Course: Machine Learning Techniques (CSE3008) - Module1

Dr.Mohana S D,
Assistant Professor,
(Course In-charge - CSE3008)
School of Computer Science and Engineering & Information Science,
Presidency University Bengaluru.

Machine Learning

- Machine learning (ML) is a type of artificial intelligence that enables machines to learn and improve from experience without being explicitly programmed.
- ML algorithms can be divided into three main categories: supervised learning, unsupervised learning, and reinforcement learning.
- Supervised learning involves training a model on a labeled dataset, where the input features are labeled with the corresponding output values.
- Unsupervised learning involves training a model on an unlabeled dataset, where the goal is to discover patterns and structure in the data.
- Reinforcement learning involves training a model to make decisions by interacting with an environment and receiving feedback in the form of rewards or penalties.

ML Workflow

- The ML workflow typically consists of the following steps: data collection, data cleaning, feature engineering, model selection, model training, model evaluation, and model deployment.
- Data collection involves gathering data from various sources, such as databases, APIs, and web scraping.
- Data cleaning involves preprocessing the data to remove missing values, outliers, and other anomalies that could affect the model's performance.
- Feature engineering involves selecting and transforming the input features to improve the model's performance. This can include techniques such as feature scaling, one-hot encoding, and dimensionality reduction.

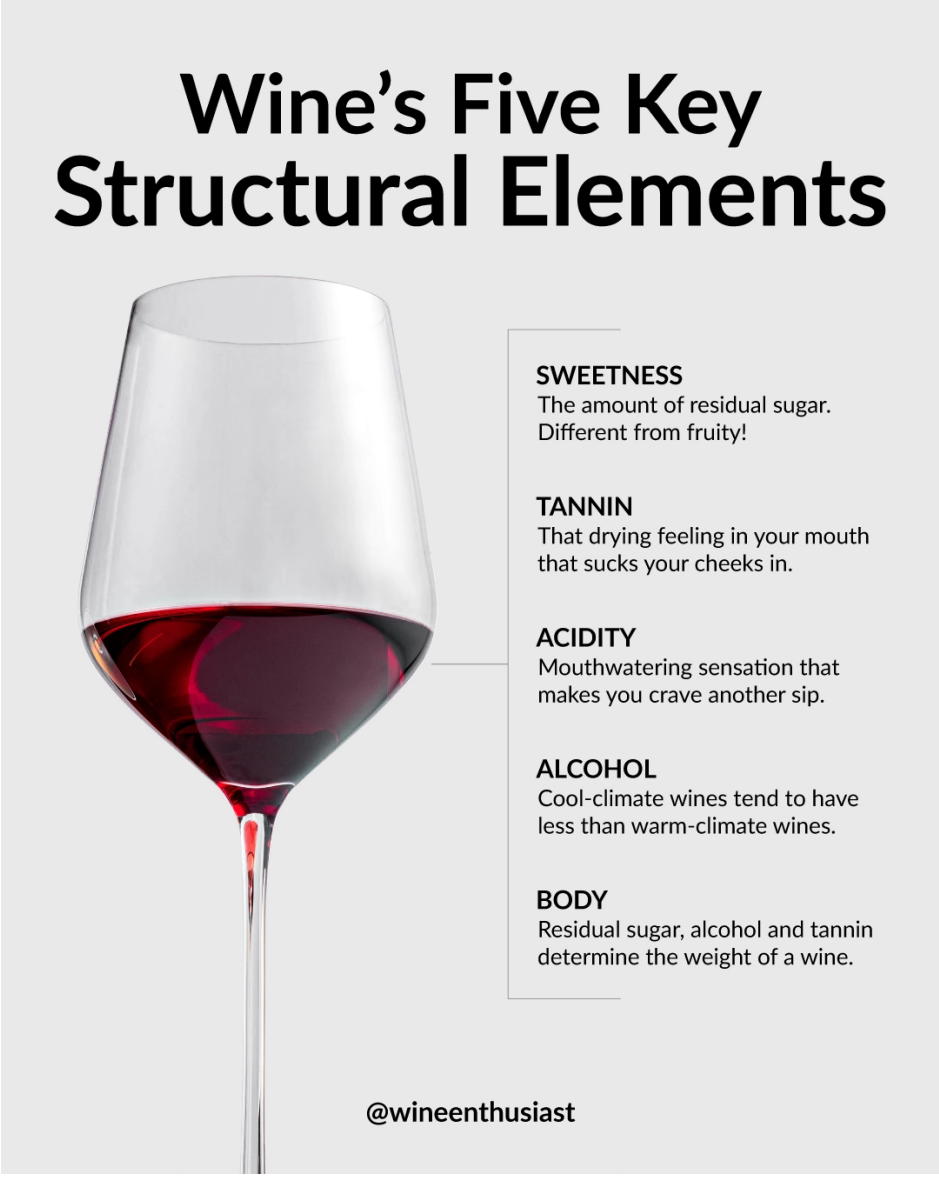
ML Workflow

- Model selection involves choosing the appropriate algorithm for the task at hand. This can involve experimenting with different algorithms and comparing their performance.
- Model training involves fitting the model to the training data using an optimization algorithm such as gradient descent.
- Model evaluation involves testing the model on a held-out validation set to assess its performance.
- Model deployment involves integrating the trained model into a larger system or application.

Knowledge Representation

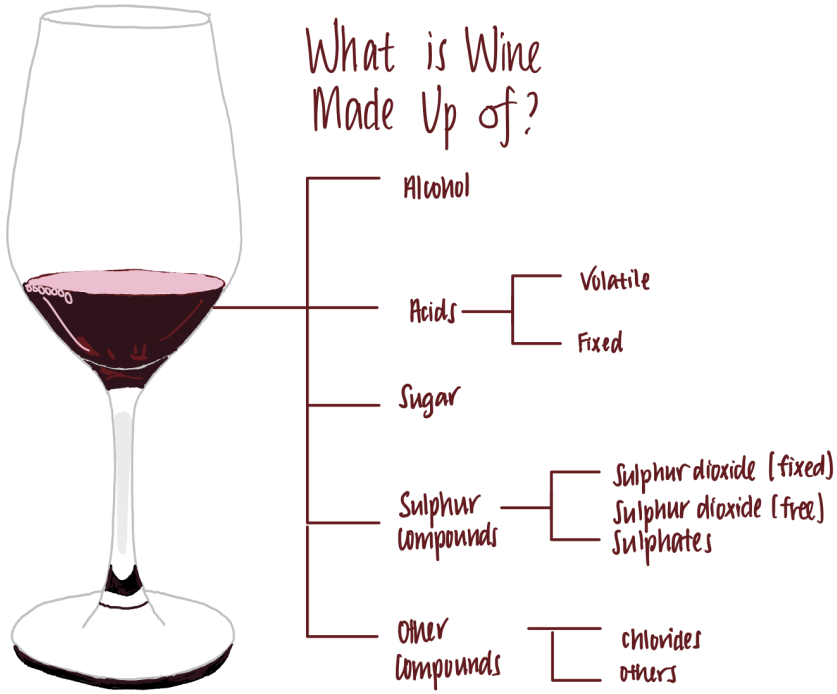
- Wine quality is a subjective measure of the overall quality of a wine, based on factors such as its taste, aroma, and appearance.
- There are many different factors that can affect wine quality, including the grape variety, the region where the grapes were grown, the weather conditions during the growing season, and the winemaking process.
- Wine quality is typically rated on a scale of 0 to 100, with higher scores indicating better quality.

Knowledge Representation



Knowledge Representation

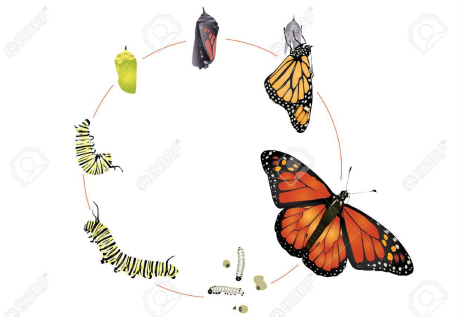
- Color: deep red
- Aroma: ripe blackberries and vanilla
- Taste: full-bodied with firm tannins and a long finish
- Ratings: expert- 93 rated by a wine expert and consumer: 85 rated by a consumer



Types of Features

- Features can be divided into two main categories: categorical and numerical.
- Categorical features represent discrete values, such as colors, types, and categories.
- Numerical features represent continuous values, such as measurements, counts, and percentages.
- Features can also be divided into binary, ordinal, and interval/ratio types, depending on their characteristics and properties.

Data Transformation



Natural Language Processing

by National Research University Higher School of Economics

We can count token pairs, triplets, etc.

- Also known as n-grams
  - 1-grams for tokens
  - 2-grams for token pairs
  - …

	good movie	movie	did not	a	…
good movie	1	1	0	0	…
not a good movie	1	1	0	1	…
did not like	0	0	1	0	…

Data Transformation

- Image data transformation: Image data is typically represented as a matrix of pixel values, and a common transformation is to normalize the pixel values so they have a mean of zero and a standard deviation of one.

Data Transformation

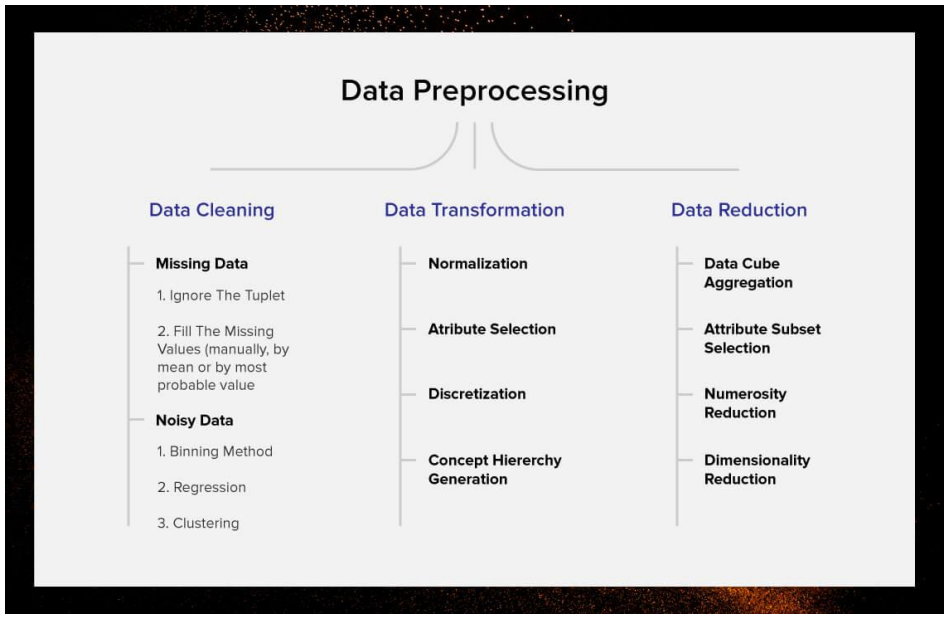
- Text data transformation: A common technique is to represent the text as a bag-of-words, which involves creating a dictionary of all the unique words in the text and representing each document as a vector of word frequencies. The text as a sequence of n-grams (i.e. contiguous sequences of words or characters) and using word embeddings (i.e. dense vectors that represent the meaning of a word) to capture the semantic relationships between words.

Data Transformation

- Numerical data transformation:Numerical data typically requires scaling and normalization to ensure that all the input features have a similar scale and range. Common techniques include min-max scaling (i.e. scaling the values to be between 0 and 1), z-score normalization (i.e. scaling the values to have a mean of 0 and a standard deviation of 1), and logarithmic transformation (i.e. transforming the values to their logarithm to reduce skewness in the distribution). Other techniques include feature selection (i.e. selecting a subset of the most relevant input features) and feature engineering (i.e. creating new features by combining or transforming existing features).

Feature Engineering

- Feature engineering is the process of selecting and transforming the input features to improve the model's performance.
- Some common feature engineering techniques include feature scaling, one-hot encoding, and dimensionality reduction.
- Feature scaling involves scaling the numerical features to a common scale to prevent bias in the model.



Methods of Data Imputation

- Mean Imputation: Replace missing values with the mean of the non-missing values in the variable.
- Regression Imputation: Predict the missing values using a regression model based on the other variables in the dataset.
- Multiple Imputation: Create multiple imputed datasets and combine the results for analysis.
- K-Nearest Neighbor (KNN) Imputation: Predict the missing values using the values of the nearest neighbors in the dataset.

Data Imputation

- Data imputation is a statistical technique used to fill in missing data points in a dataset.
- There are several methods for data imputation, but one common approach is to use a statistical model to estimate the missing values based on the available data.

Example

- Suppose we have a dataset with five data points: 3, 6, 7, NaN, 9, where "NaN" represents a missing data point. We want to impute the missing value using a simple linear regression model. The linear regression model is given by:
  $y = mx + b$

- where y is the dependent variable (the missing value we want to impute), x is the independent variable (in this case, the index of the data point), m is the slope of the line, and b is the y-intercept.

To estimate the missing value, we first need to fit the linear regression model to the available data. We can do this by using the least squares method, which finds the values of m and b that minimize the sum of the squared differences between the predicted values and the actual values of the dependent variable. The formula for the slope m and y-intercept b are given by:

- $m = \frac{(n\sum(xy) - \sum x \sum y)}{(n\sum(x^2) - (\sum x)^2)}$
- $b = \frac{(\sum y - m\sum x)}{n}$

where n is the number of data points (excluding the missing value), $\sum$ represents the sum over all data points, x and y represent the independent and dependent variables, respectively, and xy represents the product of x and y.

Using the available data, we can calculate the values of m and b as follows:

$n = 4$

$\sum x = 0 + 1 + 2 + 4 = 7$

$\sum y = 3 + 6 + 7 + 9 = 25$

$\sum xy = (03) + (16) + (27) + (49) = 44$

$\sum x^2 = 0^2 + 1^2 + 2^2 + 4^2 = 21$

$m = (444 - 725) / (421 - 7^2) \approx 1.67$

$b = (25 - 1.677) / 4 \approx 2.54$

- Now that we have the values of m and b, we can use the linear regression model to estimate the missing value. Since the missing value has an index of 3, we can substitute $x = 3$ into the linear regression equation to obtain:
- $y = 1.67 \star 3 + 2.54 \approx 7.15$
- Therefore, we can impute the missing value with an estimate of 7.15 based on the available data and our linear regression model.

K-Nearest Neighbor (KNN) Imputation

- The KNN algorithm selects the k most similar observations to the one with the missing value and then takes the average or weighted average of the values of these observations to fill in the missing value.

- Calculate the distance between the observation with the missing value and all other observations in the dataset using a distance metric (e.g. Euclidean distance or Manhattan distance). Let's call the observation with the missing value O.
- Select the k observations with the smallest distance to O. Let's call these observations S.
- Take the average or weighted average of the values of the missing variable (income) for the k observations in S to impute the missing value for O.

The formula for calculating the Euclidean distance between two observations X and Y with n variables can be written as:

$$d(X, Y) = \sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2}$$

where X_i and Y_i are the values of the i-th variable in observations X and Y, respectively.

To impute the missing value for O using KNN imputation, we can use the following formula to calculate the average value of the income variable for the k observations in S:

$$\hat{Y}O = \frac{1}{k}\sum i = 1^k Y_{S_i}$$

where $\hat{Y}O$ is the imputed value for the missing income variable in observation O, YS_i is the value of the income variable for the i-th observation in S, and k is the number of observations in S.

What is Regression?

Regression is a statistical modeling technique used to explore the relationship between a dependent variable and one or more independent variables. It is used to model and predict the value of the dependent variable based on the values of the independent variables.

- Regression is a type of statistical analysis used to model the relationship between a dependent variable and one or more independent variables.
- The goal of regression is to find a function that can accurately predict the value of the dependent variable based on the values of the independent variables.

Simple Linear Regression:

- Simple linear regression is a type of regression analysis that models the relationship between a dependent variable and a single independent variable.
- It assumes a linear relationship between the two variables and uses a line to model the relationship.
- The line is fitted to the data using a least squares regression approach, which minimizes the sum of the squared differences between the predicted values and the actual values of the dependent variable.

Simple Linear Regression

Simple linear regression is a type of regression where there is only one independent variable. The goal of simple linear regression is to find the line of best fit that minimizes the difference between the predicted values and the actual values of the dependent variable.

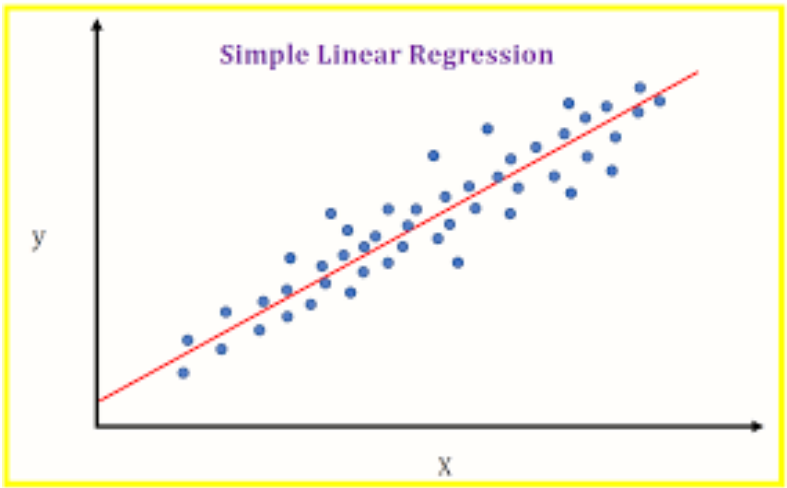


Figure: Simple Linear Regression

Loss Functions

In order to find the line of best fit in simple linear regression, we need to define a loss function that measures the difference between the predicted values and the actual values of the dependent variable.

Two commonly used loss functions in simple linear regression are the Mean Squared Error (MSE) and the Mean Absolute Error (MAE).

- MSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- MAE $= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$

Loss Functions:

- Loss functions are used in regression to quantify the difference between the predicted values and the actual values of the dependent variable.
- The goal of regression is to minimize the loss function by adjusting the parameters of the model.
- The most commonly used loss function in simple linear regression is the mean squared error (MSE), which is the average of the squared differences between the predicted values and the actual values.
- Other loss functions include mean absolute error (MAE) and root mean squared error (RMSE).

Polynomial Regression

Polynomial regression is a type of regression where the relationship between the dependent variable and the independent variable(s) is modeled as an nth degree polynomial. This allows for a more complex relationship between the variables than simple linear regression.

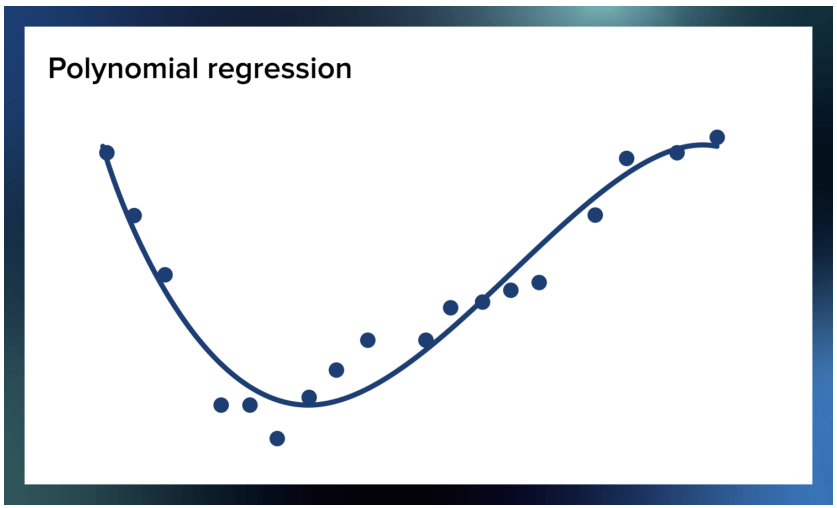


Figure: Polynomial Regression

Polynomial Regression:

- Polynomial regression is a type of regression analysis that models the relationship between a dependent variable and one or more independent variables using a polynomial function.
- It allows for more complex relationships between the variables by fitting a curve to the data instead of a straight line.
- The polynomial function can be of any degree, but higher degree functions can overfit the data and result in poor performance on new data.
- To prevent overfitting, techniques such as cross-validation can be used to evaluate the performance of the model on new data.

- Regression is a type of statistical analysis used to model the relationship between a dependent variable and one or more independent variables.
- The goal of regression is to find a function that can accurately predict the value of the dependent variable based on the values of the independent variables.
- One type of regression is Simple Linear Regression, where there is only one independent variable.
- The goal of Simple Linear Regression is to find the line of best fit that minimizes the difference between the predicted values and the actual values of the dependent variable. This line can be expressed as:

Eq:

$y = \beta 0 + \beta 1 * x$

where y is the dependent variable, x is the independent variable, $\beta 0$ is the intercept, and $\beta 1$ is the slope.

$$y = b_0 + b_1 x$$

$$b_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - b_1 \bar{x}$$

- In order to find the line of best fit in Simple Linear Regression, we need to define a loss function that measures the difference between the predicted values and the actual values of the dependent variable.
- Two commonly used loss functions in Simple Linear Regression are the Mean Squared Error (MSE) and the Mean Absolute Error (MAE).

Suppose we have a dataset of n observations (x, y), where x is the independent variable and y is the dependent variable. We want to fit a polynomial model to the data, which will allow us to make predictions for new values of x.

Polynomial regression is a form of linear regression where the relationship between x and y is modeled as an nth-degree polynomial. That is, we assume that there exists a polynomial of the form:

$$y = b_0 + b_1 x + b_2 x^2 + \cdots + b_n x^n$$

where $b_0, b_1, b_2, \ldots, b_n$ are the coefficients of the polynomial.

To find the coefficients of the polynomial, we can use the method of least squares. We want to minimize the sum of the squared errors between the predicted values and the actual values of y:

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where y_i is the actual value of the dependent variable and $\hat{y}_i$ is the predicted value of the dependent variable. Substituting the polynomial equation for $\hat{y}_i$, we get:

$$SSE = \sum_{i=1}^{n} \left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 + \cdots + b_n x_i^n\right)\right)^2$$

To find the values of $b_0, b_1, b_2, \ldots, b_n$ that minimize the sum of the squared errors, we take the partial derivative of SSE with respect to each coefficient and set it equal to 0:

$$\frac{\partial SSE}{\partial b_0} = -2\sum_{i=1}^{n}\left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 + \cdots + b_n x_i^n\right)\right)$$

$$\frac{\partial SSE}{\partial b_1} = -2\sum_{i=1}^{n} x_i\left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 + \cdots + b_n x_i^n\right)\right)$$

$$\frac{\partial SSE}{\partial b_2} = -2\sum_{i=1}^{n} x_i^2\left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 + \cdots + b_n x_i^n\right)\right)$$

$$\vdots$$

$$\frac{\partial SSE}{\partial b_n} = -2\sum_{i=1}^{n} x_i^n \left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 + \cdots + b_n x_i^n\right)\right)$$

These equations can be solved using linear algebra to find the values of $b_0, b_1, b_2, \ldots, b_n$ that minimize the sum of the squared errors.

$$y = b_0 + b_1 x + b_2 x^2$$

$$\frac{\partial SSE}{\partial b_0} = -2 \sum_{i=1}^{n} \left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 \right) \right)$$

$$\frac{\partial SSE}{\partial b_1} = -2 \sum_{i=1}^{n} x_i \left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 \right) \right)$$

$$\frac{\partial SSE}{\partial b_2} = -2 \sum_{i=1}^{n} x_i^2 \left(y_i - \left(b_0 + b_1 x_i + b_2 x_i^2 \right) \right)$$

Introduction to Logistic Regression

- Logistic regression is a popular method for binary classification, where the goal is to predict the probability of an input belonging to a particular class.
- The logistic regression model uses a logistic function to model the probability of an input belonging to the positive class.
- The logistic function maps any input to a value between 0 and 1, which can be interpreted as a probability.

$$p(y = 1|x) = \frac{1}{1 + e^{-z}} \tag{1}$$

where $z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$ is the linear combination of the input features $x_1, x_2, \ldots, x_n$ and their corresponding coefficients $\theta_1, \theta_2, \ldots, \theta_n$.

Softmax Regression

- Softmax regression is a generalization of logistic regression that can be used for multi-class classification problems.
- The softmax function is used to compute the probability of each class, and the class with the highest probability is chosen as the predicted class.
- The softmax function outputs a probability distribution over the classes, and its outputs sum to 1.

$$p(y = i|x) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2}$$

where K is the number of classes, z_i is the linear combination of input features for class i, and the denominator sums over all classes.

Cross Entropy Loss

- To train a logistic regression or softmax regression model, we need a cost function that measures the difference between the predicted probabilities and the true labels.
- The cross entropy loss is a popular choice for such a cost function.

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_{i,k}\log\left(h_\theta(x_i)_k\right) + (1-y_{i,k})\log\left(1-h_\theta(x_i)_k\right) \quad (3)$$

where m is the number of training examples, K is the number of classes, $y_{i,k}$ is the true label of example i for class k, and $h_\theta(x_i)_k$ is the predicted probability of example i belonging to class k.

Introduction to Bayesian Learning

- Bayesian learning is a popular method for modeling data and making predictions using probabilistic reasoning.
- The key idea behind Bayesian learning is to use Bayes' theorem to compute the posterior probability of a hypothesis given some observed data.

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \tag{4}$$

where h is the hypothesis, D is the observed data, $P(D|h)$ is the likelihood of the data given the hypothesis, $P(h)$ is the prior probability of the hypothesis, and $P(D)$ is the marginal probability of the data.

Estimating Conditional Probabilities

- In order to use Bayes' theorem, we need to estimate the conditional probabilities of the data given the hypothesis and the prior probability of the hypothesis.
- For categorical features, we can estimate these probabilities using a frequency table.

	X_1	X_2	Y
1	1	1	1
2	1	0	0
3	0	1	0
4	1	1	1
5	0	0	0
6	1	0	1
7	0	1	0

$$P(X_1 = 1|Y = 1) = \frac{3}{4}, \quad P(X_2 = 0|Y = 1) = \frac{1}{4} \tag{5}$$

Estimating Conditional Probabilities (cont.)

- For continuous features, we can estimate these probabilities using a probability density function.
- A common choice is the Gaussian (normal) distribution.

$$P(X = x|Y = y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu_y)^2}{2\sigma^2}\right) \tag{6}$$

where μ_y and σ^2 are the mean and variance of X for the data points where $Y = y$.

- Naïve Bayes is a popular algorithm for supervised learning, particularly for text classification problems. It's based on Bayes' theorem, which is a formula for calculating conditional probabilities.
- The basic idea behind Naïve Bayes is to calculate the probability of a particular class given some input features.
- This is done by calculating the conditional probabilities of each feature given the class, and then using Bayes' theorem to calculate the probability of the class given the features.

- The "naïve" part of Naïve Bayes comes from the assumption that the input features are conditionally independent given the class.
- This assumption simplifies the calculation of the conditional probabilities, as the joint probability of all the features can be written as the product of the individual conditional probabilities.
- One common application of Naïve Bayes is in spam filtering.
- The input features are typically the frequency of certain words in an email, and the class is either "spam" or "not spam". Naïve Bayes can be used to calculate the probability that an email is spam given its word frequencies.

- Bayesian belief networks are a type of graphical model that can be used to represent probabilistic relationships between variables.
- Each variable is represented by a node in the network, and the edges between the nodes represent conditional dependencies.
- The basic idea behind Bayesian belief networks is to use Bayes' theorem to calculate the probabilities of each variable given the values of its parents in the network.
- This can be done by factorizing the joint probability distribution of all the variables into a product of conditional probabilities.

- One common application of Bayesian belief networks is in medical diagnosis.
- The variables in the network might represent symptoms and diseases, and the edges represent the conditional dependencies between them.
- Given a set of observed symptoms, the network can be used to calculate the probability of each possible disease.

- Naïve Bayes and Bayesian belief networks are powerful tools for probabilistic modeling and inference.
- Naïve Bayes is particularly useful for text classification problems, while Bayesian belief networks are useful for modeling complex dependencies between variables.
- With these tools, we can make more accurate predictions and better understand the relationships between different variables in our data.

- Support Vector Machines (SVMs) are a popular machine learning algorithm for classification and regression. They work by finding the hyperplane that maximally separates the data points in a high-dimensional feature space.
- The basic idea behind SVMs is to find the hyperplane that maximizes the margin between the positive and negative data points. The margin is the distance between the hyperplane and the closest data points from either class. The hyperplane that maximizes the margin is called the maximum margin hyperplane.
- SVMs can be extended to handle non-linearly separable data using a technique called the kernel trick. The kernel trick involves mapping the original feature space to a higher-dimensional space using a non-linear function, and then finding the maximum margin hyperplane in this new space.

- In practice, data is often not perfectly separable, and finding the maximum margin hyperplane is not always possible. In these cases, we can use a variant of SVM called the soft margin SVM.
- The soft margin SVM allows for some misclassifications, and introduces a slack variable that penalizes data points that are on the wrong side of the margin.
- The objective function for the soft margin SVM includes a regularization term that controls the tradeoff between maximizing the margin and minimizing the classification error.

- The kernel trick is a powerful technique for extending SVMs to handle non-linearly separable data. The basic idea is to map the data into a higher-dimensional feature space using a non-linear function, and then apply the linear SVM algorithm to this new feature space.
- The key insight behind the kernel trick is that we can compute the dot product between the mapped data points without explicitly computing the mapping. This is done by defining a kernel function that takes two data points as input and returns the dot product of their mapped features.
- Some common kernel functions include the polynomial kernel, which computes the dot product of two vectors raised to a certain power, and the radial basis function kernel, which measures the similarity between two data points based on their distance in the feature space.

- Support Vector Machines are a powerful machine learning algorithm that can handle both linear and non-linearly separable data.
- The kernel trick allows us to extend SVMs to handle complex data, and the soft margin SVM allows us to handle data that is not perfectly separable.
- With these techniques, we can build powerful models for classification and regression tasks.