media.

[21] James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). "An Introduction to Statistical Learning: With Applications in Python". Springer Nature.

[22] Hall, M. A. (1999). "Correlation-based Feature Selection for Machine Learning". (Doctoral Dissertation, The University of Waikato).

[23] Raschka, S. (2018). "Feature Selection, Model Selection, and Algorithm Selection in Machine Larning". arXiv preprint arXiv:1811.12808.

[24] Gopika, N., & ME, A. M. K. (2018). "Correlation based Feature Selection Algorithm for Machine Learning". In 2018 3rd International Conference on Communication and Electronics Systems (ICCES) (pp. 692-695). IEEE.

[25] Cai, J., Luo, J., Wang, S., & Yang, S. (2018). "Feature Selection in Machine Learning: A New Perspective". Neuro Computing, 300, 70-79.

[26] Blum, A. L., & Langley, P. (1997). "Selection of Relevant Features and Examples in Machine Learning". Artificial Intelligence, 97(1-2), 245-271.

[27] Solorio-Fernández, S., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2020). "A Review of Unsupervised Feature Selection Methods". Artificial Intelligence Review, 53(2), 907-948.

[28] Python: https://www.python.org

[29] Numpy: https://www.numpy.org

[30] Pandas: https:// pandas.pydata.org

[31] Matplotlib: https://www. matplotlib.org

[32] NLTK: https://www.nltk.org

[33] SciPy: https://scipy.org

[34] SK Learn: https://scikit-learn.org

[35] Kaggle: https://www.kaggle.com