LOGISTIC REGRESSION

ARTIFICIAL INTELLEGENCE

WE TRAINING

NADA HANAFI AHMED
Computer & Communication Department
Alexandria University

Content

1. Artificial Intelligence

1.1. Definition

Artificial intelligence (AI) refers to the development of computer systems that can perform tasks that would typically require human intelligence to complete. It defines smart algorithms capable of applying human-like decisions automatically and perform miscellaneous activities.

These tasks may include reasoning, problem-solving, learning, perception, decision-making and more. AI's applications are from voice assistants like Siri and Alexa to self-driving cars and medical diagnosis systems as shown in fig1.1.

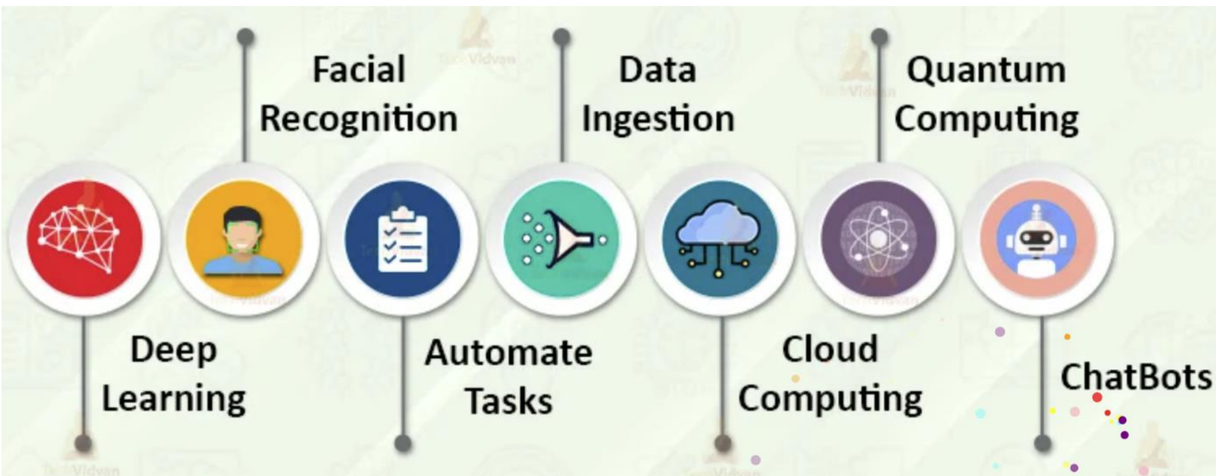

Fig1. 1. Features of AI.

1.2. Types of AI

The most rational way to classify artificial intelligence would be using techniques to compare algorithms' cognitive abilities to those of a person. These techniques enable computer systems to learn from data, recognize patterns and make decisions based on that information, which are:

1. Reactive Machines

2. Limited Memory Machines

3. Theory of Mind

4. Self-Aware AI

1.2.1. Reactive Machines

Reactive Machines uses algorithms that are programmed to perform a specific task like playing chess or converting handwritten messages to editable text. They neither create memories nor predict the future from past experiences, instead they merely respond to current situations.

Applications on Reactive Machines like automatic doors, vending machines, traffic lights chess games, elevators and chatbots.

1.2.2. Limited Memory Machines

Limited Memory Machines algorithms consider both the information they're gathering from the outside world in real time and the data they've been trained on or look into the past experiences and improve them over time, but they are limited in the amount of data that they can store and process.

Applications on Limited Memory Machines like image recognition, natural language processing systems, recommendation engines and fraud detection systems.

1.2.3. Theory of Mind

Theory of mind aims to create algorithms that possess cognitive and social skills such as people's emotions, intentions and behaviors from video data, voice recordings or text messages.

Applications on Theory of Mind like social robots that can understand and respond to human emotions and social cues, educational software that adapts to a student's learning style and pace.

1.2.4. Self-Aware AI

Self-Aware AI is the goal of AI, it develops algorithms acknowledge their existence, besides interpreting the feelings of others, they can develop their own beliefs and desires, including a sense of self-preservation.

Since this type of AI remains a sci-fi concept and only a theory therefore there are no existed applications on it yet.

1.3. Subsets of AI

Artificial intelligence is a broad term that refers to machines' ability to perform tasks that were previously handled by humans only as mentioned earlier in section 1.1, This term is often used in a way that can be exchanged with other words like machine learning and deep learning particularly.

1.3.1. Artificial Intelligence

Computers and software can't behave intelligently unless they've been programmed to do so. Therefore, artificial intelligence is a field of science and engineering which goal is to create intelligent machines that emulate and ultimately exceed the full range of human cognition such as being able to imitate reasoning and continuously learn from new data. AI programs and software can simulate intelligent behavior such as decision-making.

1.3.2. Machine Learning

Machine learning is a subset of artificial intelligence which involves training algorithms through data like images for example and learning through these data so that they can solve problems.

1.3.3. Deep Learning

Deep learning is a subset of artificial intelligence which uses neural networks with multiple hidden layers to extract additional insights from the input data and deliver more accurate predictions. Neural networks feature neurons modelled after a human brain. When receiving input data, the neurons assign to weight to it; the combined weight is then matched against a preset threshold, which denotes the probability of an event. A neural network must contain at least three layers. While deep neural networks may use labeled data to inform their algorithms, they can also learn by themselves.

2. Machine Learning

2.1. Definition

Machine learning (ML) is the field of study that gives computers the capability to learn without being in a clear and detailed manner programmed. It involves the development of algorithms and models that enable computer systems to learn from data and improve their performance at a specific task without being explicitly programmed. Or in other words, machine learning is about creating computer programs that can automatically learn from experience and adapt to new situations. It's done by feeding large amounts of data to algorithms that can identify patterns and relationships within the data, and use that knowledge to make predictions or decisions in new situations.

Machine learning has a wide range of applications, from image and speech recognition to fraud detection and autonomous vehicles as shown in Fig.2.1. and it's a growing field with many possibilities for the future.

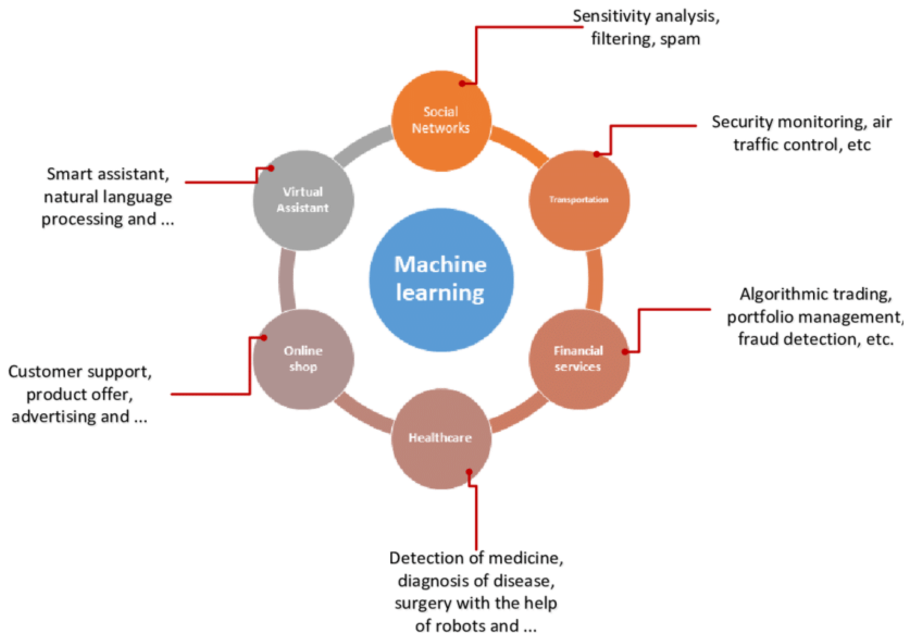

Fig.2.1. Applications on Machine Learning.

2.2. Types of ML

There are three main types of machine learning: supervised, unsupervised and reinforcement as shown in Fig.2.2.

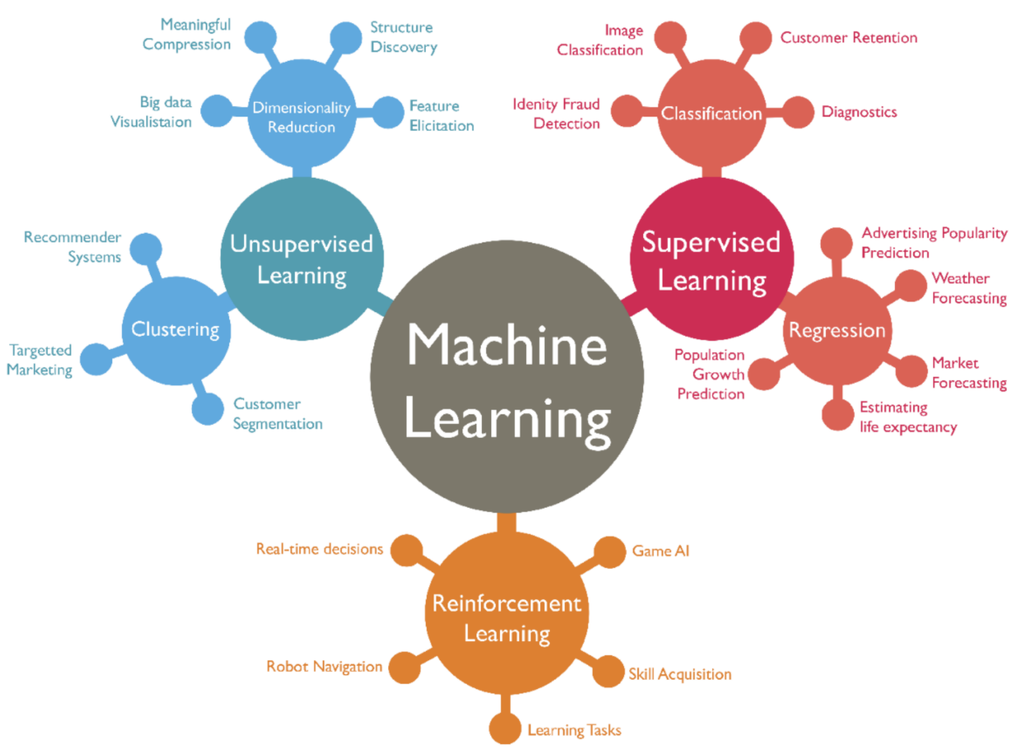

Fig.2.2. Types of ML.

2.2.1. Supervised Learning

Supervised learning involves training a model on labeled data, labeled dataset is one that has both input and output parameters (where the correct output is known), and using that model to make predictions or classifications on new data.

Supervised learning is branched into two types: regression and classification.

2.2.1.1. Regression

In regression (no labels defined) the target variable is a continuous value. The goal of regression is to predict the value of the target variable based on the input variables.

Examples of regression algorithms are linear regression, polynomial regression and decision trees.

2.2.1.2. Classification

In classification (labels defined) the target variable is a categorical value. The goal of classification is to predict the class or category of the target variable based on the input variable based on the input variables.

Examples of classification algorithms are logistic regression, decision trees, support vector machines (SVM), k-nearest neighbors (KNN) and neural networks.

2.2.2. Unsupervised Learning

Unsupervised learning involves training a model on neither classified nor labeled data, (where the correct output is not known), and having the model find patterns and relationships within the data on its own by grouping unsorted information according to similarities, patterns and differences without any prior training of data.

Unsupervised learning is branched into two types: clustering and association.

2.2.2.1. Clustering

Clustering algorithms are used to group similar data points together into clusters, based on their similarity or distance from each other, which can be useful for identifying patterns and structures in data and segmenting the data into different groups or categories.

Examples of clustering algorithms are k-means, hierarchical clustering, principal component analysis, singular value decomposition and independent component analysis.

2.2.2.2. Association

Association algorithms are used to identify relationships or associations between different variables or features in the data. The goal of association is to identify patterns such as frequent item sets which then can be used to make predictions or recommendations.

Examples of association algorithms are apriori, fp-growth and eclat algorithm.

2.2.3. Reinforcement Learning

Reinforcement learning involves training a model through trial and error, where the model receives feedback in the form of rewards or punishments for its actions. The goal of it is to have the model to take actions that maximize its reward over time.

Examples of reinforcement learning algorithms are q-learning, sarsa, deep q-networks, policy gradient methods and actor-critic methods.

2.3. General Steps To Create ML Model

1. Collect data.
2. Prepare data.
    - Data preprocessing.
    - Data wrangling.
3. Choose a model.
4. Train model.
5. Evaluate model.
6. Parameter tuning.
7. Make predictions.

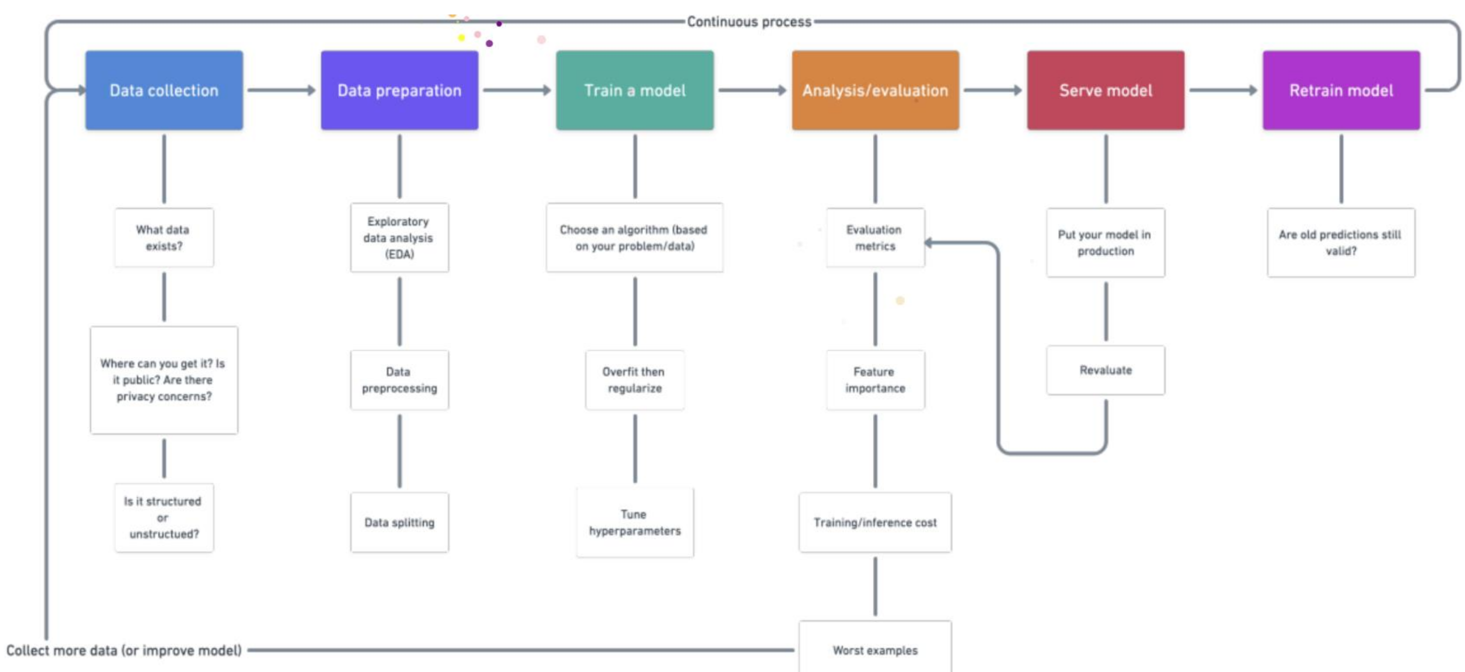

Fig.2.3. Flowchart on how to create ML model.

3. Linear Regression

3.1. Definition

Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables. It can learn from and make predictions on data by learning from the labeled datasets and maps the data points to the most optimized linear functions which can be used for prediction on new datasets. The goal of linear regression is to find the best linear relationship between the independent variables and the dependent variable.

The independent variables are also known as features. When the number of the independent features is 1 then it's known as univariate linear regression, and in the case of more than one feature, it's known as multivariate linear equation that can predict the value of the dependent variable based on the independent variables.

Linear regression is often used for predicting numerical values such as predicting a person's height based on their weight and age or predicting a company's revenue based on its advertising budget and market conditions. It can also be used for understanding the relationship between variables such as understanding the impact of education level on income or the relationship between temperature and air pollution levels.

3.2. Formulation

As mentioned, the equation provides a straight line that represents the relationship between the dependent and independent variables where Y is the dependent or target and X is the independent variable. The slope of the line indicates how much the dependent variable changes for a unit change on the independent variable/s. in regression we must find the value of Y, so a function is required that predicts continuous Y in the case of regression given X as independent features.

The function is written as:

$$\hat{Y} = \sum_{i=1}^{n} ai \; Xi \; + b$$

Where i indicate the number of features, a is the coefficient (slope) which represents the change in the dependent variable for a one-unit change in the corresponding independent variable, and b is the intercept which represents the value of the dependent variable, in other words, when all independent variables are equal to zero, it's the value of the dependent variable that the regression line crosses by then.

4. Logistic Regression

4.1. Definition

Logistic regression is a statistical method used to model the probability of a binary outcome (classification tasks) such as whether a customer will purchase a product or not, based on one or more independent variables.

It's referred to as regression even though it's a classification algorithm because it takes the output of the linear regression function as input and uses a sigmoid function to estimate the probability for the given class.

The difference between linear regression and logistic regression is that linear regression output is the continuous value that can be anything such as numerical values while logistic regression predicts the probability that an instance belongs to a given class or not.

It's used for predicting the categorical dependent variable using a given set of independent variables. Therefore, the output must be a categorical or discrete value. It can be either yes or no, 0 or 1. It gives the probabilistic values which lie in between 0 and 1 instead of fitting a regression line, we fit an S-shaped logistic function which predicts two maximum values (0 or 1).

Logistic regression is a significant machine learning algorithm because it has the ability to classify new data using continuous and discrete datasets.

4.2. Types of Logistic Regression

On the basis of the categories, logistic regression can be classified into three types: binomial, multinomial and ordinal.

4.2.1. Binomial Logistic Regression

In binomial logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, pass or fail, yes or no, etc.

In this case, sigmoid functions are used.

4.2.2. Multinomial Logistic Regression

In multinomial logistic regression, there can be 3 or more possible unordered types of the dependent variables, such as cat, dog or bird.

In this case, softmax function for k classes is used.

4.2.3. Ordinal Logistic Regression

In ordinal logistic regression, there can be 3 or more possible ordered types of the dependent variables, such as low, medium or high.

4.3. Formulation

4.3.1. Sigmoid Function

The sigmoid function is a mathematical function used in logistic regression to map the predicted values to probabilities of a binary outcome. It's also known as the logistic function. It maps any real value into another value within a range of 0 and 1, which can't go beyond this limit so it forms a curve like the "S" form.

The concept of the threshold value which defines the probability is used so that values above the threshold value tends to 1, and values below the threshold value tends to 0.

4.3.2. Parameters Affect Logistic Regression

Common terms involved in logistic regression are independent variables, dependent variable, logistic function, odds, log-odds, coefficient, intercept and maximum likelihood estimation.

4.3.2.1. Independent Variables

The input characteristics, features or predictor factors applied to the dependent variable's predictions.

4.3.2.2. Dependent Variable

The target variable in a logistic regression model, which we are trying to predict.

4.3.2.3. Logistic Function

The formula used to represent how the independent and dependent variables relates to one another, as mentioned above.

4.3.2.4. Odds

It's the ratio of something occurring to something not occurring. It's different from probability as the probability is the ratio of something occurring to everything that could possibly occur. In other words, odds are defined as the probability of the positive outcome divided by the probability of the negative outcome:

$$\text{Odds} = \frac{\mathrm{p}(\mathrm{y}=1)}{p(y=0)}$$

4.3.2.5. Log-Odds

The log-odds, known as the logit function, it's the natural logarithm of the odds. In logistic regression, the log odds of the dependent variable is modeled as a linear combination of the independent variables and the intercept.

$$\text{Log Odds} = \text{Log (Odds)} = \text{Log} \left(\frac{\mathrm{p(y=1)}}{p(y=0)}\right)$$

4.3.2.6. Coefficient

The logistic regression model's estimated parameters, show how the independent and dependent variables relate to one another.

4.3.2.7. Intercept

A constant term in the logistic regression model, which represents the log odds when all independent variables are equal to zero.

4.3.2.8. Maximum Likelihood Estimation

The method used to estimate the coefficients of the logistic regression model, which maximizes the likelihood of observing the data given the model.

4.3.3. How Logistic Regression Works

The logistic regression model transforms the linear regression function continuous value output into categorical value output using a sigmoid function, which maps any real- valued set of independent variables input into a value between 0 and 1.

Let the independent input featured be the matrix X, where:

$$X = \begin{bmatrix} x11 & \cdots & x1m \\ \vdots & \ddots & \vdots \\ xn1 & \cdots & xnm \end{bmatrix}$$

And the dependent variable is Y having only binary value either 0 or 1:

$$Y = \begin{cases} 0 & if\ class\ 1 \\ 1 & if\ class\ 2 \end{cases}$$

Then apply the multi-linear function (linear regression) to the input variables X:

$$Z = \sum_{i=1}^{n} wi\ xi\ + b$$

$$W = wi = [\ w1 \quad w2 \quad w3 \ \ \ldots \quad wn\ ]$$

Where W is the weights of coefficient matrix, and b is the bias or intercept term. It can be simply represented as the dot product of weight and bias.

$$Z = W.X + b$$

Now we use the sigmoid function where the input will be Z and we find the probability between 0 and 1, predicted Y, as shown in Fig.4.3.3.

$$\sigma(Z) = \frac{1}{1+e^{-z}}$$

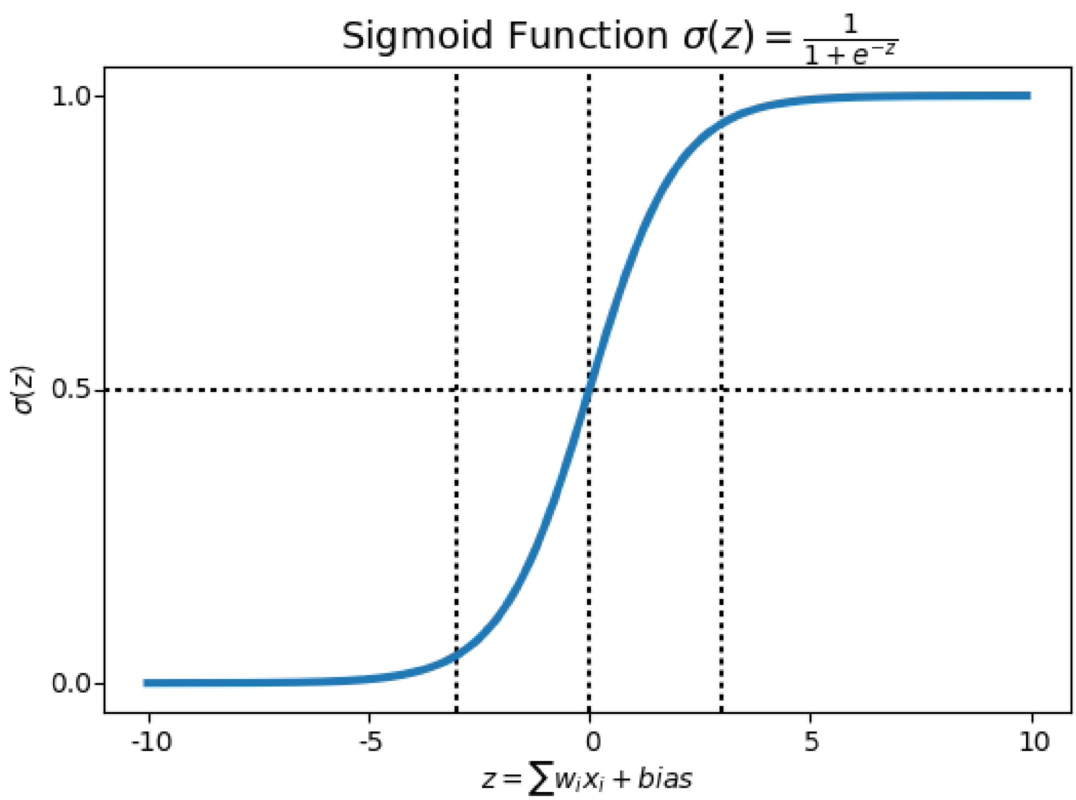

Fig.4.3.3. Sigmoid function graph.

σ(Z) tends towards 1 as Z → ∞

σ(Z) tends towards 0 as Z → -∞

σ(Z) is always bounded between 0 and 1

where the probability of being a class can be measured as:

$$p(y = 1) = \sigma(Z)$$

$$p(y = 0) = 1 - \sigma(Z)$$

4.4. Steps to Create Logistic Regression Model

1. Import required libraries.
2. Load dataset.
3. Prepare dataset.
4. Deal with outliers.
5. Get x (features), y (label). “Clear data”.
6. Split the data into training set and testing set.
7. Scale data.
8. Fit a logistic regression.
9. Apply model and make a prediction.

11. Evaluate using confusion matrix.

10. Calculate accuracy, F1 and precision scores.

4.5. Algorithms

4.5.1. Import required libraries

By using python, the required libraries varies from a model to another but mostly required libraries are

1.numpy

2.matplotlib

- Pyplot

3.pandas

4.seaborn

5.warning

- Warnings.filterwarnings(“ignore”) # this line after importing ignores all the unnecessary warnings “not errors.”

6.sklearn

- model_selection
  - Train_test_split
- linear_model
  - LogisticRegrssion
- Metrics
  - confusion_matrix
  - ConfusionMatrixDisplay
  - classification_report
  - accuracy_score
  - f1_score
  - precision_score
- preprocessing
  - MinMaxScaler
  - StandardScaler

4.5.2. Load Dataset

If the file is already uploaded, you can read dataset directly by writing the line:

```
Datasets = pd.read_csv(‘FILE NAME.csv’)
```

Or by writing the file path

```
Datasets = pd.read_csv(‘r’ PATH WHERE CSV FILE IS
STORED\FILE NAME.csv’)
```

4.5.3. Prepare Dataset

Preparing dataset includes checking the dataset is inserted probably, no null values, check shape and information, converting data types if necessary and encoding.

1.Function used in preparing data

1. `Datasets.head()`
2. `Datasets.info()`
3. `Datasets.describe()`
4. `Datasets.types`

2.Converting data types

```
Datasets[‘COLUMN NAME’] = Datasets[‘COLUMN NAME’].astype(‘REQUIRED TYPE’)
```

3.Null values

To check each individual value if it’s null or not:

```
Datasets.isnull()
```

To check how many null values in each column:

```
Datasets.isnull().sum()
```

To check how many null values in all dataset:

```
Datasets.isnull().sum().sum()
```

Notice that `isna()` function is the same as `isnull()`.

4.Encoding

Convert words to numerical codes.

```
dummy = pd.get_dummies(Datasets['COLUMN'])
```

4.5.4. Deal with outliers

To check for the presence of outliers, plot a boxplot, it can be done by multiple methods.

1.Using the column plot

```
Datasets['COLUMN'].plot(kind = ‘box’)
```

It can also be written as:

```
Datasets['COLUMN'].plot(kind = ‘box’,vert = True)
```

To keep it horizontal:

```
Datasets['COLUMN'].plot(kind = ‘box’,vert = False)
```

2.Using pyplot

```
Plt.plot(Datasets[‘COLUMN’])
```

It can also be written as:

```
Plt.plot(Datasets[‘COLUMN’],vert = True)
```

To keep it horizontal:

```
Plt.plot(Datasets[‘COLUMN’],vert = False)
```

3.Using seaborn

```
sns.boxplot(Datasets[‘COLUMN’])
```

It can also be written as:

```
sns.boxplot(y = Datasets[‘COLUMN’])
```

To keep it horizontal:

```
sns.boxplot(x = Datasets[‘COLUMN’])
```

After checking for outliers, if found, there are three methods to deal with outliers, depends on the data itself.

1.SQRT Transformation

```
Datasets['SQRT_COLUMN'] = Datasets['COLUMN']**0.5
```

Or by using numpy function

```
Datasets['sqrt_COLUMN'] = np.sqrt(Datasets['COLUMN'])
```

2.Log Transformation

```
Datasets['log_COLUMN'] = np.log(Datasets['COLUMN'])
```

3.IQR Transformation

All the values below Q1 - 1.5*IQR and values above Q3 + 1.5*IQR are outliers and can be removed.

finding the Quantiles:

```
Q1 = Datasets.COLUMN.quantile(0.25)
Q2 = Datasets.COLUMN.quantile(0.50)
Q3 = Datasets.COLUMN.quantile(0.75)
```

IQR : Inter-Quartile Range

```
IQR = Q3 - Q1
```

Lower Limit:

```
LC = Q1 - (1.5*IQR)
```

Upper Limit:

```
UC = Q3 + (1.5*IQR)
```

4.5.5. Get X and Y

Getting X and Y depends on the columns you want to be the features and the label variables.

It can be done by different methods. For example:

1. using drop function

if the whole dataset wanted to be features except the label column “can still be used if other columns wanted to be dropped”.

```
X = Datasets.drop([“LABEL OR IGNORED COLUMN NAME”], axis=1)
Y = Datasets[“LABEL NAME”]
```

2. using iloc function

iloc function basically cuts the dataset into sections based on the start and end values.

```
X = Datasets.iloc[:, [START COLUMN,END]].values
Y = Datasets.iloc[:, LABEL COLUMN ORDER].values
```

3. using range

```
X = Datasets.values[:, range(START,END)]

Y = Datasets.values[:,LABEL COLUMN ORDER]
```

Notice that before setting X and Y values, dataset needs to be clean and prepared first. Which was discussed above how.

4.5.6. Split the data into training set and testing set

```
X_Train, X_Test, Y_Train, Y_Test = train_test_split(X,
        Y, test_size = 0.25, random_state = 0)
```

4.5.7. Scale Data

There are two methods to scale data:

1.Normalization (Min Max Scaler)

By using formula:

$$\frac{COLUMN - min}{max - min}$$

```
Minimum = Datasets['COLUMN'].min()

Maximum = Datasets['COLUMN'].max()

Datasets['MinMax'] = ( Datasets['COLUMN'] - Minimum ) /
        ( Maximum - Minimum )
```

Or by using sklearn library:

```
MS = MinMaxScaler()

MinMaxScaled = MS.fit_transform(Datasets['COLUMN'])
```

2.Standardization (Z-Score Scaler)

By using formula:

$$\frac{COLUMN - mean}{standard\ deviation}$$

```
Avg = Datasets['COLUMN'].mean()

Std_COLUMN = Datasets['COLUMN'].std()

Datasets['ZScore'] = ( Datasets['COLUMN']-Avg )/Std_COLUMN
```

Or by using sklearn library:

```
SS = StandardScaler()

Scaler=SS.fit_transform(Datasets['COLUMN'])
```

Most suitable way to scale and fit a logistic regression is:

```
sc_X = StandardScaler()

X_Train = sc_X.fit_transform(X_Train)

X_Test = sc_X.transform(X_Test)
```

4.5.8. Fit Logistic Regression

```
classifier = LogisticRegression(random_state = 0)

classifier.fit(X_Train, Y_Train)
```

4.5.9. Make a prediction

```
Y_Pred = classifier.predict(X_Test)
```

4.5.10. Evaluate using confusion matrix

Confusion matrix is a matrix that summarizes the performance of a machine learning model on a set of test data. It displays the number of true positive, true negative, false positive and false negative data produced by the model on the test data.

```
cm = confusion_matrix(Y_Test, Y_Pred)

show = pd.DataFrame(cm,columns=["yes","no"],index=["yes","no"])
```

```
cmd = ConfusionMatrixDisplay(cm)

cmd.plot()
```

4.5.11. Classification Report Calculations

```
print(classification_report(Y_Test, Y_Pred))

print("Accuracy=", accuracy_score(Y_Test, Y_Pred))

print("F1=", f1_score(Y_Test, Y_Pred))

print("Precision=", precision_score(Y_Test, Y_Pred))
```