**THE UNIVERSITY OF INFORMATION TECHNOLOGY**

**VIETNAM NATIONAL UNIVERSITY IN HOCHIMINH CITY**



**REPORT FOR SUMMER CAMP**

**Topic: VIETNAMESE MOTORBIKE LICENSE PLATE RECOGNITION.**

**Report Writer**:

Võ Thị Kim Trang, from Hochiminh City, Vietnam. ***Student’s Code***: 19210048

***My University mail****:* [19210048@ms.uit.edu.vn](mailto:19210048@ms.uit.edu.vn)

***Personal mail***: [vothikimtrang0610@gmail.com](mailto:vothikimtrang0610@gmail.com)

**My Github link for this report’s project**:

<https://github.com/VoThiKimTrang06101997/Vietnamese_Motorbike_License_Plate_Recognition>

**🙡🙢** HoChiMinh City, Vietnam, July 19th, 2022 **🙠🙣**

**COMMENTS**

*……., date……...month……year 2022*

**Commentor**

*(Sign and Write fullname fully****)***

**Acknowledgement**

My name is Võ Thị Kim Trang, from Vietnam. I’m preparing to graduate from the University of Information Technology in HoChiMinh City – Vietnam National University.

I would like to say thank you and I am very grateful to the department of Computer Science of Peking University for organizing such a meaningful and valuable summer camp so that I can take an insights into the overview and knowledge of Computer Science as well as I can have a motivation to continue studying a master degree in Computer Science at the University of Information Technology in HoChiMinh City – Vietnam National University, doing research and upgrading my knowledge, profile. If I have an opportunity, I will try once to apply for other activities as well as the scholarship of Computer Science of Peking University in the future.

Thank you so much !!

**Summary**

Currently, the number of vehicles participating in traffic on the road is very large, leading to consuming a lot of human and material resources for the management of personal vehicles in the parking lot. Without a convenient tool, the management of personal vehicles is very time consuming, easy to cause confusion, damage to service users at parking lots.

To reduce the overloading on tasks such as collecting money, insuring cars, finding vehicles in parking lots, the world has developed automatic monitoring technology for vehicles. Thanks to the individuality of License Plate, the number of Vehicles License Plate that has become the main object used for research and development in this technology: Vehicles License Plate Recoginition / Detection.

Therefore, I want to choose this topic as a basic step in understanding more powerful monitoring tools such as vehicle control on the road or facial recognition ... which are being paid great attention by the world at the moment.

# LIST OF TABLES, IMAGES:

**List of tables:**

**Figure 5.1.1 – 2 :** **Post-normalization data. 45**

**Figure 5.1.1 – 3 :** **After calculating distance and rating. 45**

**Figure 6.2 – 1: Percentage of finding the license plate in the picture. 53**

**Figure 6.2 -10 : Percentage of wrong character recognition in the single-row plate. 56**

**Figure 6.2 -11 : Percentage of wrong character recognition in the two-row plate. 56**

**List of images:**

**Figure 1.2: Machine learning workflow. 10**

**Figure 1.3: Classification of Machine Learning. 11**

**Figure 1.3.1: Supervised Machine learning. 12**

**Figure 1.3.2: Unsupervised Machine learning. 13**

**Figure 1.3.3: Semi-supervised learning. 14**

**Figure 2.1: Linear Regression Algorithms. 17**

**Figure 2.2: Logistic Regresssion** **Algorithms. 18**

**Figure 2.3: Decision Tree Regresssion** **Algorithms. 18**

**Figure 2.4: SVM Algorithms. 19**

**Figure 2.5: Naive Bayes Algorithms. 20**

**Figure 2.6: KNN Algorithms. 20**

**Figure 2.7: K-Mean Algorithms. 21**

**Figure 2.8: Random Forest Algorithms. 22**

**Figure 2.9: Dimensionality Reduction Algorithms. 23**

**Figure 2.10: Boosting and AdaBoosting Algorithms. 23**

**Figure 3.2.3 - Main steps in license plate recognition. 26**

**Figure 4.1 – Identify and seperate license plates. 27**

**Figure 4.3.1 – 1: Structuring element example. 29**

**Figure 4.3.1 – 2: Erosion Operation. 29**

**Figure 4.3.1 – 3: Dilation Operation. 30**

**Figure 4.3.1 – 4: Opening Operation. 30**

**Figure 4.3.1 – 5: Closing Operation. 31**

**Figure 4.3.1 – 6: Top Hat Operation. 31**

**Figure 4.3.1 – 7: Black Hat Operation. 32**

**Figure 4.4: Image after using Increase Contrast level. 32**

**Figure 4.5.1: Noise. 33**

**Figure 4.5.2: Gauss filter matrix. 33**

**Figure 4.5.3: The result of using Gauss filter. 34**

**Figure 4.6.3: Binary with dynamic threshold image. 35**

**Figure 4.7-1: Non-maximum suppression. 37**

**Figure 4.7-2: Double threshold. 38**

**Figure 4.7-3: Image after detecting Canny edge. 38**

**Figure 4.8.1 - 1: Square Tracing Algorithm. 39**

**Figure 4.8.1 - 2: Square Tracing Algorithm runs correctly. 40**

**Figure 4.8.1 - 3: Square Tracing Algorithm runs wrongly. 40**

**Figure 4.8.1 - 4: Moore – Neighbor Algorithm. 41**

**Figure 4.8.1 - 5: Draw Contour with OpenCV. 42**

**Figure 4.9 - 1: Contour does not reach approximately the polygon. 42**

**Figure 4.9 - 2: Contour has already reached approximately the polygon. 43**

**Figure 5.1.1 - 1: Example of KNN Algorithm. 44**

**Figure 5.2 – 1 :** **Training dataset. 46**

**Figure 5.2 – 2 :** **License Plate before recognition. 47**

**Figure 5.2 – 3 :** **License Plate after recognition. 47**

**Figure 5.2 – 3 :** **License Plate are printed out with the comparison with the origin. 48**

**Figure 5.3 – 1 :** **Image obtained (right) after running the above function. 49**

**Figure 5.3 – 2 :** **Image obtained (right) after running the above function. 50**

**Figure 5.3 – 3 :** **Change the brightness (right). 51**

**Figure 5.3 – 4 :** **Image is rotated (right). 51**

**Figure 5.4 : Identify the area containing the number plate. 52**

**Figure 6.2 -2: Characters are not found. 54**

**Figure 6.2 -3: Five Characters are found . 54**

**Figure 6.2 -4: Nine Characters are found . 54**

**Figure 6.2 -5 :** **Get 1.5 high/wide threshold. 55**

**Figure 6.2 -6 :** **Get 1.4 high/wide threshold. 55**

**Figure 6.2 -7 : Contour approximation error. 55**

**Figure 6.2 -8 : Broken binary image. 56**

**Figure 6.2 -9 : The contour line is broken. 56**

**Figure 6.2 -12 : Original photo identifying 3 license plates. 57**

**Figure 6.2 -16 : Can't circle the character area. 58**

**MỤC LỤC**

[Acknowledgement 3](#_Toc109166945)

[Summary 3](#_Toc109166946)

[LIST OF TABLES, IMAGES: 4](#_Toc109166950)

[Chapter 1: The concept of machine learning 10](#_Toc109166951)

[1.1 The definition of machine learning: 10](#_Toc109166952)

[1.2 Machine learning workflow: 10](#_Toc109166953)

[1.3 Classification of Machine Learning: 11](#_Toc109166954)

[**1.3.1** **Supervised Machine Learning:** 12](#_Toc109166955)

[**1.3.2** **Unsupervised Machine Learning:** 13](#_Toc109166956)

[**1.3.3** **Semi-supervised learning:** 14](#_Toc109166957)

[1.4 Applications of Machine Learning: 15](#_Toc109166958)

[Chapter 2: 10 Algorithms in Machine Learning 17](#_Toc109166959)

[2.1 Linear Regression 17](#_Toc109166960)

[2.2 Logistic Regression: 17](#_Toc109166961)

[2.3 Decision Tree 18](#_Toc109166962)

[2.4 Support Vector Machine (SVM Algorithms) 19](#_Toc109166963)

[2.5 Naive Bayes Algorithms 19](#_Toc109166964)

[2.6 K-Nearest Neighbors (KNN) Algorithms: 20](#_Toc109166965)

[2.7 K-Means: 21](#_Toc109166966)

[2.8 Random Forest Algorithms: 22](#_Toc109166967)

[2.9 Dimensionality Reduction 22](#_Toc109166968)

[2.10 Gradient Boosting and AdaBoosting Algorithms 23](#_Toc109166969)

[Chapter 3: Overview and the obligation of the topic 24](#_Toc109166970)

[3.1 INTRODUCTION: 24](#_Toc109166971)

[**3.1.1** **Overview:** 24](#_Toc109166972)

[**3.1.2** **Thematic topic:** 24](#_Toc109166973)

[3.2 OVERVIEW OF THE PROBLEM OF IDENTIFICATION OF VEHICLES: 24](#_Toc109166974)

[**3.2.1** **Concept of license plate:** 24](#_Toc109166975)

[**3.2.2** **Image processing and Open CV:** 25](#_Toc109166976)

[**3.2.3** **Problem-solution:** 26](#_Toc109166977)

[Chapter 4: Location Detection and Separation of Vehicle Number Licene Plates. 27](#_Toc109166978)

[4.1 Problem-solution: 27](#_Toc109166979)

[4.2 Convert to grayscale image: 28](#_Toc109166980)

[4.3 Increase Contrast level: 28](#_Toc109166981)

[**4.3.1** **Morphological math:** 28](#_Toc109166982)

[4.4 Increase Contrast level: 32](#_Toc109166983)

[4.5 Reduce image noise with a Gaussian filter: 33](#_Toc109166984)

[**4.5.1** **Noise:** 33](#_Toc109166985)

[**4.5.2** **Gauss filter:** 33](#_Toc109166986)

[4.6 Image binary with dynamic threshold: 34](#_Toc109166987)

[**4.6.1** **Binary image:** 34](#_Toc109166988)

[**4.6.2** **Binary process:** 34](#_Toc109166989)

[**4.6.3** **Binary with dynamic threshold:** 35](#_Toc109166990)

[4.7 Canny Edge Detection: 35](#_Toc109166991)

[**a.** **Noise reduction:** 36](#_Toc109166992)

[**b.** **Gradient calculation:** 36](#_Toc109166993)

[**c.** **Non-maximum suppression:** 36](#_Toc109166994)

[**d.** **Double threshold:** 37](#_Toc109166995)

[**e.** **Result:** 38](#_Toc109166996)

[**4.8** **Filter license plate using Contour:** 39](#_Toc109166997)

[**4.8.1** **Several methods of finding contour:** 39](#_Toc109166998)

[**a.** **Square Tracing Algorithm:** 39](#_Toc109166999)

[**b.** **Moore – Neighbor Algorithm:** 40](#_Toc109167000)

[**c.** **Suzuki’s Tracing:** 41](#_Toc109167001)

[**4.9** **Filter number plate:** 42](#_Toc109167002)

[Chapter 5: Character segmentation, character recognition and the KNN algorithm 43](#_Toc109167003)

[5.1 Character segmentation: 43](#_Toc109167004)

[**5.1.1** **K - Nearest Neighbor (KNN) Algorithm:** 43](#_Toc109167005)

[5.2 Problem-solution: 46](#_Toc109167006)

[5.3 Methods to increase dataset diversity: 48](#_Toc109167007)

[Chapter 6: The result of performance 53](#_Toc109167008)

[6.1 How to measure and test: 53](#_Toc109167009)

[6.2 Result and Explanation: 53](#_Toc109167010)

[Chapter 7: CONCLUSIONS AND DEVELOPMENT ORIENTATIONS 59](#_Toc109167011)

[7.1 Conclusion: 59](#_Toc109167012)

[7.2 Developement Orientation: 59](#_Toc109167013)

[Chương 8: Preference 60](#_Toc109167014)

# The concept of machine learning

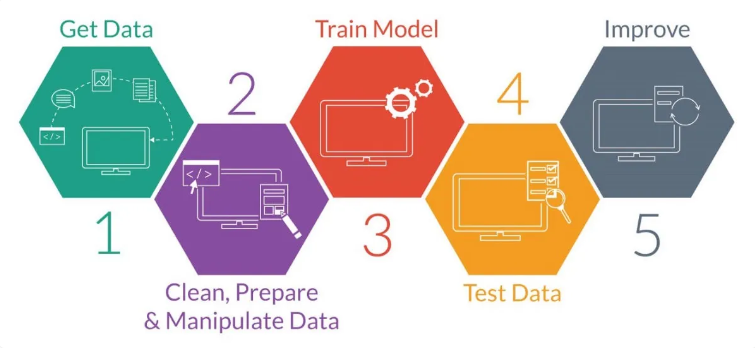
## The definition of machine learning:

Machine Learning, also known as Machine Learning, is an area of artificial intelligence (AI) and computer science. Machine Learning is concerned with studying and building techniques that allow systems to "learn" automatically from data to solve specific problems. Machine Learning focuses on using data and algorithms to mimic how humans learn, then gradually improving its accuracy.

Machine Learning is a key component of the burgeoning field of data science. Through the use of statistical methods, algorithms to make classifications or predictions, uncover important insights in data mining projects.

“Machine learning is the subfield of computer science that “gives computers the ability to learn without being explicitly programmed ”.

## Machine learning workflow:

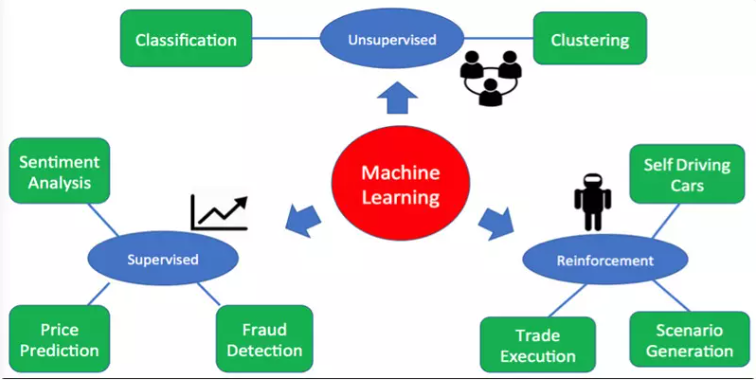


***Figure 1.2: Machine learning workflow.***

1. ***Gathering data/Data collection***: This is the most time consuming job and can take up 70-80% of the total process time. However, this work is very important and affects your machine learning model. For the model to have good quality and performance requires a good data set.
2. ***Data preprocessing***: In this sequence of processes, data preprocessing removes unnecessary attributes and normalizes the data. This work is proportional to the data collection step.
3. ***Training model & Evaluating model***: It takes less time to build, train and evaluate the model. However, the work is rotated and performed continuously, after evaluating the model, we continue to find the optimal way > train > evaluate until the accuracy increases. Although the work takes up little time, this is considered an important stage for choosing the right model.
4. ***Improve***: After you have evaluated and selected a suitable model, some models have not reached the required accuracy and need to be retrained, we will repeat from step 3, until when the expected accuracy is achieved. The time spent in 3 steps of training, evaluation and improvement accounts for about 30% of the total process.

## Classification of Machine Learning:

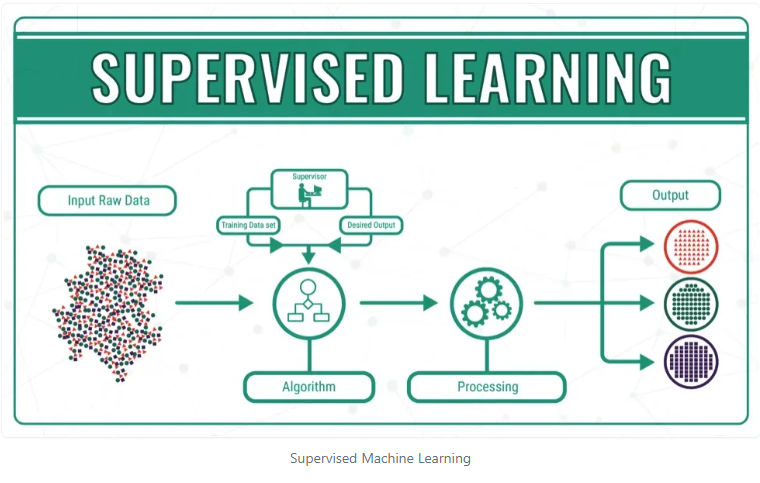
Machine learning is classified into three main categories as follows:



**Figure 1.3: Classification of Machine Learning.**

### **Supervised Machine Learning:**

Supervised Machine Learning, also known as supervised machine learning, is defined by using labeled data sets to create algorithms that categorize data or predict outcomes accurately.

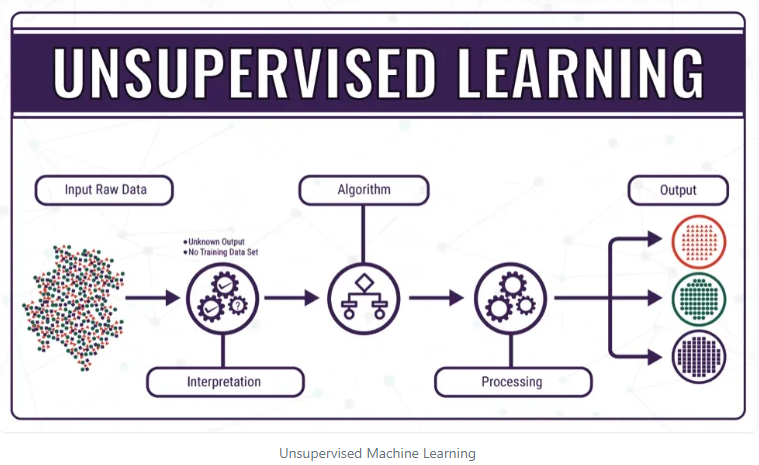


**Figure 1.3.1: Supervised Machine learning.**

Supervised machine learning helps organizations solve many real-world problems at scale, such as sorting spam in a separate folder from the inbox. Some of the methods used in supervised machine learning include neural networks, Navie Bayes, linear regression, logistic regression, Random Forest, SVM algorithms, etc.

### **Unsupervised Machine Learning:**

Unsupervised Machine Learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled data sets. These algorithms uncover hidden patterns or groups of data without human intervention.



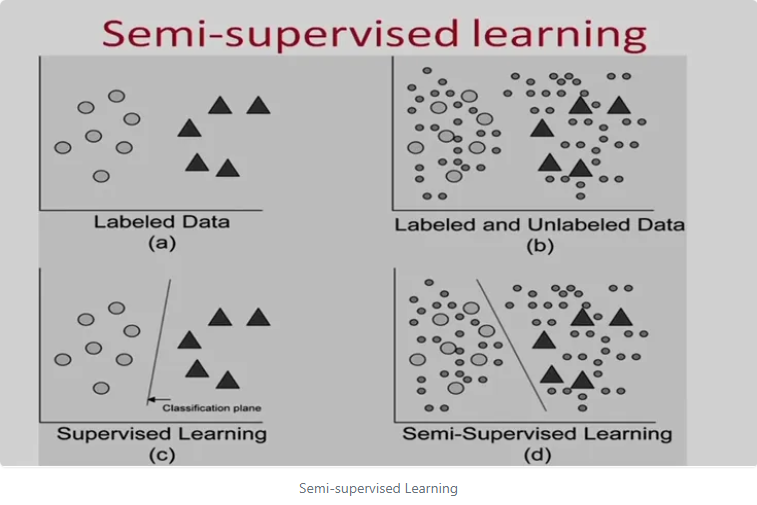
**Figure 1.3.2: Unsupervised Machine learning.**

Its ability to uncover similarities and differences in information makes it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, image and pattern recognition.

Unsupervised Machine Learning is also used to reduce the number of features in a model through dimensionality reduction. Principal component analysis (PCA) and single value analysis (SVD) are two common approaches for this. Other algorithms used in unsupervised machine learning include neural networks, K-means clustering, probabilistic clustering methods, etc.

### **Semi-supervised learning:**

Semi-supervised Learning provides a means of bridging between supervised and unsupervised Machine Learning. During training, it uses a smaller labeled dataset to guide the classification and feature extraction from a larger dataset that is not labeled. Semi-supervised Learning can solve the problem of not having enough labeled data to train a supervised learning algorithm.



**Figure 1.3.3: Semi-supervised learning.**

**Some concepts in Machine Learning**:

+ ***Dataset***: The unprocessed primitive data set that you collected in the data collection step. A dataset can contain many data points.

+ ***Data point***: As an independent unit of information in your data set, such as if you have data sets including house prices, construction date, area, etc., a data point will include that information. . A set of data points is called a dataset.

+ ***Training data and test data***: The dataset is usually divided into these two sets, the training data is responsible for training in the Machine Learning Workflow model, testing data for predicting results and evaluating the model.

+ ***Features vector***: As a feature vector, each vector is responsible for representing a data point in the dataset. Each vector has n dimensions representing the features of the data point, each feature is one-dimensional and must be numeric. Models can only be trained from these feature vectors, so the dataset needs to be converted to a set of feature vectors.

+ ***Model:*** These are models that are often used to train on training data based on the model's algorithm. And then the model will predict or make decisions based on what has been learned.

## Applications of Machine Learning:

+ ***Image Recognition***: is also one of the popular applications of Machine Learning. Machine Learning is also used to detect faces in photos of many people. There is a separate category for each person in the image database of many people.

+ ***Speech recognition***: Also known as automatic speech recognition (ASR - Automatic Speech Recognition), uses Natural Language Processing (NLP - natural language processing) to process human speech into a format write. Many mobile devices integrate voice recognition into the system to perform voice searches.

For example: Siri can access all the built-in apps on your Apple device like Mail, maps, messages, contacts, etc. by talking.

+ ***Customer Service***: Online Chatbots are replacing human agents in the process of communicating with customers. Chatbots that answer frequently asked questions (FAQs) around topics, like shipping or providing personalized advice, cross-selling or recommending sizes to users, change the way we think about things about customer engagement on websites and social media platforms.

Examples: Message bots on e-commerce sites with virtual agents, messaging apps, such as Slack and Facebook Messenger, and tasks typically performed by virtual and voice assistants.

+ ***Computer vision***: This AI technology allows computers and systems to derive meaningful information from digital images, videos, and other visual inputs. Based on those inputs action can be taken. This ability to provide recommendations distinguishes them from image recognition tasks. Powered by a complex neural network, computer vision has applications in social media tagging, radiography in healthcare, and self-driving cars in the automotive industry.

+ ***Recommendation Engine***: Using past consumer behavior data, AI algorithms can help uncover data trends that can be used to develop more effective cross-selling strategies. This is used to make relevant additional recommendations to customers during checkout for online retailers.

+ ***Automated Stock Trading***: Designed to optimize stock portfolios, AI-driven high-frequency trading platforms execute thousands or even millions of trades a day, without the need for human intervention. human intervention.

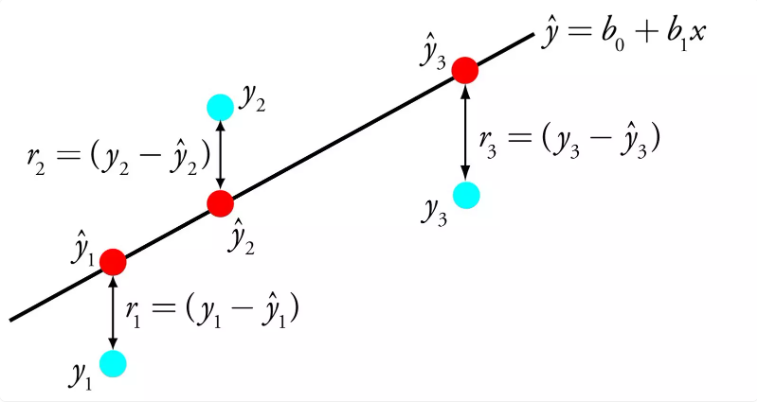
# 10 Algorithms in Machine Learning

## Linear Regression

Linear regression is one of the most famous algorithms in Machine Learning. This is a statistical method for data regression with the dependent variable having continuous values ​​while the independent variables can have either continuous values ​​or categorical values.

To understand how this algorithm works, imagine you would arrange random logs in ascending order of their weight. However, you cannot weigh every single log. You have to guess its weight just by looking at the height and circumference of the log (visual analysis) and sort them using a combination of these visible parameters. This is linear regression in Machine Learning.

Linear regression was invented more than 200 years ago and is widely studied. This is a good algorithm, fast and easy to use.

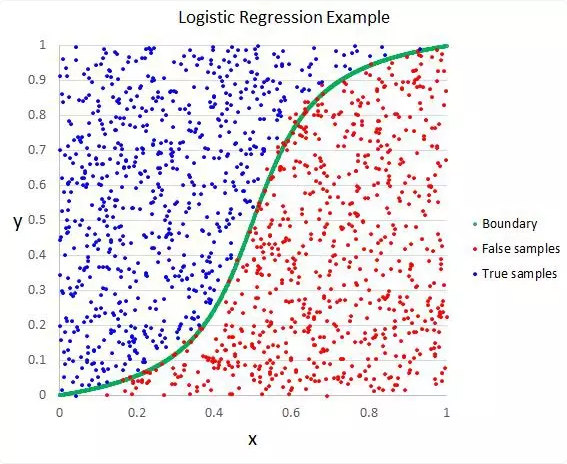


**Figure 2.1: Linear Regression Algorithms.**

## Logistic Regression:

Logistic regression is used to estimate discrete values (usually binary values like 0/1) from a set of independent variables. Logistic regression helps to predict the probability of an event by fitting the data to a logit function.

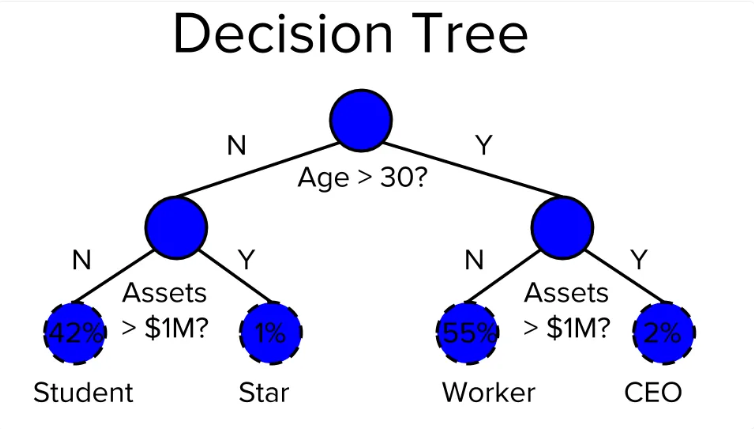
Similar to linear regression, logistic regression will perform better when removing attributes that are not related to the output variable or similar attributes. This is a model that can be learned quickly and effectively with binary classification problems.



**Figure 2.2: Logistic Regresssion** **Algorithms.**

## Decision Tree

The Decision Tree algorithm in Machine Learning is one of the most popular algorithms used today. This is a supervised learning algorithm used to classify problems. Decisiom Tree performs well when classifying for both categorical and continuous dependent variables. In this algorithm it is possible to divide the population into two or more homogeneous sets based on the most important independent variables or attributes.

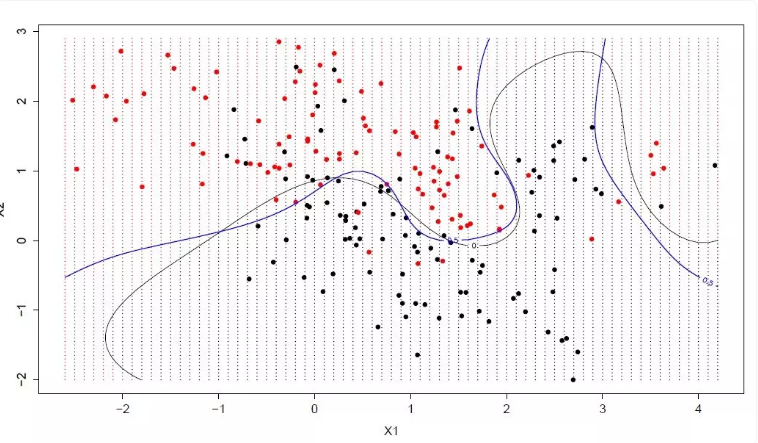


**Figure 2.3: Decision Tree Regresssion** **Algorithms.**

## Support Vector Machine (SVM Algorithms)

The SVM algorithm is a method of classification algorithm in which you graph raw data as points in N-dimensional space (where n is the number of objects you have). The value of each feature is then tied to a specific coordinate, making it easy to classify the data. Those lines are called classifiers that can be used to separate the data and plot them on the graph.

The SVM algorithm solves many major problems such as wide-ranging image classification, advertising display, and image gender detection.

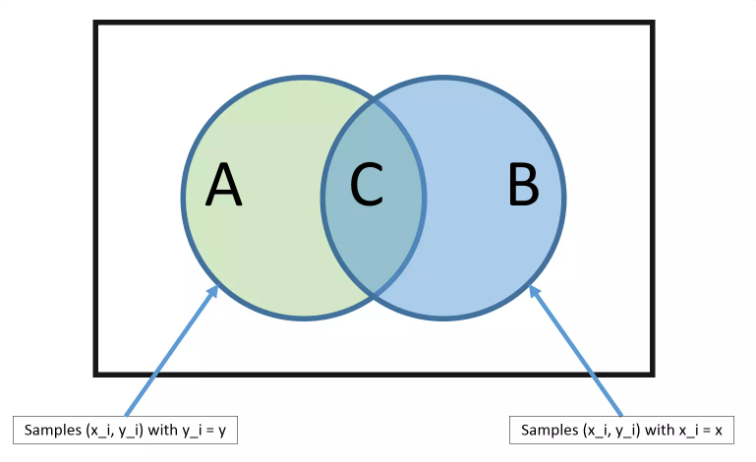


**Figure 2.4: SVM Algorithms.**

## Naive Bayes Algorithms

Naive Bayes is a simple algorithm but has an extremely accurate predictive model. The Naive Bayes model assumes that the presence of a particular feature in a class is not related to the presence of any other feature.

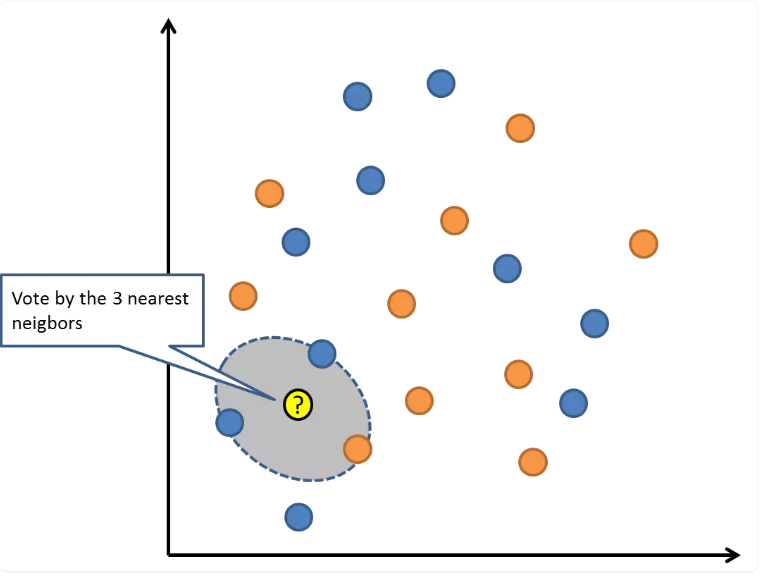
Even if these features are related, the Naive Bayes classifier will consider all these attributes independently when calculating the probability and give a specific outcome. Naive Bayes models are easy to build and useful for large datasets with complex problems.



**Figure 2.5: Naive Bayes Algorithms.**

## K-Nearest Neighbors (KNN) Algorithms:

This algorithm can be applied to both classification and regression problems. Obviously, in Data Science, KNN is more widely used to solve classification problems. This is a simple algorithm that stores all available cases and classifies any new cases by taking the majority of votes of K neighbors. The instance is then assigned to the class with which it has the most in common. A distance function performs this measurement.



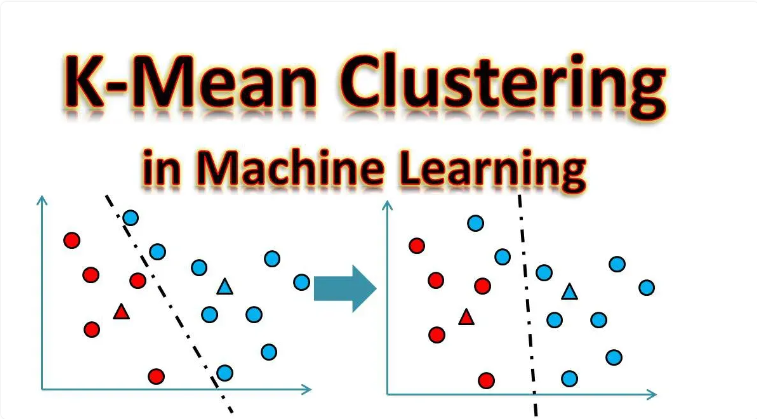
**Figure 2.6: KNN Algorithms.**

KNN can be understood easily by comparing it with real life. Example: If you want information about a person, talk to their friends and colleagues.

However, a few things need to be considered before choosing a KNN such as: Variables must be normalized, otherwise variables with a higher range may skew the algorithm; The data still needs to be preprocessed.

## K-Means:

K-Means is an unsupervised learning algorithm for solving clustering problems. Data sets are classified into a specific number of clusters (let's call that number K) in such a way that all the data points in one cluster are homogeneous and heterogeneous with the data in other clusters.



**Figure 2.7: K-Mean Algorithms.**

How K-means form clusters:

+ K-means algorithm chooses K number of points for each cluster, called centroid.

+ Each data point forms a cluster with nearest centers, it means: K cluster.

+ Create new hubs based on existing member clusters.

+ With these new centers, the closest distance for each data point is determined. This process is repeated until the centers remain unchanged.

## Random Forest Algorithms:

A collection of Decision Trees is called a Random Forest. To classify a new object based on its attributes, each tree will be classified, and “vote” for that class.

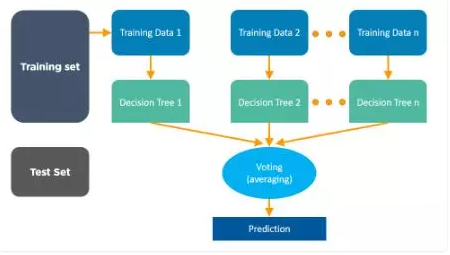
The Random Forest algorithm works in the following steps:

+ Select random samples from the given document set.

+ Set a Decision Tree for each sample and get the results of each Decision Tree prediction.

+ Vote for each prediction outcome.

+ Select the most voted result as the result.



**Figure 2.8: Random Forest Algorithms.**

## Dimensionality Reduction

In today's world, vast amounts of data are being stored and analyzed by companies, government agencies, and research institutions. These raw data contain a lot of information, the challenge here is to identify important patterns and variables.

Simply describing, Dimensionality Reduction is the transformation of data from high-dimensional space into low-dimensional space for low-dimensional representation retaining some meaningful properties of the original data.

Dimension reduction, or dimensionality reduction algorithms such as Decision Tree, Factor Analysis, Missing Value Ratio, and Random Forest can help you find relevant details.

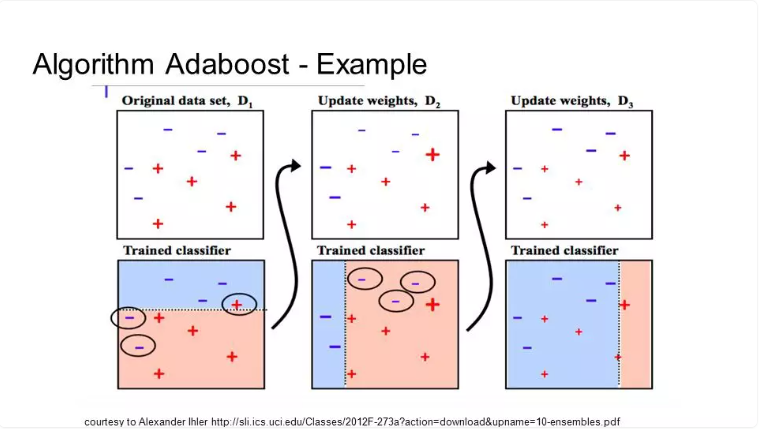


**Figure 2.9: Dimensionality Reduction Algorithms.**

## Gradient Boosting and AdaBoosting Algorithms

These are impulse algorithms used when large amounts of data have to be processed to make predictions with high accuracy. Boosting is a synthetic learning algorithm that combines the predictive power of several base estimators to improve robustness.

AdaBoost is a powerful learning algorithm that accelerates the generation of a strong classifier. This is the first successful boosting algorithm developed for binary classification.



**Figure 2.10: Boosting and AdaBoosting Algorithms.**

# Overview and the obligation of the topic

## INTRODUCTION:

### **Overview:**

**Content** :

- Learn about license plates and license plate recognition systems.

- Problem statement and solution orientation.

- Research on some image processing algorithms and character recognition applications in vehicle number plate recognition.

### **Thematic topic:**

From the above content, my thesis will include the following tasks:

- Learn about image processing and license plate recognition problems.

- Find out information about license plates and classification of license plates of Vietnam.

- Learn the main stages of the license plate recognition problem including 3 main stages:

+ Location detection and license plate separation.

+ Character segment in license plate.

+ Character recognition.

- Test settings.

## OVERVIEW OF THE PROBLEM OF IDENTIFICATION OF VEHICLES:

### **Concept of license plate:**

In Vietnam, a motor vehicle control plate (also known as the abbreviation of a control plate or license plate) is a plate attached to each motor vehicle, issued by the police (for military vehicles issued by the Vietnamese Ministry of National Defense) grant when Vietnamese citizens buying a new car, motorbike or transferring it. The license plate is made of iron and aluminum alloy, rectangular or slightly square in shape, with numbers and letters printed on it (civil license plates do not use the letters I, J, O, Q, W. The letter R indicates used for trailers, semi-trailers) indicates: Management area and locality, specific numbers when looking up on the computer also indicates the identity of the owner or the unit who bought it, the time bought it for security work, especially on a license plate, there is an embossed image of the National Emblem of Vietnam.

***Size standards***: In each country, there are usually certain size standards, but in Vietnam, the size ratio between number plates is almost the same. The license plate has 2 types, the dimensions are as follows: The long license plate has a height of 110 mm, a length of 470 mm; the short number plate type has a height of 200 mm, a length of 280 mm, so the Vietnamese government will limit the height / width ratio to 3.5 ≤ high/wide ≤ 6.5 (single-row sign) and 0.8 ≤ high/wide ≤ 1.5 (double-row sign).

The number of characters in the license plate is in the range [7,9]. Height of letters and numbers: 80mm, width of letters and numbers: 40mm.

From the above characteristics, we can set up parameters and controls to select the appropriate objects that we need.

### **Image processing and Open CV:**

Image processing is a sub-discipline of digital signal processing where the processed signal is an image. This is a new branch of science that has developed very rapidly in recent years. Image processing includes four main areas: image enhancement processing, image recognition, image compression and image query. The development of image processing brings many benefits to people's lives. Nowadays, image processing has been widely applied in life such as: photoshop, image compression, video compression, license plate recognition, face recognition, handwriting recognition, astronomical image processing, medical images ,....

OpenCV (Open Computer Vision) is a leading open source library for computer vision processing, machine learning, image processing. OpenCV is written in C/C++, so it has very fast computation speed, can be used with real-time related applications. Opencv has interfaces for C / C ++, Python Java, so it supports Windows, Linux, MacOs and Android, iOS OpenCV has a community of more than 47 thousand users and the number of downloads exceeds 6 million times. Opencv has many applications such as:

• Image recognition

• Image processing

• Photo/video recovery

• Virtual reality

• Other Apps

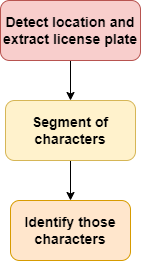
### **Problem-solution:**

Currently in the world there are many different approaches to license plate recognition, but within the scope of this project I will solve the problem in 3 main steps:

1. Detect location and extract license plate from a pre-existing image from camera input.

2. Segment of characters in license plate.

3. Identify those characters and return them to ASCII code.

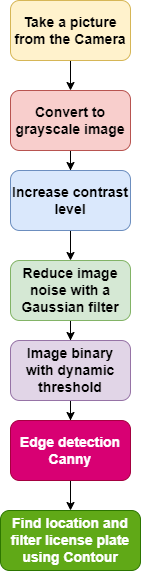


**Figure 3.2.3 - Main steps in license plate recognition.**

# Location Detection and Separation of Vehicle Number Licene Plates.

## Problem-solution:

The diagram below summarizes the steps to identify and separate license plates:

****

**Figure 4.1 – Identify and seperate license plates.**

Firstly, we will cut each image frame out from the input clip to process and separate the number plate. In the scope of this project, the main idea is to recognize the license plate from the sudden change in light intensity between the license plate and the surrounding environment, so we will remove the RGB color data by converting gray image. Next, we increase the contrast with two morphological operations Top Hat and Black Hat to highlight more number plates in the background, supporting binary processing later. Then, we reduce the noise with a Gaussian filter to remove the noise details that can affect the recognition process, and at the same time increase the processing speed.

Taking threshold will help us to separate license plate information and background information, here we choose to take dynamic threshold (Adaptive Threshold). Next, we use Canny edge detection algorithm to extract the edge details of the license plate. During computer processing, the number plate may be confused with noisy details, final filtering by high/wide ratios or the area of the license plate will help identify the correct number plate. Finally, we will determine the position of the number plate in the image by drawing a Contour around it.

## Convert to grayscale image:

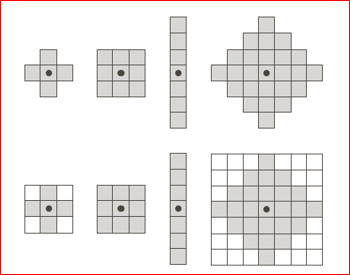
A Gray Scale image is simply an image where the colors are shades of gray with 256 gray levels varying from black to white, ranging from 0 to 255, that is, 8 bits or 1 byte to represent each of these pixels. The reason it is important to distinguish between grayscale images and other images is that grayscale images provide less information per pixel. With a normal image, each pixel is usually provided with 3 fields of information while with a gray image there is only 1 information field, reducing the amount of information helps to increase processing speed, simplifies the algorithm but still ensures the correct information and required task.

In this topic, I will convert gray images from HSV color system instead of RGB because with HSV color space we have three main values: Color area (Hue), saturation (Saturation), light intensity (Value). For that reason, the HSV color space is better adapted to changes in ambient light. When converting, the gray image we need is a matrix of intensity values extracted from the HSV color system.

## Increase Contrast level:

### **Morphological math:**

Mathematical morphology is a theory and technique for analyzing and processing geometric structures, the output image is determined mainly based on the structural elements/kernel.



**Figure 4.3.1 – 1: Structuring element example.**

Mathematical morphology has been developed for binary images, and then extended to grayscale images, etc. This is one of the techniques applied in the preprocessing stage. Two commonly used operations are dilation and contraction. From these two basic operations, people develop a number of operations such as closing (Closing) and opening (Opening) and Top Hat, Black Hat operations.

1. **Erosion:**

The erosion operation has applications in reducing the size of objects, separating close objects, and slicing and finding bones of objects.

The contraction of image A by structuring element B produces image G. Each location of structure B scanned will select the smallest value and return the corresponding landmark on image G.



**Original image** **Image after using erosion operation.**

**Figure 4.3.1 – 2: Erosion Operation.**

1. **Dilation**

This operation has the effect of making the original object in the image increase in size (expand). The application of dilation is to make the object in the image increase in size, the small holes in the image are filled, connecting the image border for small discrete segments.

The stretching of image A by structuring element B produces image G. Each position of structure B scanned will select the maximum value and return the corresponding landmark on image G.



**Original image** **Image after using dilation operation.**

**Figure 4.3.1 – 3: Dilation Operation.**

1. **Opening:**

Is to perform the erosion operation first and then perform the dilation operation. Open operation is applied in removing protrusions and smoothing the contours of objects in the image.



**Original image** **Image after using opening operation.**

**Figure 4.3.1 – 4: Opening Operation.**

1. **Closing:**

Do the dilation operation first, then perform the erosion operation. The closing operation used in the application smooths the contours of objects, fills in boundary gaps, and removes small holes.



**Original image** **Image after using closing operation.**

**Figure 4.3.1 – 5: Closing Operation.**

1. **Top Hat Operation:**

Top Hat operation is the result of image subtraction of the original image with the image after performing the opening operation, used to highlight white details in the dark background.

**Original image** **Image after using Top Hat operation.**

**Figure 4.3.1 – 6: Top Hat Operation.**

1. **Black Hat Operation:**

Black Hat operation is the result of image subtraction of the image after performing the closing operation with the original image. Use to highlight dark details in a white background.

**Original image** **Image after using Black Hat operation.**

**Figure 4.3.1 – 7: Black Hat Operation.**

## Increase Contrast level:

To increase the contrast of the number plate, I mainly use two spells Top Hat and Black Hat. The general idea is that the output image will be the original image plus the image using the Top Hat operation and subtracting the image using the Black Hat operation. Bright details will be brighter and dark details will be darker, thereby increasing the contrast of the number plate.

**Original image** **Image after using Increase Contrast level.**

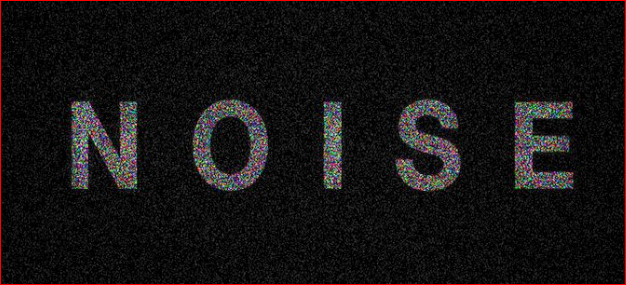
**Figure 4.4: Image after using Increase Contrast level.**

## Reduce image noise with a Gaussian filter:

### **Noise:**

Noise is basically understood as the form of small particles distributed on the image. Noise can distort details in an image, resulting in low image quality.

In fact, there are many types of noise, but it is usually divided into three types: additive noise, multiplicative noise and impulse noise. The nature of noise usually corresponds to high frequencies, and the theoretical basis of the filter is to allow only signals of certain frequencies to pass, so it is common to use low or medium pass filters.

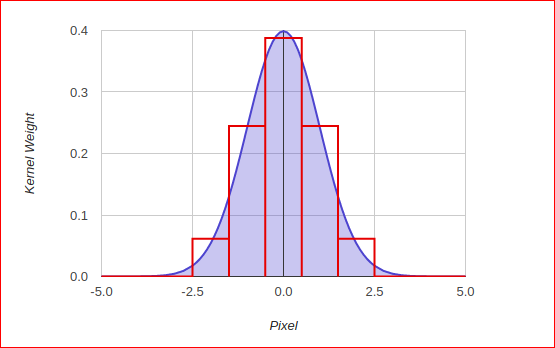


**Figure 4.5.1: Noise.**

### **Gauss filter:**

The Gaussian filter is said to be the most useful filter, implemented by convolutionalizing the input image with a Gaussian filter matrix and then adding them together to form the output image.

The general idea is that the value of each pixel will depend more on the nearby pixels than on the distant pixels. The weight of the dependency is taken as a Gaussian function (also used in the normal distribution).



**Figure 4.5.2: Gauss filter matrix.**

Assume the image is one-dimensional. The pixel in the center will have the greatest weight. Pixels farther from the center will have a decreasing weight as their distance from the center increases. Thus, the closer the point is to the center, the more it will contribute to the central point value.

**Original image** **Image after blurring, reduce noise.**

**Figure 4.5.3: The result of using Gauss filter.**

## Image binary with dynamic threshold:

### **Binary image:**

An image where the value of pixels is represented by only two values, 0 (Black) and 255 (White).

### **Binary process:**

It is the process of converting a grayscale image into a binary image.

- Call the luminous intensity value at a pixel I(x,y) .

- INP(x,y) is the intensity of the pixel on the binary image.

- (With 0 < x < image.width) and (0 < y < image.height).

To convert grayscale image to binary image. We compare the pixel's luminance value with a binary threshold T.

- If I(x,y) > T, then INP(x, y) = 0.

- If I(x,y) > T, then INP(x, y) = 255.

### **Binary with dynamic threshold:**

It is very difficult to binaryize an image with a global threshold as it usually is when you have to manually calculate and choose the appropriate threshold level for each different image. Dynamic thresholding image binarization will help calculate the threshold to suit each image, the second advantage is that it is very suitable when the image has areas that are too bright or too dark, leading to the loss of images in that area if use global thresholds.

About the main idea will follow the following 3 steps:

1. Divide the image into many different areas, windows (Region).

2. Use an algorithm to find a matching T value for each window.

3. Apply the binary method to each area and window with the appropriate threshold T.

There are many methods to find T, here I use an algorithm that the OpenCV library supports, which is ADAPTIVE\_THRESH\_GAUSSIAN\_C ie averaging the values around the dynamic threshold point under consideration T(x,y) with a Gaussian distribution, then subtract the constant C.



**Figure 4.6.3: Binary with dynamic threshold image.**

## Canny Edge Detection:

In images, there are often components such as: smooth areas, corners/edges, and noise. Edges in images have important features, often belonging to objects in the image. Therefore, for edge detection in the image, there are many different algorithms such as Sobel operator, Prewitt operator, Zero crossing .... But here I choose Canny algorithm because this method is superior to other methods due to its advantages. less affected by noise and has the ability to detect weak edges. This method follows 4 main steps:

1. Noise reduction.
2. Gradient calculation.
3. Non-maximum suppression.
4. Double threshold.
5. **Noise reduction:**

Blur image, reduce noise using Gauss filter size 5x5. The 5x5 size usually works well for the Canny algorithm.

1. **Gradient calculation:**

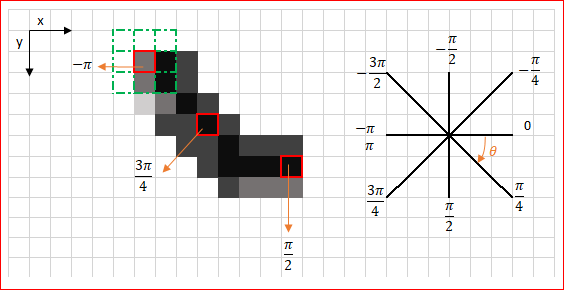
We use 2 filters Sobel X and Sobel Y (3x3) to calculate the derivative Gx and Gy.



Find the gradient and direction rounded in 4 directions: horizontal (0 degrees), right diagonal (45 degrees), vertical (90 degrees) and left diagonal (135 degrees).

1. **Non-maximum suppression:**

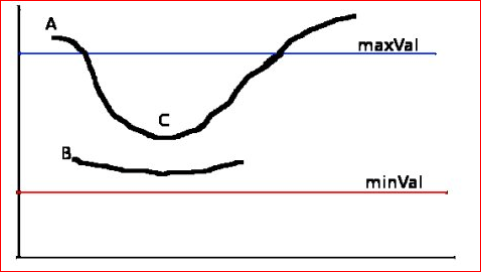
In this step, use a 3x3 filter that runs through the pixels on the gradient image in turn. During the filtering process, consider whether the gradient magnitude of the central pixel is maximum compared to the gradients in the surrounding pixels. If it is the maximum, we will note that we will keep that pixel. And if the pixel there is not a neighboring maximum, we will set its gradient magnitude to zero. We only compare the central pixel with 2 neighboring pixels in the gradient direction. For example, if the gradient direction is 0 degrees, we will compare the center pixel with its left and right adjacent pixels. In other case, if the gradient direction is 45 degrees, we will compare it with 2 neighboring pixels, the upper right corner and the lower left corner of the center pixel.



**Figure 4.7-1: Non-maximum suppression.**

1. **Double threshold:**

Threshold filtering: we will consider positive pixels on the resulting binary mask of the previous step. If the gradient value exceeds the max\_val threshold, the pixel is definitely an edge. Pixels with gradient magnitude less than min\_val threshold will be discarded. Pixels that fall within the upper two thresholds will be considered to be adjacent to those that are said to be "definitely edge". If adjacent, we keep, if not adjacent to any edge pixels, we remove. After this step, we can apply additional post-processing to remove noise (ie, discrete or short edge pixels) if desired.



**Figure 4.7-2: Double threshold.**

1. **Result:**

After using canny edge detection, although we have extracted the edge details of the number plate, but there are still too many redundant details in the image, from here we will draw contour, apply the characteristics of the license plate number to filter to get the correct number plate.



**Figure 4.7-3: Image after detecting Canny edge.**

* 1. **Filter license plate using Contour:**
     1. **Several methods of finding contour:**

Contour can be understood as a set of points forming a closed curve around an object. Often used to determine the position and characteristics of objects. There are 4 most common Contour Tracing algorithms. Two of them named: Square Tracing algorithm and Moore - Neighbor Tracing are easy to implement and are frequently used to detect the contour of a pattern. With the OpenCV library, Suzuki's Contour tracing algorithm is applied. Below I will describe in more detail the three methods above:

1. **Square Tracing Algorithm:**

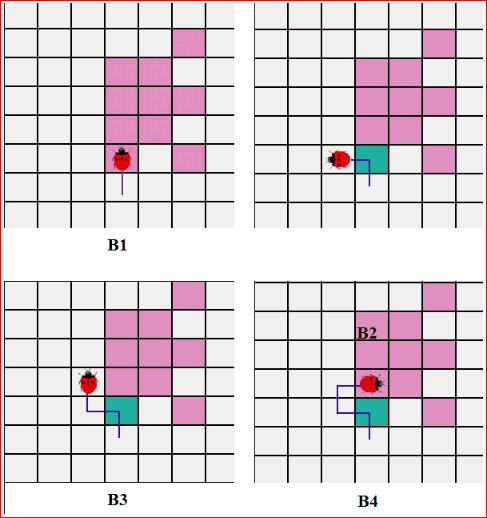
Browse from the bottom leftmost pixel, go up until we meet a pixel with a value of 255 (this pixel will be called the start pixel), then start moving according to the following rule:

- If we encounter a Pixel with a value of 255, turn left.

- If we encounter a Pixel with a value of 0, turn right.

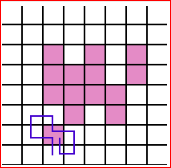
- Move until we return to the start pixel, then stop.

The following figure shows how the algorithm works:

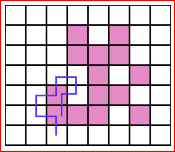


**Figure 4.8.1 - 1: Square Tracing Algorithm.**

The algorithm will finish correctly when moving into the start pixel for the 2nd time after going through n other pixels and in the right direction entering the start pixel for the first time. And it is wrong to move into the start pixel which is not in the original direction. So this algorithm only runs correctly on the object 4 - connected.



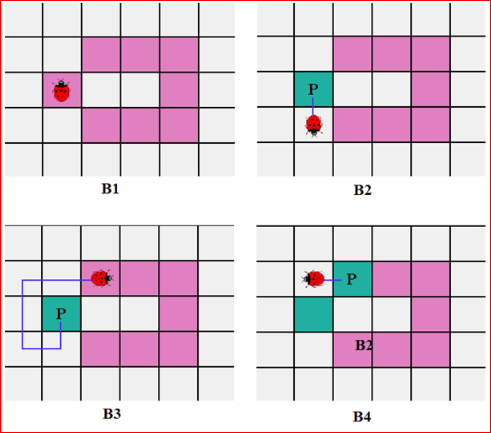
**Figure 4.8.1 - 2: Square Tracing Algorithm runs correctly.**



**Figure 4.8.1 - 3: Square Tracing Algorithm runs wrongly.**

1. **Moore – Neighbor Algorithm:**

This algorithm is slightly different from the Square Tracking algorithm, specifically: when we encounter the first pixel with a value of 255 (pixel start), we will return to the previous pixel, then cycle through the pixels of 8- connected clockwise until another pixel has a value of 255. And the termination condition is the same as Square Tracking algorithm.



**Figure 4.8.1 - 4: Moore – Neighbor Algorithm.**

1. **Suzuki’s Tracing:**

This is the algorithm used by the OpenCV library, in addition to the ability to determine the boundary of the object like the two methods above. Suzuki's Tracing method is also capable of distinguishing the outer boundary (Outer) and the inner boundary (Hole) of the object.

The function in OpenCV is represented as follows:

**findContours**(InputOutputArray image, OutputArrayOfArrays contours, OutputArray hierarchy, int mode, int method, Point offset=Point())

**Parameters:**

**image** : The image to find the edge is a binary image..  
**contours** : stores the found contours, each boundary is stored as a vector of points.  
**hierarchy** :  contains information about the image such as number of contours, rating of borders by size, inside out, ..

**mode** :  
CV\_RETR\_EXTERNAL : when using this flag it only retrieves the outer boundary, but the inner boundary of the object is removed.

CV\_RETR\_LIST : When using this flag it retrieves all found contours.  
CV\_RETR\_CCOMP : when using this flag it takes all the boundaries and divides it into 2 levels, the boundaries outside the object, and the borders inside the object.  
CV\_RETR\_TREE : when using this flag it takes all boundaries and creates a full hierarchy of nested lines.

**method** :  
CV\_CHAIN\_APPROX\_NONE : using this flag will store all points of the contour.  
CV\_CHAIN\_APPROX\_SIMPLE : For example, a rectangle will be encoded using the coordinates of 4 vertices.  
CV\_CHAIN\_APPROX\_TC89\_L1 or CV\_CHAIN\_APPROX\_TC89\_KCOS : Apply the Tech-Chin approximation algorithm.



**Figure 4.8.1 - 5: Draw Contour with OpenCV.**

In the image, the pink lines are the contour lines surrounding the object, but because there are too many lines around the objects other than the number plate, we will apply the specific features of the high/wide scale, the area in the fixed frame image to filter out the correct number plate to look for.

* 1. **Filter number plate:**

First, we approximate the contour to a polygon and only take those polygons that have only 4 sides. That is, when approximating the contour ’s memory only remembers the position of the vertices of that polygon into an array. The number of sides of the polygon will be equal to the number of vertices and the length of the array. 

**Figure 4.9 - 1: Contour does not reach approximately the polygon.**



**Figure 4.9 - 2: Contour has already reached approximately the polygon.**

Next we calculate the height/width ratio and the area of the appropriate number plate, then we save all the number plates shown in the figure as the coordinates of the vertices

From here, we cut the number plate image from the known location coordinates to serve the next purpose "Separation of characters in the number plate". Note here that we cut from the binary image so that the computer can process it faster and take less time.

# Character segmentation, character recognition and the KNN algorithm

## Character segmentation:

In essence, the character recognition process is the process of converting from an image which is a matrix of pixel values to another form of information such as ASCII code in this topic to be able to communicate with users. To better understand recognition, we need to go back to the science of artificial intelligence (Artificial Intelligent) or AI as mentioned in chapter 1 and chapter 2.

### **K - Nearest Neighbor (KNN) Algorithm:**

KNN is one of the simplest supervised learning algorithms in Machine Learning that can be used for both classification and regression problems. About the idea is to assign the result with the training data that most closely resembles the sample. For example, when we go fishing, we don't know if the fish caught up is perch or carp, but when we compare the features of eyes, gills, fins, ... from the perch, carp which have been seen, then we can made a decide which group of fish we catch belongs to.

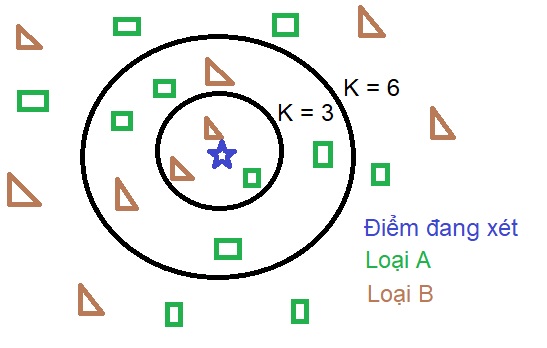
KNN operates according to a process of 4 main steps:

1. Determine the parameter K (number of nearest neighbors).

2. Calculate the distance from the point in question to all points in the given data set.

3. Sort those distances in ascending order.

4. Considering in the set of K points closest to the point under consideration, if the number of points of any type is higher, it is considered that the point under consideration belongs to that type.



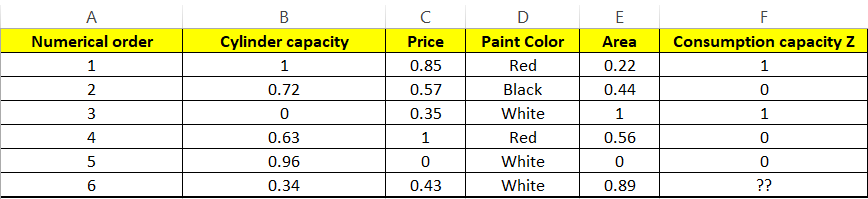
**Figure 5.1.1 - 1: Example of KNN Algorithm.**

The type of point being considered depends on the K coefficient or the distance weight... that the user sets to suit the problem under consideration. For example, in the above figure, if we consider K = 3, the point under consideration will be of type B, otherwise if K = 6, it is of type A. In addition, one can give higher weights to points closer or less when using K = 1 to ensure optimal output.

Usually, the calculation of the distance to the points will follow the Euclidean formula:

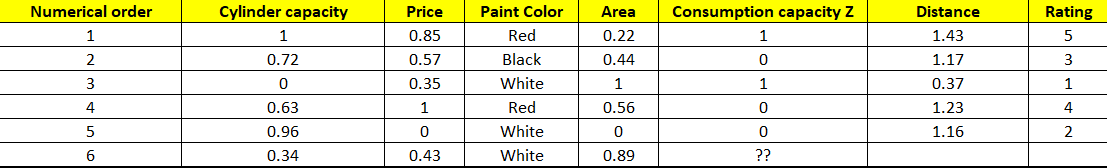
When performing comparisons, the square root sign can be omitted. In addition, if the distance between variables is too large, such as the x variable is approximately 1000000 times larger, we also need to normalize the data according to the formula:

To make it easier to understand, I would like to take an example of a company that wants to bring its new car line to the market and wants to know if the car will sell well or not. They synthesize data on cylinder capacity, price, paint color, vehicle area of the models that have been exported to the market. Set Z to be the instrument consumption capacity of the vehicle line. Z = 0 means the car is poor, Z = 1 means the car will sell well



**Figure 5.1.1 – 2 :** **Post-normalization data.**

Then we calculate the distance between the point considering the given data. Note in variables with qualitative values, if different, the distance will be 1, the same will be 0. For example, we calculate the interval between cars 6 and 1:



**Figure 5.1.1 – 3 :** **After calculating distance and rating.**

Look at the picture above if K=1, the result is Z=1, if K=3 then Z=0, if K=5 then Z=0.

However, this method has some advantages and disadvantages as follows:

- Advantages:

+ Easy to use and install.

+ Small computational complexity.

+ Predicting the outcome is very simple.

- Disadvantages:

+ With small K, when encountering noise, it is easy to give inaccurate results.

+ It takes a lot of time to save the training set and if the test increases, it will take a lot of time.

## Problem-solution:

At this final stage, the following steps are performed:

1. Create data set for training.

2. KNN model training.

3. Put the image from the step “Character segmentation” into the generated KNN model to give the result.

4. Print out the number plate result.

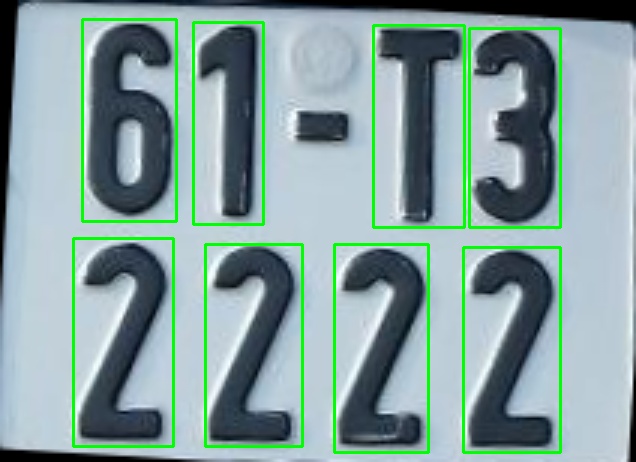
Steps 1 and 2 will create a separate KNN model from the main code. So that when we need to recognize characters, we do not need to redo the steps from the beginning. First, I create a data set (a set of images of numbers and characters) to train from paint software. In Paint software, we write numbers and characters (except O, I, J) with the font "Vietnamese license plate", which can be rotated with angles of -5°,5 respectively. °,-10°,10°. The result has the following form:



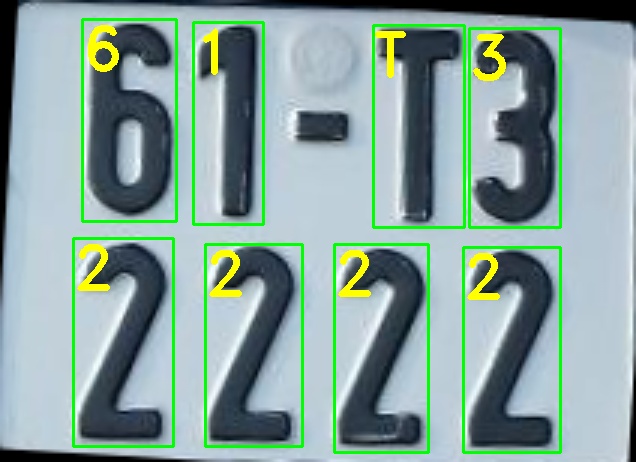
**Figure 5.2 – 1 :** **Training dataset.**

Next we take the threshold, draw the contour and cut each character. Because each character has a different size, the processing is complicated, so it is necessary to normalize the image with a height: width of 30:20 pixels. Instead of each character entering the model for the machine to recognize, these characters will be labeled by the keys on the computer. After labeling all the characters, we will save two .txt files, classifications.txt and flattened\_images.txt. The file classifications.txt is responsible for saving the ASCII codes of those characters and the flattened\_images.txt file will store the values of pixels in the character image (20x30 pixel image has a total of 600 pixels with values of 0 or 255 ).

Steps 3 and 4. We take the image under consideration and calculate the distance to all points in the sample, the result will be the ASCII code representing that image. Finally, we print out the license plate number. However, in Vietnam, there are two types of number plates: single-row and double-row plates. About the general idea to distinguish these two rows, we rely on the position of the character image, if the position is low 1/3 of the height of the number plate, the character will be placed in row one. Otherwise, it will be ranked second.



**Figure 5.2 – 2 :** **License Plate before recognition.**



**Figure 5.2 – 3 :** **License Plate after recognition.**

****

**Figure 5.2 – 3 :** **License Plate are printed out with the comparison with the origin.**

## Methods to increase dataset diversity:

**Various sizes of number plates:**

Diversify the size in 2 ways:

• Method 1: Reduce the size of the license plate by adding a random size border to the original image, then resize the image to the original image size.

• Method 2: Crop the image containing the number plate with a random size, then resize the image by size of the original photo.

**Code for Method 1:**

def add\_boder(image\_path, output\_path, low, high):

"""

low: lowest border size (pixels)

height: highest border size (pixels)

"""

# random boundary sizes in the range (low, high)

top = random.randint(low, high)

bottom = random.randint(low, high)

left = random.randint(low, high)

right = random.randint(low, high)

image = cv2.imread(image\_path)

original\_width, original\_height = image.shape[1], image.shape[0]

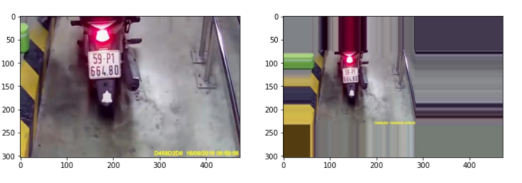
# use opencv's function to add border

image = cv2.copyMakeBorder(image, top, bottom, left, right, cv2.BORDER\_REPLICATE)

# then resize the image to its original size

image = cv2.resize(image, (original\_width, original\_height))

cv2.imwrite(output\_path, image)



**Figure 5.3 – 1 :** **Image obtained (right) after running the above function.**

**Code for Method 2:**

def random\_crop(image\_path, out\_path):

image = cv2.imread(image\_path)

original\_width, original\_height = image.shape[1], image.shape[0]

x\_center,y\_center = original\_height//2, original\_width//2

x\_left = random.randint(0, x\_center//2)

x\_right = random.randint(original\_width-x\_center//2, original\_width)

y\_top = random.randint(0, y\_center//2)

y\_bottom = random.randint(original\_height-y\_center//2, original\_width)

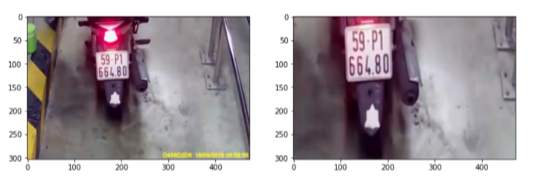
# crop to image area with random size

cropped\_image = image[y\_top:y\_bottom, x\_left:x\_right]

# resize image to original image size

cropped\_image = cv2.resize(cropped\_image, (original\_width, original\_height))

cv2.imwrite(out\_path, cropped\_image)



**Figure 5.3 – 2 :** **Image obtained (right) after running the above function.**

**Change the brightness of the photo:**

def change\_brightness(image\_path, output\_path, value):

"""

value: change the brightness

"""

img=cv2.imread(image\_path)

hsv = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV)

h, s, v = cv2.split(hsv)

v = cv2.add(v, value)

v[v > 255] = 255

v[v < 0] = 0

final\_hsv = cv2.merge((h, s, v))

img = cv2.cvtColor(final\_hsv, cv2.COLOR\_HSV2BGR)

cv2.imwrite(output\_path, img)



**Figure 5.3 – 3 :** **Change the brightness (right).**

**Rotate photo:**

import imutils

def rotate\_image(image\_path, range\_angle, output\_path):

"""

range\_angle: Rotation angle range

"""

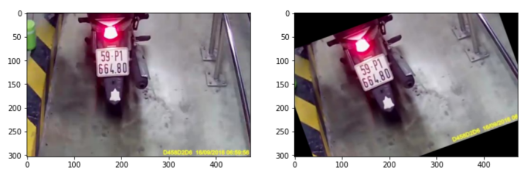
image = cv2.imread(image\_path)

# Randomly choose the rotation angle

angle = random.randint(-range\_angle, range\_angle)

img\_rot = imutils.rotate(image, angle)

cv2.imwrite(output\_path, img\_rot)



**Figure 5.3 – 4 :** **Image is rotated (right).**

**5.4 Labelling dataset:**

****

**Figure 5.4 : Identify the area containing the number plate.**

LabelImg supports labeling on both PASCAL VOC and YOLO formats with the file extension annotations are .xml and .txt respectively.

Each line in an annotation file includes: <object-class> <x> <y> <width> <height>.

Where: <x> <y> <width> <height> are the coordinates of the center and size of the object, respectively. These values have been renormalized, so the value is always in the range [0,1]. object-class is the index that marks the classes.

Note: With the problem with many labels, many people assign the same label, it is necessary to agree with each other before about label order. The reason is that in the annotation file, only the index (0,1,3,4,...) of the label is saved, but not the label name.

After assigning labels, put the annotation file and the corresponding image in the same folder.

# The result of performance

## How to measure and test:

Captured by the camera from the Samsung J7 Prime phone has the following parameters:

- Clip size: FHD 1920x1080

- Camera: 9.6 MP

- 8 images/s

Then it will be processed by the program to cut each image frame to identify and print the number plate. The processing program is written on the Visual Studio 2022 application interface in Python (3.7), including OpenCV, Numpy, Math.

I create two tests specifically for single-row and double-row number plates with all angles of tilt left, right, top and bottom, far and near, and environments with different backgrounds and lighting. For each test clip with length X image frames, each image frame containing Y number plates, we will get:

Total number plate = . ( with Y = 3 for 2-row sign, Y = 1 for single-row sign).

Percentage of finding license plates = 100 x (%)

We will find the ratio of wrong signs n characters to the total number of signs. The wrong note here means that the character is misrecognized, not zoned, or the character is in the wrong position.

Recognition rate= 100 x (%)

## Result and Explanation:

|  |  |  |  |
| --- | --- | --- | --- |
| Type of license plate | Total number plate | Finding license number plate | Percentage of finding license number plate (%) |
| Single-row plate | 370 | 182 | 49,2% |
| 2-row plate | 2349 | 924 | 39,3% |

**Figure 6.2 – 1: Percentage of finding the license plate in the picture.**

In the single-row plate has a higher recognition rate because the sample set is small and each image frame has only 1 plate, leading to the number plate is usually located in the center, so it is easier to identify. In contrast, the 2-row plate has a large sample set, and the number of plates in 1 frame is more. Not to mention, the number plate here is counted as a parallelogram with at least 7 characters to be counted as number plates, resulting in many number plates that have been cut correctly but are still not counted. However, this approach helps to significantly eliminate the details and wrong number plates from the external environment.

Figure 6.2 -2: Characters are not found.

***Figure 6.2 -5: Original image***.

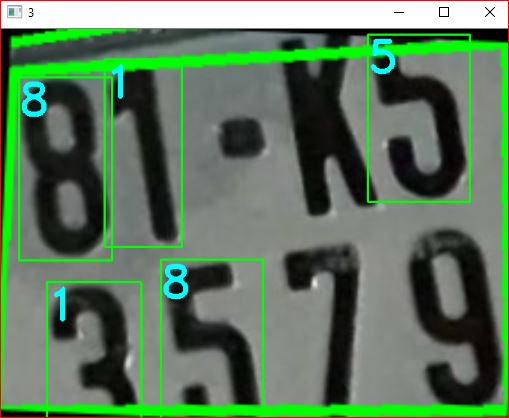


Figure 6.2 -3: Five Characters are found .

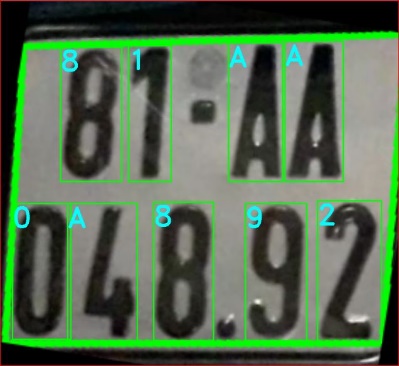


Figure 6.2 -4: Nine Characters are found .

In Figure 6.2 - 5, we see that the processing program finds 3 areas that are supposed to be number plates, but the number of characters found in Figures 6.2 - 2 and 6.2 - 3 is not enough, so it will be eliminated, resulting in only remaining in the only license plate image figure 6.2 - 4.

When we rotated from many angles, many positions, leading to the calculation of the area, the high / wide ratio of the number plate no longer met the set conditions, so it was eliminated. The number plate can be affected by external details, so the contour approximation does not produce a quadrilateral, leading to the loss of the number plate. This error especially occurs in motorbikes because motorbikes often have a background around the number plate that is strongly reflective materials, which greatly affects the process of determining the number plate area.



**Figure 6.2 -5 :** **Get 1.5 high/wide threshold.**



**Figure 6.2 -6 :** **Get 1.4 high/wide threshold.**



**Figure 6.2 -7 : Contour approximation error.**

In Figure 6.2 – 7: although the pink contour line surrounds the number plate, after approximating only 2 and 3 edges, to overcome the scale errors, contour approximation needs to be adjusted to the threshold level for optimization.

In the processing, the binary processing also plays an important role, in Figure 6.2 - 9 we see that the image is noisy and the number plate itself is dark and dusty, leading to the binary processing will be interrupted. and the contour is wrong, to overcome it is necessary to use morphological operations such as expansion, closure to make white lines in binary images.



**Figure 6.2 -8 : Broken binary image.**



**Figure 6.2 -9 : The contour line is broken.**

Below, we consider the ability to localize and recognize characters corresponding to the stages of "Character segmentation" and "Character recognition" set out at the beginning of the problem.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type of license plate | Finding license number plate | No wrong | 1 character wrong | 2 characters wrong | 3 characters wrong above |
| Single-row plate | 182 | 61 | 88 | 19 | 14 |
| Percentage (%) | | 33,5 | 48,4 | 10,4 | 7,7 |

**Figure 6.2 -10 : Percentage of wrong character recognition in the single-row plate.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type of license plate | Finding license number plate | No wrong | 1 character wrong | 2 characters wrong | 3 characters wrong above |
| Two-row plate | 924 | 286 | 273 | 175 | 190 |
| Percentage (%) | | 31 | 29,5 | 18,9 | 20,6 |

**Figure 6.2 -11 : Percentage of wrong character recognition in the two-row plate.**

In general, the KNN recognition model is also quite good, there are characters that are recognized correctly even though they are blurred or slanted. This is partly thanks to the program that has rotated the number plate to increase recognition, even if it is tilted, the character is only tilted from 3° to 7°. However, there is still a lot of confusion between characters such as numbers: 1 with the number 7, The letter G, the letter D, the number 6 with the number 0. The letter B with the number 8...



**Figure 6.2 -12 : Original photo identifying 3 license plates.**

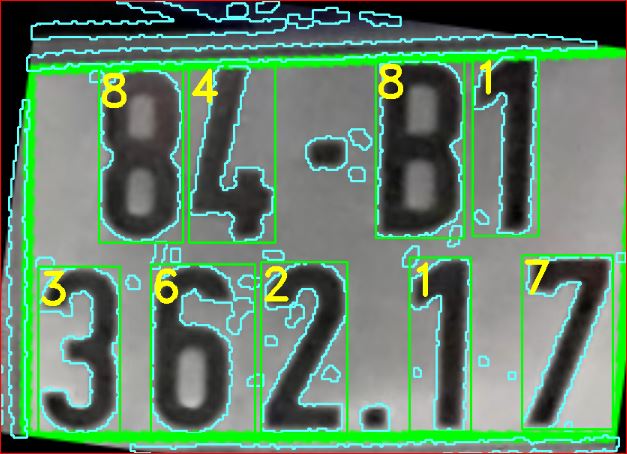
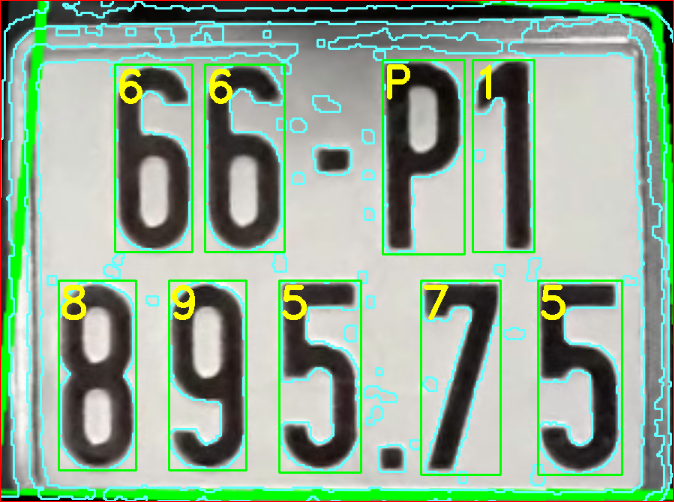
  

Figure 6.2 – 13: The first plate Figure 6.2 – 14: The second plate Figure 6.2 – 15: The third plate.

According to the original image 6.2 - 12, we can correctly identify all the number plates 2 and 3. The number plate 1 is wrong between the letter B and the number 8.



**Figure 6.2 -16 : Can't circle the character area.**

In the image 6.2 – 16 places where the zero is stuck with a snail, so it cannot be localized, However, in the letter F, although it still has the screw in the cutout image for identification, the KNN model still gives the correct answer.

# CONCLUSIONS AND DEVELOPMENT ORIENTATIONS

## Conclusion:

From the results obtained in the above chapter, I found that the method of recognizing license plates by image processing and the KNN algorithm has the following advantages and disadvantages:

* **Advantages**:

+ Easy to install and use.

+ It is quite light, so computers with weak configuration can also handle it smoothly compared to other algorithms such as CNN, SVM.

+ Suitable for students who want to learn the basics of image processing or artificial intelligence.

* **Weakness:**

+ The recognition ability of KNN is still low, when the data set is too large, the processing time will increase because it has to scan the entire train dataset.

+ Poor recognition with the reflection of license plates, movement, glare from the outside environment, plates with unclear numbers, with car license plates

Therefore, it is best to place the camera fixed, with the ambient light environment set in advance. The background needs to limit the darkening of the skin, but the flare details cause noise. This method still needs a lot of human supervision, but it cannot be fully automated.

## Developement Orientation:

It is necessary to change the KNN recognition algorithm to other more sophisticated and complex algorithms such as CNN, SVM or can use existing libraries in the world such as YOLO, YOLOv3 ....

Using a dedicated camera for license plate recognition because it is resistant to fog, dark, bright,...

Use other image processing algorithms to better determine license plate position such as Hough transform method for line recognition, color identification, algorithms that limit image movement when the vehicle is moving .

Combined with other programs to manage warehouses, manage vehicles in traffic, find lost vehicles, track,...

# Preference

1. M. J. Ahmed, M. Sarfaz, A. Zidouri, and K. G. AI-Khatib, “License plate recognition system,” *Proc. IEEE Int. Conf. Electron. Circuits, Syst.*, vol. 2, no. January, pp. 898–901, 2003, doi: 10.1109/ICECS.2003.1301932.
2. C. N. E. Anagnostopoulos, “License plate recognition: A brief tutorial,” *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 1, pp. 59–67, 2014, doi: 10.1109/MITS.2013.2292652.
3. A. Badr, M. M. Abdel, A. M. Thabet, and A. M. Abdelsadek, “Automatic number plate recognition system,” *Ann. Univ. Craiova, Math. Comput. Sci. Ser.*, vol. 38, no. 1, pp. 62–71, 2011, doi: 10.5120/ijca2018917277.
4. S. L. Chang, L. S. Chen, Y. C. Chung, and S. W. Chen, “Automatic License Plate Recognition,” *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 1, pp. 42–53, 2004, doi: 10.1109/TITS.2004.825086.
5. N. D. Linh, N. Van Nhan, and D. Van Dat, “12.pdf, ”*Quang Binh Science and Technology Information Magazine”*, 2018.
6. D. V. R. Mohan, M. T. Communication, S. Srkr, and E. College, “Number Plate Recognition by using open CV- Python,” pp. 4987–4992, 2019.
7. Nguyễn Vĩnh An, “Compare some edge detection methods,” Scientific Journal of Hanoi National University”, vol. 31, no. 2, pp. 1–7, 2015.
8. F. Patel, J. Solanki, V. Rajguru, and A. Saxena, “Recognition of Vehicle Number Plate Using Image Processing Technique,” *Adv. Emerg. Med.*, vol. 7, no. 1, pp. 2–8, 2018, doi: 10.18686/aem.v7i1.
9. L. F. Sanchez, “Automatic Number Plate Recognition System Using Machine Learning Techniques,” no. August, pp. 2017–2018, 2018.
10. K. Sarbjit, “An Efficient Approach for Automatic Number Plate Recognition System under Image Processing,” *Int. J. Adv. Res. Comput. Sci.*, vol. 5, no. (6), pp. 43–50, 2014.
11. N. Simin, F. Choong, and C. Mei, “Automatic Car-plate Detection and Recognition System,” pp. 113–114, 2013.
12. G. D. Yeshwant, S. Maiti, and P. B. Borole, “Automatic Number Plate Recognition System (ANPR System),” *Int. J. Eng. Res.*, vol. 3, no. 7, p. 5, 2014, [Online]. Available: https://www.ijert.org/research/automatic-number-plate-recognition-system-anpr-system-IJERTV3IS071132.pdf.
13. A. Zelinsky, *Learning OpenCV---Computer Vision with the OpenCV Library (Bradski, G.R. et al.; 2008)[On the Shelf]*, vol. 16, no. 3. 2009.
14. Chris Dahms (2016), OpenCV 3 License Plate Recognition Python. <https://www.youtube.com/watch?v=fJcl6Gw1D8k>
15. OpenCV. Morphological Transformations.<https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html>
16. Plate car Recognition with opencv step by step. <https://thorpham.github.io/blog/2018/04/11/regconite-plate-car/>
17. Find and draw Contours - OpenCV 3.4 with python 3. <https://www.youtube.com/watch?v=_aTC-Rc4Io0>
18. Contour Features. <https://docs.opencv.org/trunk/dd/d49/tutorial_py_contour_features.html>
19. Learn about Contour, moments in image processing. <https://congdongopencv.blogspot.com/2017/11/tim-hieu-ve-contour-moments-trong-xu-ly.html>
20. Suzuki’s contour tracing algorithm OpenCV - Python. <https://theailearner.com/tag/suzuki-contour-algorithm-opencv/>
21. Chris Dahms (2016), KNN character recognition python. <https://www.youtube.com/watch?v=c96w1JS28AY&t=6s>
22. K-nearest neighbor in opencv2.   
    <https://viblo.asia/p/k-nearest-neighbour-trong-opencv2-V3m5W2wWlO7>
23. Son Nguyen (2017), Machine learning with OpenCV python - 10 K-Nearest Neighbour Algorithms”: <https://www.youtube.com/watch?v=ETOqRZIrLY8>