A comparative study of Credit Scoring and Risk Management Techniques in Fintech: Machine Learning vs. Regression Analysis

Elda Xhumari [1], Sabina Haloci [2]

[1] *University of Tirana, Faculty of Natural Sciences, Department of Informatics, Boulevard "Zogu I", Tirana, 1001, Albania*
[2] *University of Tirana, Faculty of Natural Sciences, Department of Informatics, Boulevard "Zogu I", Tirana, 1001, Albania*

Abstract

In the fintech industry, credit scoring plays a vital role in helping lenders assess the creditworthiness of potential borrowers. While traditional methods rely on regression analysis, machine learning techniques have become a popular alternative due to their increased accuracy and efficiency. However, effective credit scoring involves more than just selecting the right method; it requires a robust risk management framework that considers various data sources. In this study, we evaluate the effectiveness and efficiency of regression analysis and machine learning techniques in predicting creditworthiness, while also considering their role in an overall risk management strategy.

Keywords

Credit Scoring, Risk Management, Fintech, Regression, Machine Learning

1. Introduction

In the period of active development of information technologies, introducing the latest developments into the activities of companies gives them a great advantage. The processes of collecting and processing information are significantly accelerated, reliability and safety are increased. Neural networks are currently becoming one of the topical and most actively developing areas of scientific thought. The purpose of the work is to develop a methodology for solving the problem of credit scoring by using an artificial neural network.

The use of neural networks and machine learning technology in information systems of the banking sector can serve as one of the ways to reduce the risk of default by "discarding" customers with high risks of late loan repayment, which will lead to minimization of the risk of credit transactions.

The essence of the concept of "credit scoring" means the process of assessing the likelihood of failure to fulfill the loan obligations of a potential borrower, i.e. bankruptcy. Credit scoring is designed to classify potential borrowers based on their creditworthiness. Scoring models are based on various statistical methods of dividing potential borrowers into groups that determine the level of risk of their bankruptcy, based on a number of statistics. [1]

2. How it works

Neural network-based machine learning algorithms typically do not require programming with precise rules defining what to anticipate from the input. Instead, the neural network learning algorithm learns by analyzing many labeled examples provided during training and by utilizing this answer key to determine what qualities of the input are required to generate the proper output. The neural network can start processing new, unknown inputs and effectively produce correct results once a sufficient number of examples have been processed.

Proceedings of RTA-CSIT 2023, April 26–27, 2023 Tirana, Albania
EMAIL: elda.xhumari@fshn.edu.al (A. 1); sabina.haloci@fshnstudent.info (A. 2);
ORCID: 0000-0002-1206-6424 (A. 1);

The results usually grow more accurate as the program gains experience and observes a wider range of instances and inputs. For neural networks to function properly, there are four essential procedures to follow: [2]

1. Patterns can be "remembered" by neural networks through associating or training. The computer will match an unfamiliar pattern with the closest match it has in memory if it is presented with that pattern.
2. Putting information or patterns into categories that have already been established.
3. Clustering or identifying a unique element of each data instance to classify it without additional context.
4. Prediction, or the generation of anticipated outcomes utilizing pertinent input, even when the relevant information is not immediately available.

Artificial Neural Networks consists of two phases: [3]

- Forward Propagation
- Backward Propagation

Forward propagation is the process of multiplying weights with each feature and adding them. The bias is also added to the result. Backward propagation is the process of updating the weights in the model. Backward propagation requires an optimization function and a loss function.

3. Types of neural networks

There are several types of neural networks, each with its own structure and function. Here are some of the most common types: [1]

A. Feedforward neural networks: These networks are the simplest type of neural network, and they process input data in one direction, from input to output. The input layer receives data, and the output layer produces predictions based on the patterns learned from the input data.

B. Convolutional neural networks: These networks are commonly used in image and video recognition tasks. They use filters or kernels to extract features from the input data and pass them through multiple layers of processing to produce predictions.

C. Recurrent neural networks: These networks are used for tasks that involve sequential data, such as natural language processing and speech recognition. They have loops that allow information to be passed between neurons over time, allowing the network to remember previous inputs and make predictions based on them.

D. Autoencoder neural networks: These networks are used for unsupervised learning, where there is no labeled data to learn from. They are designed to learn a compressed representation of the input data and then use that representation to reconstruct the original data.

E. Generative adversarial networks: These networks are used for generating new data based on a given input. They consist of two networks: a generator network that produces new data based on an input, and a discriminator network that evaluates the quality of the generated data.

Each type of neural network has its own strengths and weaknesses, and the choice of which type to use depends on the specific task and the characteristics of the data being analyzed.

In credit scoring, a type of feedforward neural network called a Multilayer Perceptron (MLP) is commonly used. MLPs are a type of feedforward neural network that consist of multiple layers of neurons, including an input layer, one or more hidden layers, and an output layer. The input layer receives data from the credit applicant, and the hidden layers process that data to produce an output, which in this case is the credit score.

MLPs are well suited for credit scoring tasks because they are capable of learning complex patterns in large datasets and can handle nonlinear relationships between input features and the output score. Additionally, MLPs can be trained using backpropagation, a powerful algorithm that adjusts the weights and biases of the network to minimize the error between the predicted and actual credit scores.

MLPs are a powerful tool for credit scoring, and they are widely used in the finance industry to assess creditworthiness and manage risk.

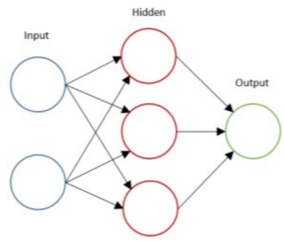

Figure 1: A schematic diagram of a Multi-Layer Perceptron (MLP) neural network

4. Neural Network model

4.1. Data analysis

To develop our model, we will be performing exploratory data analysis (EDA) on the dataset. This will involve gaining an understanding of the data, identifying patterns, and detecting outliers or missing values. Following EDA, we will preprocess the data, which may include steps such as normalization, scaling, feature selection, and handling missing values. Once the data is preprocessed, we will proceed to test various machine learning models on it. The dataset that we will be using contains several columns, including those related to the target variable as well as potential predictors. We will thoroughly analyze each column to determine its relevance and potential impact on the model's accuracy. By leveraging EDA, preprocessing, and testing various machine learning models, we aim to create a model that can accurately predict the target variable based on the available data.

The dataset that we will be using was obtained through a survey and contains authentic responses from a diverse set of loan applicants. This dataset adheres to a tabular format and is suitable for use with the chosen ANN model for evaluation purposes. The data was collected specifically for this study and will be used to train and test our classification model. The target variable for the classification problem is Personal Loan, which indicates whether a loan was accepted or not. This variable will be our main focus throughout the analysis.

To facilitate the analysis, we will be importing important libraries into the code. One such library is Seaborn, which will be used for data visualization purposes. Seaborn is a popular Python library that provides a variety of visualization tools to explore and present data effectively. Another crucial library that we will be using is Pandas, which is a powerful data manipulation library in Python. Pandas will allow us to manipulate, clean, and preprocess the dataset in various ways, making it easier to train and test our classification model. Overall, these libraries will be instrumental in analyzing the data and developing a robust classification model.

```
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   ID                  5000 non-null   int64
 1   Age                 5000 non-null   int64
 2   Experience          5000 non-null   int64
 3   Income              5000 non-null   int64
 4   ZIP_Code            5000 non-null   int64
 5   Family              5000 non-null   int64
 6   CCAvg               5000 non-null   float64
 7   Education           5000 non-null   int64
 8   Mortgage            5000 non-null   int64
 9   Personal_Loan       5000 non-null   int64
 10  Securities_Account  5000 non-null   int64
 11  CD_Account          5000 non-null   int64
 12  Online              5000 non-null   int64
 13  CreditCard          5000 non-null   int64
```

Figure 2: Dataset columns

4.2. Data description

The dataset we will be using contains several columns, each with a specific business meaning. The ID column serves as a unique identifier for each customer. The Personal Loan column indicates whether a personal loan was approved for the customer, with 1 indicating approval and 0 indicating denial. Age represents the customer's age, while Experience refers to the number of years of professional experience they have. Income represents the customer's annual income, while Zip code corresponds to their home address zip code. Family refers to the size of the customer's family, and CCAvg indicates their average spending on credit cards per month. Education is another important column that specifies the customer's education level, with 1 indicating undergraduate, 2 indicating graduate, and 3 indicating advanced/professional. Mortgage represents the value of the customer's house mortgage, while Securities indicates whether the customer has a securities account with the bank, with 1 indicating yes and 0 indicating no. CDAccount specifies whether the customer has a certificate of deposit with the bank, with 1 indicating yes and 0 indicating no. Online indicates whether the customer uses Internet banking facilities, with 1 indicating yes and 0 indicating no. Finally, CreditCard indicates whether the customer uses a credit card issued by the bank, with 1 indicating yes and 0 indicating no. These columns will be used to train and test our classification model.

4.3. Data preparation

The objective of this study is to develop a predictive model that can accurately determine whether a personal loan will be approved or rejected. The target variable for this analysis is the 'Personal Loan' column, which takes a binary value of 1 for approved loans and 0 for rejected loans. Our predictors include demographic and financial information, such as 'Age', 'Experience', 'Income', 'ZIP_Code', 'Family', 'CCAvg', 'Education', 'Mortgage', 'Securities_Account', 'CD_Account', 'Online', and 'CreditCard'.

To evaluate the effectiveness of our predictive model, we will split the dataset into two subsets: a training set and a testing set. The training set will be used to train the model, while the testing set will be used to evaluate the accuracy of the model's predictions. Specifically, we will allocate 70% of the data for training and 30% for testing. This approach will enable us to assess the model's generalization ability and identify any potential issues with overfitting or underfitting.

```
bank_df = pd.read_csv("loan_history.csv")
bank_df.shape

# data preparation
# separate Target Variable and Predictor Variables
X = bank_df.drop(columns=["Personal_Loan"])
y = bank_df["Personal_Loan"]

y = to_categorical(y)

# split training and testing data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

# training and testing datasets
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

Figure 3: Load dataset and split training/testing data

4.4. Creating the Deep Learning model

In this study, an Artificial Neural Network (ANN) classification model was developed using the sampled data. The output layer of the ANN comprises a single neuron, as the classification problem at hand is binary. In cases where there are multiple classes, the number of neurons in the output layer must be chosen to match the number of classes, with each neuron providing the probability estimate for that specific class. The predicted class is then the one with the highest probability estimate.

The model architecture includes two hidden layers, with the first layer consisting of ten neurons and the second layer consisting of six neurons. The output layer has one neuron, which outputs the probability estimate for class “1”.

The use of Artificial Neural Networks (ANNs) can provide numerous benefits, including their adaptive nature, enhanced learning ability, gradual corruption, and distributed information storage. ANNs can modify themselves post-initial training with subsequent information, learn events on their own, and perform multiple tasks simultaneously due to their node strength. In addition, ANNs are fault-tolerant, meaning that a fault in one or a handful of cells in the network doesn't prevent it from delivering output, and corruption tends to occur gradually over time, rather than all at once. [4]

When using ANNs, there are several important hyperparameters to consider, such as the number of units or neurons in a layer, the number of input predictors, the algorithm used to decide the value for each weight, the activation function for calculations inside each neuron, the optimizer that helps to find the optimum values of each weight, the batch size for the data passed to the network in one go, and the number of epochs for adjusting weights. These hyperparameters must be carefully selected through hyperparameter tuning to prevent overfitting or underfitting and achieve optimal performance.

To achieve the results of the study, the high-level open-source deep learning API, Keras, will be used. Keras allows for fast experimentation with deep neural networks and simplifies the process of building and training complex deep learning models. In a sequential model using dense layers, the output of one layer serves as the input to the next, with each neuron in the layer connected to every neuron in the previous layer. Dense layers are often used for feature extraction in deep learning models, and the number of neurons and activation functions used in each layer can be tuned to improve model performance. [5]

```python
#  Compilation and training of deep learning model
# sequential model
ann_model = keras.Sequential()

# adding dense layer
ann_model.add(Dense(250, input_dim=13, kernel_initializer='normal', activation='relu'))
ann_model.add(Dropout(0.3))
ann_model.add(Dense(500, activation='relu'))
ann_model.add(Dropout(0.3))
ann_model.add(Dense(500, activation='relu'))
ann_model.add(Dropout(0.3))
ann_model.add(Dense(500, activation='relu'))
ann_model.add(Dropout(0.4))
ann_model.add(Dense(250, activation='linear'))
ann_model.add(Dropout(0.4))

# adding dense layer with softmax activation/output layer
ann_model.add(Dense(2, activation='softmax'))
ann_model.summary()
```

Figure 4: Compilation and training of ANN deep learning model

```python
history = ann_model.fit(X_train, y_train, epochs=20, validation_split=0.2, verbose=1)

predictions = ann_model.predict(X_test)
predict = []

for i in predictions:
    predict.append(np.argmax(i))

y_test = np.argmax(y_test, axis=1)

f1_test = metrics.f1_score(y_test, predict)
prec = metrics.precision_score(y_test, predict)
rec = metrics.recall_score(y_test, predict)
acc = metrics.accuracy_score(y_test, predict)

print ("F1 Score: {:.4f}.".format(f1_test))
print ("Precision: {:.4f}.".format(prec))
print ("Recall: {:.4f}.".format(rec))
print ("Accuracy: {:.4f}.".format(acc))


print(metrics.classification_report(y_test, predict))
```

Figure 5: The accuracy of the final trained model on the testing data

4.5. Model performance

In deep learning, the effectiveness of a neural network model is evaluated using various metrics. These metrics provide different insights into the performance of the model in predicting the outcomes of a test set. Common metrics include: accuracy, precision, recall, and F1 score. [6]

Accuracy refers to the percentage of correct predictions made by the model on the test set. Precision measures the proportion of true positive predictions over the total number of positive predictions, providing an evaluation of the quality of positive predictions. Recall measures the proportion of true positive predictions over the total number of actual positive cases, reflecting how well the model identifies positive cases. [6]

The F1 score is the harmonic mean of precision and recall, providing a balanced measure of the model's accuracy. In order to assess the performance of a deep learning model, custom functions can be developed to calculate these metrics. By doing so, we can accurately evaluate the performance of our model and make necessary adjustments to improve its effectiveness. [6]

The subsequent section presents the prediction metrics outcomes

```
F1 Score: 0.9011.
Precision: 0.9318.
Recall: 0.8723.
Accuracy: 0.9544.
```

Figure 6: ANN Model metrics

In order to assess the efficacy of the model, it was subjected to a historical loan application dataset, yielding a 95.4% accuracy. This suggests that the model aptly classified the financial condition of banks on the basis of pertinent indicators and historical data.

4.6. Comparison of Model Performance across Different Neural Network Algorithms

The purpose of this section is to compare the performance of two different neural network algorithms: Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN). The models were trained and evaluated using a dataset of customer credit history and loan application information to predict creditworthiness.

Our results show that both ANN and CNN models achieved high levels of accuracy, with the ANN model achieving an accuracy of 95.4% and the CNN model achieving an accuracy of 94.3%. However, the CNN model demonstrated slightly better performance in precision, achieving a precision score of 0.94 compared to the ANN model's precision score of 0.93.

The relatively similar performance of the ANN and CNN models in this study suggests that both algorithms are effective for creditworthiness prediction. However, the CNN model may have a slight advantage in identifying spatial and temporal patterns in the data, which could potentially improve its ability to classify creditworthiness accurately.

Further research is needed to explore the potential of both ANN and CNN algorithms in other domains and to investigate their scalability and interpretability.

Table 1

ANN vs. CNN Algorithms [7]

Types of NN	ANN	CNN
Basics	One of simplest types of NN	One of the most popular types of NN
Structural Layout	Its simplicity comes from its feed forward nature – information flows in one direction only	Its structure is based on multiply layers of nodes including one or more convolutional layers
Data Type	Fed on tabular and text data	Relies on image data
Complexity	Simple in contrast with the other two models	More powerful than the other two
Commendable Feature	Ability to work with incomplete knowledge and high fault tolerance	Accuracy in recognizing images
Feature type: Spatial recognition	No	Yes
Feature type: Recurrent connections	No	No
Main Drawback	Hardware dependence	Large training data required
Uses	Complex problem solving such as predictive analysis	Computer vision including image recognition

Before implementing the CNN code, it is necessary to reshape the input data into 4D arrays, denoting samples, rows, columns, and channels. For this particular implementation, the input data will be reshaped to (n_samples, 13, 1, 1), where 13 signifies the number of features, and 1 represents the number of channels.

The Keras API will be employed once again to develop the CNN model.

```
from keras.layers import Dense, Flatten, Conv2D, Dropout

# normalize the input data
sc = StandardScaler()
X_train = sc.fit_transform(X_train.reshape(-1, 13)).reshape(-1, 13, 1, 1)
X_test = sc.transform(X_test.reshape(-1, 13)).reshape(-1, 13, 1, 1)

# create the CNN model
cnn_model = keras.Sequential()
cnn_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu', input_shape=(13, 1, 1)))
cnn_model.add(Dropout(0.3))
cnn_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
cnn_model.add(Flatten())
cnn_model.add(Dense(128, activation='relu'))
cnn_model.add(Dense(2, activation='softmax'))
```

Figure 7: Compilation and training of CNN deep learning model

The CNN model was tested, and the subsequent performance results were obtained:

```
F1 Score: 0.8988.
Precision: 0.9411.
Recall: 0.8799.
Accuracy: 0.9434.
```

Figure 8: CNN Model Metrics

When running both ANN and CNN algorithms multiple times, the obtained values are often similar. However, the selection between these two algorithms is dependent on the problem type, data nature, and neural network architecture. ANN is a feedforward network that connects the input layer to the output layer through hidden layers, and is best suited for handling tabular data or simple structured data. On the other hand, CNN is a deep learning network designed to extract features from images or data with a grid-like structure. It extracts patterns in data by using filters that slide over the data to extract features.

In general, CNN algorithms perform better on image data, while ANN is more appropriate for tabular data. However, the performance of ANN and CNN algorithms may be comparable in certain cases, depending on the dataset and the neural network architecture used.

If both ANN and CNN algorithms are implemented on the same dataset using the same set of hyperparameters and performance metrics, and yield the same performance, we can conclude that the architecture of the neural network does not significantly impact the performance, and both algorithms can be used interchangeably for that specific problem. However, it is important to note that this conclusion is specific to this particular problem, and may not be generalizable to other problems or datasets.

4.7. Testing

After training a model on a historical dataset of loan applications, the next step was to assess its performance on actual loan applicants. To do this, we conducted a survey to collect data from a diverse set of loan applicants. The chosen model for this evaluation was ANN, as it is well-suited for tabular data. We selected a sample of 100 respondents from the survey, and the model's predicted outcomes were compared with the actual outcomes for these applicants. The analysis revealed that only 8% of the model's predictions were incorrect, indicating a high degree of accuracy in predicting loan outcomes for this sample.

While these results are promising, further testing and validation on larger and more diverse samples of loan applicants will be necessary to determine the model's efficacy in real-world loan application processes within the banking industry. Nonetheless, the model's demonstrated success suggests potential for practical use in the field.

5. Regression analysis

Regression analysis can be used for credit scoring assessment, which involves predicting the likelihood of a borrower defaulting on a loan or being delinquent in making loan payments. In this context, regression analysis can be used to create a credit scoring model that predicts the probability of default based on a set of predictor variables. [8]

The predictor variables used in the regression analysis were the same as those used in the development of the neural network (NN) model. These factors were carefully selected based on their known influence on credit scoring assessment and included variables such as income, credit history, age, employment status, and loan amount, among others. The regression analysis aimed to establish a statistical relationship between these predictor variables and the target variable, which was the creditworthiness of loan applicants. By analyzing the coefficients and significance levels of the predictor variables, we were able to gain insights into the relative importance of each variable in determining creditworthiness and to develop a predictive model for credit scoring assessment.

Once the regression model is developed, it can be used to assign a credit score to individual loan applicants. This score indicates the level of risk associated with lending to that borrower and can inform the lender's decision-making process. A higher credit score indicates a lower risk of default, while a lower credit score indicates a higher risk.

The regression model presented can be represented as follows:

$$
\begin{aligned}
y = \beta_0 + \beta 1x1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 \\
+ \beta_5 x_5 + \beta_6 x_6 + \beta_7 x_7 \\
+ \beta_8 x_8 + \beta_9 x_9 + \beta_{11} x_{10} \\
+ \beta_{12} x_{12} + \beta_{13} x_{13} + \beta_{14} x_{14}
\end{aligned}
$$

Here, y represents the dependent variable, and x1 to x14 represent the independent variables. The model aims to estimate the value of y based on the values of x_1 to x_{14}. The β_1 to β_{14} are the coefficients that represent the relationship between each independent variable and the dependent variable. The regression model can be used to assess the creditworthiness of loan applicants by using the independent variables that were selected based on their predictive power in the development of the neural network model. The coefficients of the model can be interpreted as the change in *y* for a one-unit change in the corresponding independent variable, holding all other independent variables constant.

Predictor	Coefficient	Estimate	Standard Error	t-statistic	p-value
Constant	β_0	0.4461	0.1646	2.71	0.0068
x1	β_1	0	0	-0.7727	0.4397
x2	β_2	-0.0036	0.0028	-1.264	0.2063
x3	β_3	0.0036	0.0028	1.2778	0.2014
x4	β_4	0.022	0.0003	67.6574	0
x5	β_5	0	0	-3.1925	0.0014
x6	β_6	-0.0305	0.003	-10.1177	0
x7	β_7	-0.0029	0.0025	-1.1575	0.2471
x8	β_8	-0.038	0.0044	-8.6895	0
x9	β_9	0	0	-0.0823	0.9344
x10	β_{10}	0.1929	0.0145	13.2601	0
x11	β_{11}	-0.0064	0.0117	-0.547	0.5844
x12	β_{12}	0.0093	0.0169	0.5484	0.5835
x13	β_{13}	-0.0185	0.007	-2.6456	0.0082
x14	β_{14}	-0.0035	0.0078	-0.451	0.652

R-Squared:	r^2 =0.7203
Adjusted R-Squared:	r^2_{adj} =0.7196
Residual Standard Error:	0.2373 on 4985 degrees of freedom.
Overall F-statistic:	917.2036 on 14 and 4985 degrees of freedom.
Overall p-value:	0

Figure 9: Summary of overall fit

Regression analysis has proven to be an effective tool in credit scoring assessment, enabling lenders to make well-informed decisions about loan applicants by relying on quantifiable data instead of subjective judgments. It is crucial to note that the accuracy and usefulness of the credit scoring model heavily rely on the quality and relevance of the predictor variables employed in the analysis.

6. Conclusions

This study highlights the importance of utilizing accurate and reliable credit scoring methods in risk management. The findings demonstrate that machine learning models outperform traditional regression analysis in predicting credit scores and identifying risky borrowers. Fintech companies can benefit from incorporating machine learning techniques in their credit scoring processes to mitigate risk and make informed lending decisions. However, it is crucial to ensure the quality and relevance of predictor variables employed in the analysis to enhance the effectiveness of the credit scoring model. Future research should investigate the potential of combining machine learning with other credit scoring techniques to further improve risk management in lending.

7. References

[1] Van Gestel, T., et al.: Linear and nonlinear credit scoring by combining logistic regression and support vector machines. (2005): J. Credit Risk 1(4)

[2] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. (2015): Nature 521(7553), 436

[3] Hu, Y. H., & Hwang, J. (2002). Handbook of Neural Network Signal processing.

[4] Lessmann, S., Baesens, B., Seow, H.-V., Thomas, L.C.: Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. (2015): J. Oper. Res. 247(1), 124–136

[5] Ketkar, Nikhil, and Nikhil Ketkar. "Introduction to keras." Deep learning with python: a hands-on introduction (2017): 97-111.

[6] Yacouby, Reda, and Dustin Axman. "Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models." In Proceedings of the first workshop on evaluation and comparison of NLP systems, pp. 79-91. 2020.

[7] Kareem, Shahab, Zhala Jameel Hamad, and Shavan Askar. "An evaluation of CNN and ANN in prediction weather forecasting: A review." Sustainable Engineering and Innovation 3, no. 2 (2021): 148.

[8] Freund, Rudolf J., William J. Wilson, and Ping Sa. Regression analysis. Elsevier, 2006.