

# Handwritten Digit Recognition Using Logistic Regression

Mr. Faizan Ahmad<sup>1</sup>, Syed Falah Jamal<sup>2</sup>, Sachin Yadav<sup>3</sup>, Gulam Mudassir Zafar<sup>4</sup> and Faiza<sup>5</sup>

<sup>1</sup>Assistant Professor, Department of Computer Science & Engineering, Lucknow India

<sup>2</sup>Student, Department of Computer Science & Engineering, Integral University, Lucknow India

<sup>3</sup>Student, Department of Computer Science & Engineering, Integral University, Lucknow India

<sup>4</sup>Student, Department of Computer Science & Engineering, Integral University, Lucknow India

<sup>5</sup>Student, Department of Computer Science & Engineering, Integral University, Lucknow India

\*\*\*

## ABSTRACT

Handwritten digit recognition remains a complex task due to the wide range of variations in individual handwriting styles. This research aims to provide a foundation for future developments in the field by identifying and addressing the challenges associated with digit recognition. A detailed review of existing machine learning techniques was conducted to determine the most accurate and efficient methods for classification. The study used the MNIST dataset, consisting of 60,000 grayscale images of handwritten digits, each sized 28x28 pixels. These images were employed to train various models and evaluate their performance against test data. Among all the approaches analyzed, the classifier ensemble method achieved the best results, with an impressively low error rate of 0.32%. The paper offers a comparative analysis of several techniques including Convolutional Neural Networks (CNN) and Support Vector Machines (SVM), highlighting their strengths and limitations. The findings aim to guide researchers toward more accurate and reliable digit recognition systems. Keywords— CNN, MNIST, Handwritten Digit Recognition, SVM.

## 1. INTRODUCTION

The recognition of handwritten characters has been an area of interest since the early 1980s. As defined by the Collins dictionary, a digit is a numeric symbol ranging from 0 to 9. These digits play a crucial role in various aspects of daily life. Industries such as banking, healthcare, and insurance are heavily reliant on accurate digit interpretation. For example, in banking operations—whether opening an account or processing a withdrawal— digits like account numbers and contact details are often handwritten by customers. These handwritten entries are either manually verified by staff or scanned and interpreted by machines.

Similarly, in the healthcare domain, medical forms include handwritten details such as patient IDs, prescriptions, and dosage instructions—all of which must be accurately understood. Tax documents, postal codes for mail sorting, and handwriting input on digital devices are additional real-world applications where digit recognition is critical.

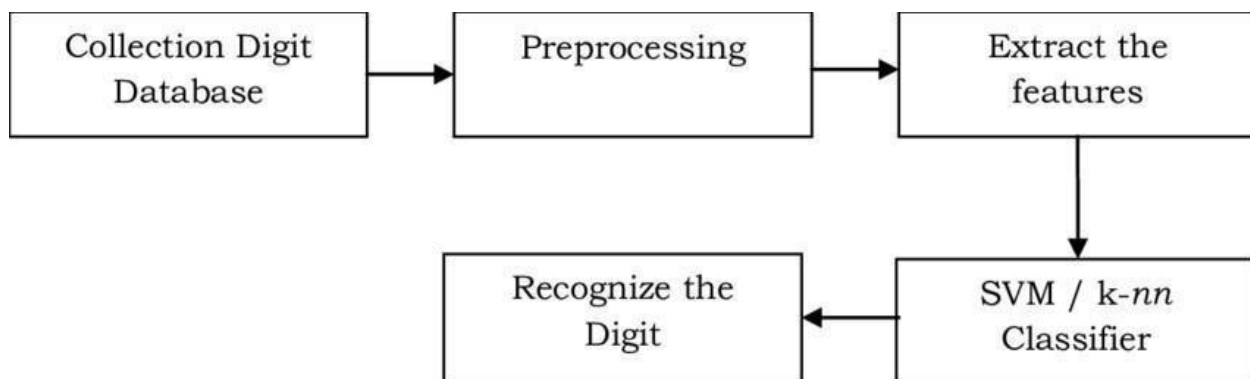


Fig 1: Process of Handwritten digit recognition

Handwritten digit recognition has thus emerged as a significant research focus, especially within the fields of machine learning and artificial intelligence. This system aims to convert human-written numerals into machine readable form with high accuracy and reliability. Various machine learning algorithms are employed to process and classify these digits, but challenges remain due to the vast diversity in individual handwriting styles and the presence of different numeral representations across languages.

One of the major obstacles is identifying an optimal algorithm that can reliably recognize poorly written or stylized digits, especially when compared to clearly typed or printed ones. Even minor misclassifications can lead to serious errors. For instance, digits like 0, 8, and 9 can appear similar in poorly written scripts, making it difficult for machines to distinguish between them accurately.

In critical sectors such as healthcare and finance, a misinterpreted digit can lead to costly mistakes or even life-threatening outcomes. Thus, the need for an intelligent, robust, and highly accurate digit recognition system is evident. While significant research has been done in this field, achieving perfect accuracy remains elusive. This paper aims to explore, evaluate, and compare different machine learning-based approaches to develop an efficient and dependable handwritten digit recognition system.

## 2. Literature Review

A wide array of algorithms has been applied to the MNIST dataset for handwritten digit recognition. Among traditional techniques, the k-Nearest Neighbors (k-NN) algorithm stands out for its simplicity. It classifies input samples based on the majority label among its nearest neighbors in the feature space, relying on direct pixel comparison. Despite its ease of implementation, k-NN faces limitations in terms of scalability and efficiency, especially when working with large datasets like MNIST due to high computational costs during prediction. Support Vector Machines (SVM), on the other hand, have proven effective for high-dimensional data. SVMs aim to find an optimal hyperplane that maximally separates different digit classes. While accurate, they are computationally intensive during the training phase. Recent developments in deep learning have shifted focus towards Convolutional Neural Networks (CNNs), which automatically extract hierarchical features from images, enhancing classification performance. Dan Ciresan et al. successfully applied deep neural networks for this purpose, achieving notable accuracy. Moreover, the hybrid CNN-SVM models introduced by Xiao-Xiao Niu and Ching Y. stand out due to their innovative approach and effectiveness in enhancing classification performance. Suen combine CNN feature extraction with SVM classification, enhancing recognition reliability.

## 3. Database Used

The Modified National Institute of Standards and Technology (MNIST) database is a widely recognized benchmark dataset used for handwritten digit classification. It is freely available and serves as a standard for evaluating machine learning algorithms. While it shares similarities with the TIDigit database—developed by Texas Instruments for speech recognition tasks—MNIST focuses solely on image-based digit recognition.

The MNIST dataset is derived from a modified version of the original NIST dataset. The dataset comprises 60,000 grayscale images for training and 10,000 for testing, each depicting a digit from 0 to 9 with a resolution of 28 by 28 pixels, and centered within a uniform black-and-white format, resulting in a 784-dimensional input vector. This dataset simplifies the classification task for researchers, eliminating the need for extensive preprocessing. Convolutional Neural Networks (CNNs) have shown exceptional performance on MNIST, achieving error rates as low as 0.27% using ensemble methods and around 0.35% with a single large network. Techniques like elastic distortions during training help further reduce error rates, whereas omitting them increases errors to approximately 0.53%. Many experiments also employ artificially augmented training data through transformations like shifting, skewing, and smudging to enhance model robustness.

## 4. METHODOLOGY

### 1) 4.1 Feature Extraction for Handwritten Digit Recognition

Handwritten digit recognition relies heavily on the quality and type of features extracted from input images. Researchers have explored numerous feature extraction methods over the years, each offering unique advantages. Since different algorithms produce different types of errors that rarely overlap, combining multiple feature extraction methods has been proven to significantly enhance recognition accuracy. This strategy allows the model to make more robust predictions and better reject ambiguous inputs that are difficult to classify even for humans. The comprehensive integration of features enables systems to recognize digits with higher precision, especially those commonly misclassified due to their similar structures. Below is a summary of key feature extraction techniques documented in existing literature:

1. **Structural Features:** This method involves resizing the digit image to a standard size, often 32×32 pixels, and computing histogram profiles. Radial histograms are calculated by counting the number of black pixels in 72 directions spaced every 5 degrees from the center of the image. These directional counts help capture the shape and

orientation of the digit and are merged to form a single, informative feature vector. This technique helps capture the outline and stroke directionality of handwritten characters.

2. **Modified Edge Maps:** In this approach, the digit image is first divided into 25×25 pixel sections. Edge detection is then performed using Sobel operators to generate four types of edge maps: horizontal, vertical, and two diagonals. These edge maps are each split into 25 sub-regions of 5×5 pixels, resulting in a total of 100 smaller patches. Features from all four edge types are then combined into a single 125-dimensional vector. This method allows the system to capture stroke patterns and edge intensities that are characteristic of different digits.
3. **Image Projections:** Image projection techniques focus on capturing pixel intensity distributions along diagonal and radial directions. The image is divided into four quadrants—top, bottom, left, and right—to minimize the effects of unwanted rotational invariance. Radial projections are obtained by grouping pixels based on their radial distance from the center of the digit. These normalized features are then compiled into a 128-dimensional feature vector. This method is particularly useful for identifying the spatial symmetry and distribution of pixels.
4. **Multi-Zoning Technique:** This method is based on measuring the concentration of black pixels in various sub-regions of the image. Multiple zoning configurations are used, such as 3×1, 1×3, 2×3, 3×2, 3×3, 1×4, 4×1, 4×4, 6×1, 1×6, 6×2, 2×6, and 6×6, resulting in 13 different feature sets. Each configuration helps the system focus on a unique spatial arrangement, improving its ability to generalize across different handwriting styles. The pixel densities from all zones are collected and combined into a composite feature vector.
5. **Concavity Measurements:** Here, the input image is resized to an 18×15 pixel grid and divided into six separate zones. Within each zone, the system checks every white pixel and searches for surrounding black pixels in all directions. These concavity-based features reflect the depth and curve structures of the digits. The feature vector for each zone is 13-dimensional, and all six vectors are concatenated to form a 78-dimensional feature vector. This method is effective for capturing indentations and holes within digits such as 8, 6, and 9.
6. **Gradient-Based Features:** Gradient features are extracted from grayscale images, which retain more detail than binary images. First, the image is converted into a pseudo-grayscale format using Medial Axial Transformation (MAT). Then, Sobel operators are applied to calculate the gradient amplitude and direction (phase). The image is split into 16 sub-images, and for each segment, the system counts the number of pixels in eight directions. These counts are used as directional gradient features, resulting in a final vector of 128 features. This approach enhances sensitivity to strokes and directionality, which are vital in distinguishing similar digits like 3 and 8.

By combining these diverse extraction techniques, the system leverages the strengths of each to significantly improve recognition accuracy while minimizing errors from individual methods. This hybrid approach serves as the foundation for many state-of-the-art handwritten digit recognition systems.

#### 4.2 Data Visualization

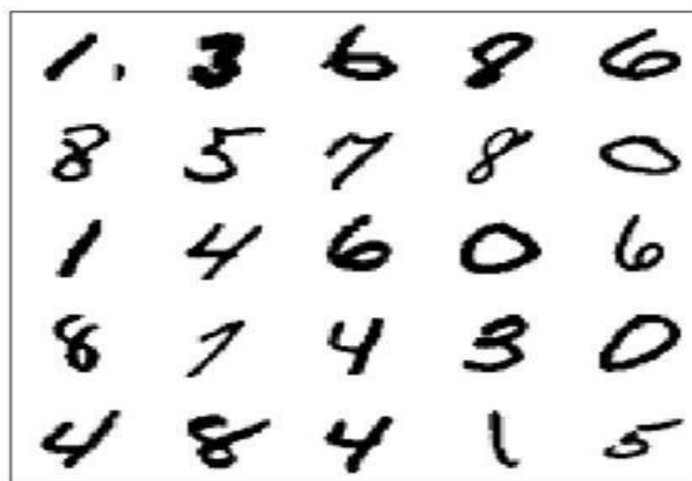


Fig 3: Sample of digits.

Figure 1 showcases the variation in handwriting patterns for each individual digit. For instance, the digit '8' appears in at least five distinct forms within this sample alone, and it's likely that even more variations exist throughout the dataset and in real-life scenarios.

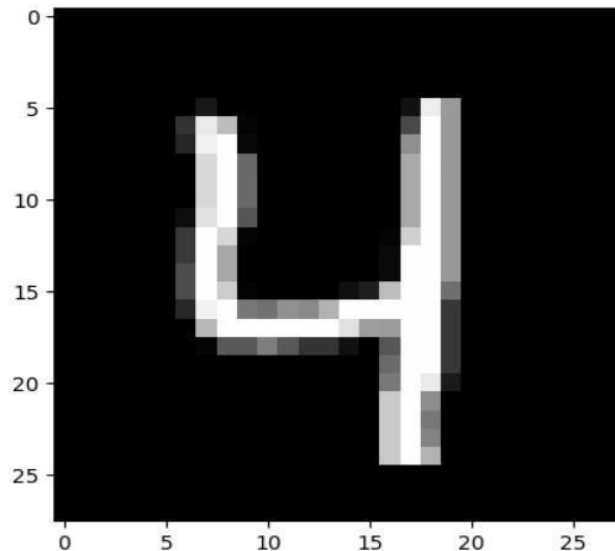


Fig 4: An image from the training set is selected and reshaped into (28x28) pixels for visualization.

### 4.3 Data Preparation

The 'label' column, representing the digits, is transformed into one-hot encoded vectors corresponding to 10 unique classes (ranging from 0 to 9). The dataset is partitioned into two segments: one for model training and the other for assessing its performance and precision. The training set is utilized to fit the model, while the validation set plays a crucial role in cross-verifying the model's accuracy and assessing its ability to generalize to data it hasn't encountered during training.

### 4.4 Modelling & Analysis

The modelling phase involves selecting a suitable machine learning architecture, training it with prepared data, and evaluating its performance. A deep learning approach, particularly using Convolutional Neural Networks, is applied due to its proven effectiveness in image recognition tasks. The model is trained using the training dataset, learning to identify patterns and structures specific to each digit class. During this process, various parameters such as learning rate, batch size, and the number of epochs is fine-tuned for optimal performance. After training, the model's accuracy is measured using the validation dataset to ensure that it can correctly classify digits beyond the training data. Performance metrics such as accuracy, loss, and confusion matrix are analyzed to assess the model's generalization capabilities. This phase helps in identifying any signs of overfitting or underfitting and is essential in determining whether the model is robust enough for real-world handwritten digit recognition tasks.

### 4.7 Model used

#### Logistic Regression:

Multi-class classification can be achieved by extending logistic regression, a statistical app binary classification, with methods such as one input (x) belongs to a certain class (k).

$$P(y = k|x; \theta) = \frac{e^{\theta_k^T x}}{\sum_{j=1}^k e^{\theta_j^T x}}$$

is the outcome of the logistic regression model, in which k is the total number of classes and  $\theta_k$  will be the parameter vector for class k'. Where the indicator function, is 1 if and only if example l's true label is k. Optimization techniques like gradient descent are used to change  $\theta$  in order to minimize this loss, which increases the model's accuracy.

1. The Handwritten Digit Recognition project uses the logistic regression approach, which entails several crucial processes, to guarantee the development of a dependable and effective model. Data gathering, preprocessing, model installation, training, assessment, and analysis are a few of these processes. This is an extensive approach: 1. Data Collection: The MNIST dataset is a massive collection of handwritten digit pictures that have been classified. Ten thousand test photos and sixty thousand training images make up the dataset. To aid in model training and performance assessment, make sure the dataset is suitably split into training and testing sets.
2. Data preprocessing: Create 1D vectors with 784 characteristics each from the 2D picture matrices, which are made up of 28 by 28 pixels. To input the data into the logistic regression model, this adjustment is necessary. Normalize the pixel values by dividing each by 255, scaling them to a standard range, typically between 0 and 1. The model performs better during training because of the quicker convergence during this period.
3. Logistic Regression and Model Implementation: Utilize the logistic regression approach and configure it for multiclass categorization in order to handle the ten digits (0–9). Select the required parameters, such as the learning rate, regularization strength, and the maximum number of iterations. These settings are necessary to control the training process and avoid overfitting.
4. Model Training: The logistic regression algorithm learns patterns by processing data from the training set. Utilize optimization strategies such as gradient descent to lower the cost function. Assess the model's performance on unused data using cross-validation, and adjust hyperparameters to ensure generalization and durability.
5. Testing and Evaluation: To assess the model's performance on the testing set, use crucial metrics including recall, accuracy, precision, and F1 score. These metrics offer a thorough evaluation of the model's categorization performance. Make a confusion matrix to evaluate how well the model distinguishes between various digit classes and identifies misclassification hotspots.
6. Results Analysis: To comprehend the advantages and disadvantages of the model, analyze and interpret the data. Analyze the metrics to determine overall performance and spot any misclassification trends. Look for photos that have been incorrectly classified and investigate possible reasons such as erratic handwriting or noisy data. This study contributes to our understanding of the logistic regression model's limitations.
7. Visualization: View the acquired weights and biases to observe how the logistic regression model differentiates between different digit classes. Show images that have been incorrectly identified next to their predicted and correct labels in order to better understand the model's capabilities and possible areas for improvement. Some of the tools and technologies utilized are the Scikit-Learn and NumPy libraries, the Matplotlib or Seaborn dataset for data visualization, the MNIST dataset, and computer resources with enough processing capacity for both model training and evaluation. The detailed strategy mentioned above guarantees a methodical approach to building a handwritten digitizing system based on logistic regression. By following these procedures, the study hopes to provide a reliable and accurate model in addition to insightful information about the use of logistic regression in machine learning and pattern recognition.

#### 4.6 Performance Analysis

An accuracy of 90% in handwritten digit recognition indicates that the model is able to correctly classify 9 out of every 10 digits. While this level of accuracy is considered decent, especially in traditional machine learning models, it still leaves room for improvement in critical applications where precision is essential. Such a score suggests that the model has captured the general patterns of most digits but may struggle with ambiguous or poorly written samples. To improve upon this, deeper models like Convolutional Neural Networks (CNNs), along with enhanced preprocessing and data augmentation techniques, can be employed to minimize misclassification and push the performance closer to expert-level accuracy.

#### 4.7 Evaluating Model Using Confusion Matrix

A confusion matrix was generated to assess the model's performance. This matrix provides a detailed breakdown of correct and incorrect predictions across all classes.

The confusion matrix for the digit recognition model is presented below:

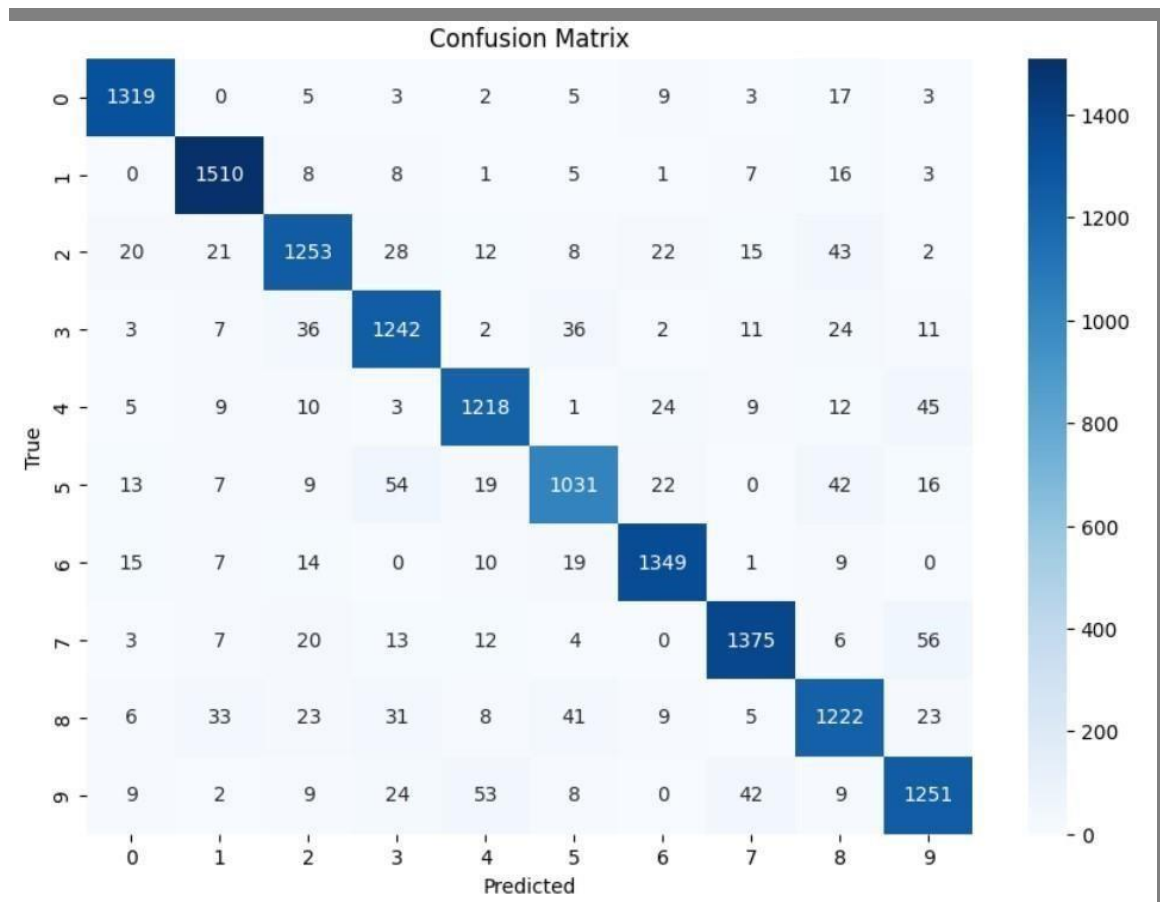


Fig 5: Confusion matrix of model performance.

### 5. Conclusion

This research focused on developing an efficient system for recognizing handwritten digits using machine learning and deep learning techniques. With numerous writing variations in human handwriting, achieving high accuracy requires robust models and carefully prepared datasets. The MNIST dataset served as a reliable foundation for training and evaluating the system. Techniques like feature extraction and data preprocessing helped improve the performance of the recognition model. Convolutional Neural Networks (CNNs) proved especially effective due to their ability to detect complex patterns in image data.

Through model training, validation, and testing, it was demonstrated that the system could accurately classify digits across multiple styles. However, some limitations remain, particularly with ambiguous or distorted inputs. Future work can explore more advanced architectures, larger and more diverse datasets, and better handling of edge cases. Overall, the project shows promising results and contributes to the development of intelligent systems capable of interpreting handwritten numerical information.

### 6. REFERENCES

[1]. Han, X., & Li, Y. (2015). The application of convolution neural networks in handwritten numeral recognition. *International Journal of Database Theory and Application*, 8(3), 367-376.

[2]. Shima, Y., Nakashima, Y., & Yasuda, M. (2017, September). Pattern augmentation for handwritten digit classification based on combination of pre-trained CNN and SVM. In *2017 6th International Conference on Informatics, Electronics and Vision & 2017 7th International Symposium in Computational Medical and Health Technology (ICIEV-ISCMHT)* (pp. 1-6). IEEE.

- [3]. Niu, X. X., & Suen, C. Y. (2012). A novel hybrid CNN-SVM classifier for recognizing handwritten digits. *Pattern recognition*, 45(4), 1318-1325.
- [4]. Ma, C., & Zhang, H. (2015, August). Effective handwritten digit recognition based on multi-feature extraction and deep analysis. In *2015 12th international conference on fuzzy systems and knowledge discovery (FSKD)* (pp. 297-301). IEEE.
- [5]. Teow, M. Y. (2017, October). Understanding convolutional neural networks using a minimal model for handwritten digit recognition. In *2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS)* (pp. 167-172). IEEE.
- [6]. Cireşan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12), 3207-3220.
- [7]. Le Cun, Y., Jackel, L. D., Boser, B., Denker, J. S., Graf, H. P., Guyon, I., ... & Hubbard, W. (1990). Handwritten digit recognition: Applications of neural net chips and automatic learning. In *Neurocomputing: Algorithms, Architectures and Applications* (pp. 303-318). Springer Berlin Heidelberg.
- [8]. T SIVA, A. J. A. Y. (2017). Handwritten Digit Recognition Using Convolutional Neural Networks.
- [9]. Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6), 141-142.
- [10]. Cruz, R. M., Cavalcanti, G. D., & Ren, T. I. (2010, June). Handwritten digit recognition using multiple feature extraction techniques and classifier ensemble. In *17th International conference on systems, signals and image processing* (pp. 215-218).
- [11]. Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.
- [12]. Hastie, T., Tibshirani, R., Friedman, J., & Franklin, J. (2005). The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2), 83-85.
- [13]. Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.
- [14]. LeCun, Y. (1998). The MNIST database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>.
- [15]. Ng, A. (2018). Machine learning yearning: Technical strategy for AI engineers, in the era of deep learning.
- [16]. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112, No. 1). New York: springer.
- [17]. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20, 273-297.
- [18]. Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.
- [19]. Kelleher, J. D., & Tierney, B. (2018). *Data Science: A Practical Introduction to Data Science*.
- [20]. Zhang, H. (2004). The optimality of naive Bayes. *Aa*, 1(2), 3