ONLINE AD CLICK PREDICTION

A Project Report for Industrial Training and Internship

submitted by

Dilkhush Kumar - 230205011017

Neha Kumari - 230205011004

In the partial fulfillment of the award of the degree of

Bachelor of Computer Applications (BCA)

in the

COMPUTER SCIENCE AND ENGINEERING

Of

SANDIP UNIVERSITY

At

Ardent Computech Pvt. Ltd.

ARDENT COMPUTECH PVT. LTD.
High-End Technology Training and Project

Ardent Computech Pvt. Ltd.
Drives you to the Industry
Module 132, SDF Building, Sector V, Salt Lake, Pin - 700091
www.ardentcollaborations.com

CERTIFIED ISO 9001:2015 CERTIFIED

UDYAM REGISTRATION MSME

N·E·A·T National Educational Alliance for Technology

CERTIFICATE FROM SUPERVISOR

This is to certify that **“Dilkhush Kumar, Neha Kumari, 230205011017, 230205011004**” have completed the project titled "**Online Ad Click Prediction**" under my supervision during the period from “25/06/2025” to “30/07/2025” which is in partial fulfillment of requirements for the award of the **BCA** degree and submitted to the Department of “**COMPUTER SCIENCE AND ENGINEERING**” of **“SANDIP UNIVERSITY”**.

____

Signature of the Supervisor

Date:

Name of the Project Supervisor:

22 YEARS OF EXCELLENCE

ARDENT
COMPUTECH PVT. LTD.
High-End Technology Training and Project

Ardent Computech Pvt. Ltd.
Drives you to the Industry
Module 132, SDF Building, Sector V, Salt Lake, Pin - 700091
www.ardentcollaborations.com

CERTIFIED ISO 9001:2015 CERTIFIED

UDYAM REGISTRATION
MSME

N·E·A·T
National Educational Alliance for Technology

BONAFIDE CERTIFICATE

Certified that this project work was carried out under my supervision

"Online Ad Click Prediction" is the bonafide work of

Name Of Student: *Dilkhush Kumar* ***Signature:*** Dilkhush Kumar

Name Of Student: *Neha Kumari* ***Signature:*** Neha Kumari

SIGNATURE

Name : SOURAV GOSWAMI

PROJECT MENTOR

SIGNATURE

Name:

EXAMINERS

Ardent Original Seal

ACKNOWLEDGEMENT

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful.

We take this opportunity to express our deep gratitude towards our project mentor, **Mr. Sourav GoswamiSir** for giving such valuable suggestions, guidance and encouragement during the development of this project work.

Last but not the least we are grateful to all the faculty members of **Ardent Computech Pvt. Ltd.** for their support.

Table of Contents

Introduction

In today's digital age, online advertising plays a crucial role in the business strategies of companies around the world. From small startups to large multinational corporations, organizations rely on online ads to reach their target audience, promote products, and increase sales. However, not every ad impression results in a click. Understanding which ads are likely to be clicked by users is a significant challenge — and solving this problem can save billions in advertising costs while improving the effectiveness of marketing campaigns. This is where **Online Ad Click Prediction** becomes essential.

Online Ad Click Prediction refers to the process of using machine learning and data analysis techniques to predict whether a user will click on a particular advertisement. It involves analyzing historical user data, ad features, browsing behavior, and other contextual information to forecast the likelihood of a click. The goal is to optimize ad placements and increase conversion rates by targeting users who are most likely to engage with the content.

The rise of big data and the availability of user interaction logs have made it possible to build intelligent systems that learn from past behavior. For example, features such as user demographics, time of the day, type of device, and previous interactions with ads are commonly used as input to predictive models. Machine learning algorithms like Logistic Regression, Decision Trees, Random Forests, Gradient Boosting, and even Deep Learning models like Neural Networks are employed to analyze these features and make accurate predictions.

One of the major challenges in ad click prediction is the **high imbalance in data**. Typically, only a small fraction of users click on ads, while the majority do not. This imbalance can lead to biased models that favor the majority class (non-clicks). Therefore, techniques such as **oversampling**, **undersampling**, and **custom evaluation metrics** like AUC-ROC, precision-recall, and F1-score are used to ensure the model performs well on both classes.

The applications of accurate ad click prediction are vast. For advertisers, it means better Return on Investment (ROI), as they can focus their budgets on high-performing ads. For users, it results in a more personalized and less intrusive ad experience. For platforms like Google, Facebook, and Amazon, it enables more effective ad auctions and higher satisfaction among both users and advertisers.

Moreover, the process of building such a system also raises important considerations around **privacy**, **data ethics**, and **fairness**. As these models rely heavily on user data, it is crucial to ensure that personal information is handled securely and ethically.

In conclusion, Online Ad Click Prediction is a powerful use case of machine learning in the real world. It combines data science, behavioral analytics, and business strategy to improve digital advertising. As the online ecosystem continues to evolve, the importance of accurate and efficient ad targeting will only grow — making ad click prediction an essential area of research and development in the field of artificial intelligence and marketing.

History of Online Ad Click Prediction

The concept of **Online Ad Click Prediction** evolved alongside the growth of the internet and the rise of digital marketing. Initially, advertising was mostly traditional — done through newspapers, magazines, television, and radio. With the introduction of the World Wide Web in the early 1990s, a new opportunity emerged: **online advertising**.

1. Early Days (1994–2000): Static Ads and Simple Tracking

The first online ad is widely believed to be a **banner ad** published in 1994 by AT&T on the website *HotWired*. In the beginning, these ads were static and not personalized. Advertisers would track **Click-Through Rate (CTR)** manually — calculated as the number of clicks divided by the number of impressions. There was **no predictive modeling**, just basic analytics like counting views and clicks.

2. Rise of Contextual Ads (2000–2005): Basic Targeting

As search engines like **Google** grew, the advertising landscape changed dramatically. Google introduced **AdWords (2000)** and **AdSense (2003)** — platforms that showed ads based on the **context of the webpage** and user queries. These systems began using **basic algorithms** and **rules** to match ads to content.

During this time, **logistic regression** and other simple statistical models were first introduced to predict whether users would click on an ad. Still, these models used limited features like keywords, location, and time.

3. User-Based Targeting (2005–2012): Machine Learning Enters

With the growth of e-commerce and social media platforms like **Facebook**, advertisers started collecting more **user data** — such as browsing history, age, gender, location, and interests. This allowed for **behavioral targeting**.

Machine Learning started becoming popular in this phase. Models such as:

- Decision Trees
- Random Forest
- Support Vector Machines (SVM)

were used to **predict ad click behavior** based on rich datasets. Platforms like **Yahoo!**, **Microsoft**, and **Google** began investing in predictive modeling to improve ad relevance.

4. Big Data & Real-Time Bidding (2012–2017): Scale & Speed

With the explosion of **big data**, online advertising moved to real-time environments. **Real-Time Bidding (RTB)** became standard — where ad impressions were auctioned in milliseconds. This required **real-time click prediction** models.

Companies used powerful algorithms like:

- Gradient Boosting Machines (GBM)
- XGBoost
- Online Learning Algorithms

Click prediction became more complex and accurate due to the availability of **massive datasets** and **faster computation**.

5. Deep Learning & Personalization (2017–Present)

In recent years, **deep learning models** like **Deep Neural Networks (DNNs)**, **Recurrent Neural Networks (RNNs)**, and **Embedding techniques** have been used to model user behavior more accurately.

Major platforms (Google, Facebook, TikTok) now use advanced deep learning systems trained on billions of interactions to predict user intent. These models learn from:

- User profiles
- Device usage patterns
- Time and location
- Historical ad interactions

These systems are **self-learning**, continuously improving with every user interaction.

Goals of Online Ad Click Prediction

1. Increase Click-Through Rate (CTR)

The main goal is to show ads to users who are most likely to **click** on them. By accurately predicting user interest, businesses can improve the **CTR**, which means more clicks per 1000 ad views.

CTR = (Number of Clicks / Number of Impressions) × 100

2. Maximize Return on Investment (ROI)

Online advertising costs money. Predicting which users are likely to click helps in **targeting only the right audience**, reducing waste of money and increasing profits.

Better targeting = Less spending + More conversions

3. Improve User Experience

When users see **relevant ads**, they are more likely to engage. Ad click prediction helps show ads that match user interests, leading to a better **user experience** without annoying, irrelevant ads.

4. Optimize Ad Campaigns

Advertisers use prediction models to analyze which ad campaigns are performing well. This helps in **budget allocation**, creative design improvements, and better ad placement strategies.

5. Real-Time Bidding (RTB) Support

In real-time advertising, ad spaces are sold in milliseconds. Accurate click prediction allows advertisers to **bid smartly** — only bidding high for impressions likely to convert.

6. Reduce Ad Fraud and Wastage

Click prediction models can detect **abnormal click patterns** or bots. This helps reduce **fraudulent clicks** and ensures that real users are targeted.

7. Personalized Advertising

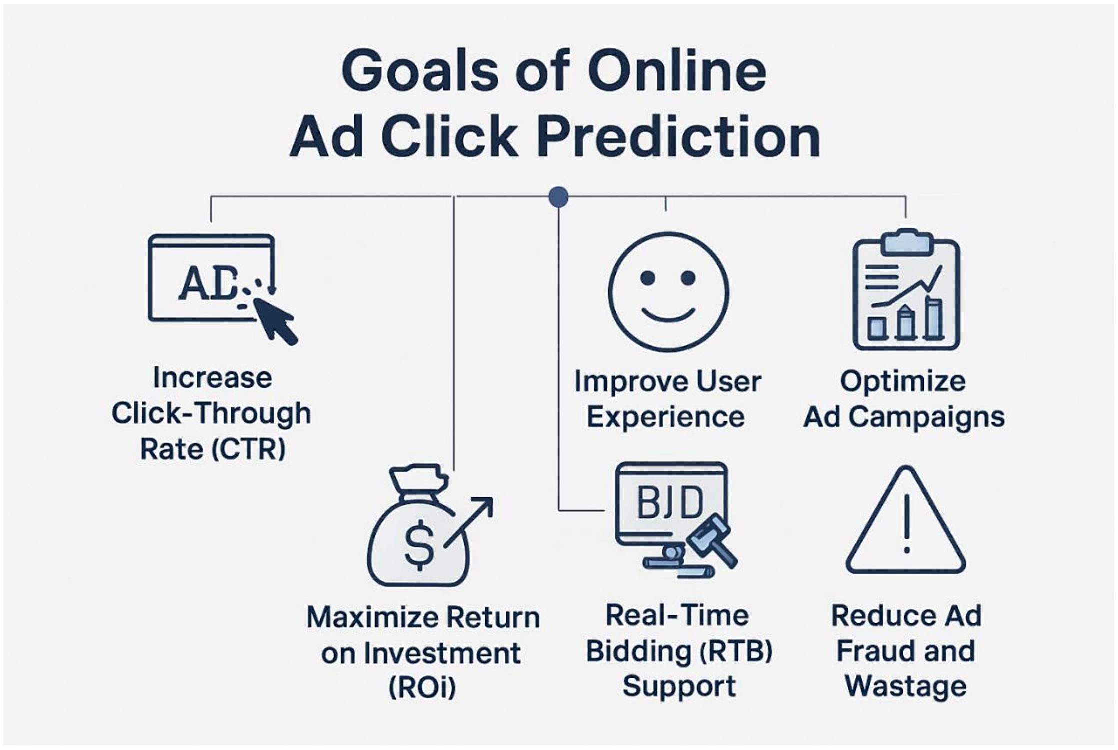
By understanding user behavior, models help in **personalizing** ads — making them more relevant to individual users based on past activity, location, device type, etc.

8. Data-Driven Decision Making

Click prediction provides **actionable insights** using machine learning and data analytics, allowing advertisers to take **data-driven** decisions rather than relying on guesswork.

Conclusion

The goal of Online Ad Click Prediction is not just to increase clicks, but to make **advertising smarter**, more **efficient**, and more **personalized** — benefiting both **advertisers** and **users** in the growing digital economy.


Goals of Online Ad Click Prediction
AD
Increase Click-Through Rate (CTR)
Improve User Experience
Optimize Ad Campaigns
$
Maximize Return on Investment (ROi)
BID
Real-Time Bidding (RTB) Support
Reduce Ad Fraud and Wastage

Application Online Ad Click Prediction

1. Targeted Advertising

By predicting which users are most likely to click on an ad, advertisers can **target specific audiences** more effectively. This ensures ads are shown to people who are actually interested, improving conversion rates.

2. Budget Optimization

Ad budgets are often limited. Click prediction models help advertisers spend their money wisely by **focusing only on high-performing ads or user segments**, reducing waste.

3. Real-Time Bidding (RTB) Systems

In RTB platforms like **Google Ads** or **Facebook Ads**, prediction models help in making **real-time decisions** about whether to bid for an ad slot and how much to bid — all in milliseconds.

4. Personalized User Experience

Using click prediction, platforms can serve **relevant and personalized ads** to users based on their behavior and preferences, which improves user satisfaction and engagement.

5. Recommendation Engines

Click prediction can also be used in **recommending products or services** that users are more likely to be interested in, based on their ad interaction patterns.

6. A/B Testing and Campaign Analysis

Prediction models help in analyzing **which ads perform best** under different conditions (e.g., time of day, region, age group), aiding in better **A/B testing** and performance tracking.

7. Fraud Detection

Advanced click prediction systems can detect **anomalous patterns**, such as bot clicks or fake engagements, helping platforms reduce **ad fraud**.

8. Dynamic Ad Placement

Online ad platforms use predictions to **dynamically decide** where, when, and how an ad should appear (e.g., mobile vs desktop, homepage vs inside pages) for maximum impact.

9. Performance-Based Billing

Platforms often charge advertisers per click (PPC). Accurate prediction helps ensure that the clicks billed are likely to result in meaningful interactions, improving **trust and transparency**.

10. Social Media Marketing

Platforms like **Instagram, Facebook, Twitter**, and **LinkedIn** use click prediction to serve sponsored posts or suggested ads in users’ feeds based on interaction history and interests.

Conclusion

Online Ad Click Prediction is a key technology that powers much of the **modern digital advertising ecosystem**. Its applications are seen everywhere — from e-commerce to social media, and from mobile apps to search engines — making advertising smarter, faster, and more relevant.

Machine Learning

Abstract Machine.

Learning has always been an integral part of artificial intelligence and its methodology has evolved in the concert with the major concerns in the field. In response of increasing of ever increasing of the volume of knowledge in modern AI systems, many researchers have turned their attention into machine learning as to overcome the knowledge acquisition bottleneck. Together with many other disciplines, machine learning methods have been widely employed in bioinformatics. The difficulties and cost of biological analyses have led to the development of sophisticated machine learning approaches for this application area. In this chapter, we first review the fundamental concepts of machine learning such as feature assessment, unsupervised versus supervised learning and types of classification. Then, we point out the main issues of designing machine learning experiments and their performance evaluation. Finally, we introduce some supervised learning methods.

Introduction to Machine Learning.

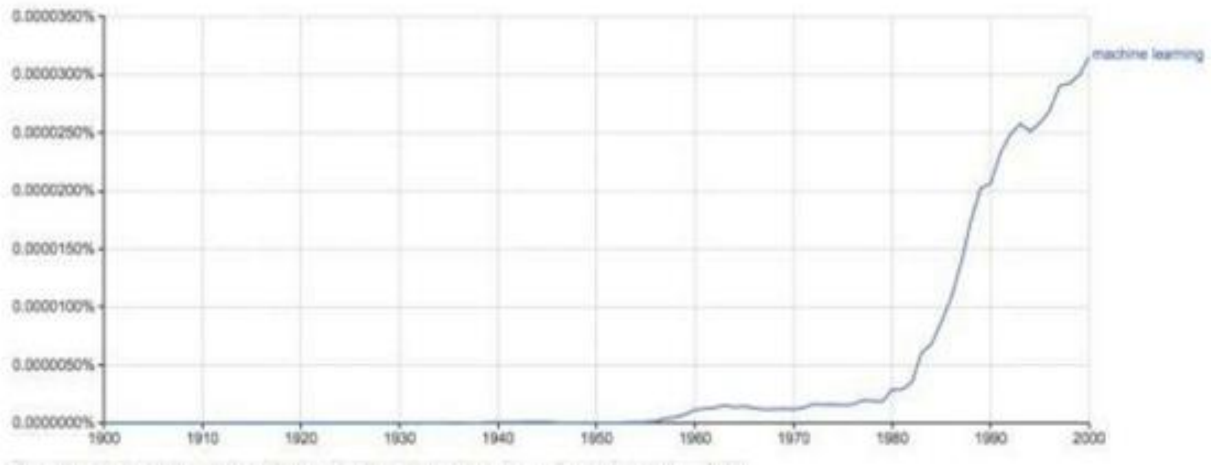
Machines have come a long way since the Industrial Revolution. They continue to fill factory floors and manufacturing plants, but now their capabilities extend beyond manual activities to cognitive tasks that, until recently, only humans were capable of performing. Judging song competitions, driving automobiles, and mopping the floor with professional chess players are three examples of the specific complex tasks machines are now capable of simulating. But their remarkable feats trigger fear among some observers. Part of this fear nestles on the neck of survivalist insecurities, where it provokes the deep-seated question of what if? What if intelligent machines turn on us in a struggle of the fittest? What if intelligent machines produce offspring with capabilities that humans never intended to impart to machines? What if the legend of the singularity is true? The other notable fear is the threat to job security, and if you're a truck driver or an accountant, there is a valid reason to be worried. According to the British Broadcasting Company's (BBC) interactive online resource Will a robot take my job?, professions such as bar worker (77%), waiter (90%), chartered accountant (95%), receptionist (96%), and taxi driver (57%) each have a high chance of becoming automated by the year 2035. [1] But research on planned job automation and crystal ball gazing with respect to the future evolution of machines and artificial intelligence (AI) should be read with a pinch of skepticism. AI technology is moving fast, but broad adoption is still an unchartered path fraught with known and unforeseen challenges. Delays and other obstacles are inevitable. Nor is machine learning a simple case of flicking a switch and asking the machine to predict the outcome of the Super Bowl and serve you a delicious martini. Machine learning is far from what you would call an out-of-the-box solution. Machines operate based on statistical algorithms managed and overseen by skilled individuals—known as data scientists and machine learning engineers. This is one labor market where job opportunities are destined for growth but where, currently, supply is struggling to meet demand. Industry experts lament that one of the biggest obstacles delaying the progress of AI is the inadequate supply of professionals with the necessary expertise and training.

According to Charles Green, the Director of Thought Leadership at Belatrix Software: “It’s a huge challenge to find data scientists, people with machine learning experience, or people with the skills to analyze and use the data, as well as those who can create the algorithms required for machine learning. Secondly, while the technology is still emerging, there are many ongoing developments. It’s clear that AI is a long way from how we might imagine it.”

To build and program intelligent machines, you must first understand classical statistics. Algorithms derived from classical statistics contribute the metaphorical blood cells and oxygen that power machine learning. Layer upon layer of linear regression, k-nearest neighbors, and random forests surge through the machine and drive their cognitive abilities. Classical statistics is at the heart of machine learning and many of these algorithms are based on the same statistical equations you studied in high school. Indeed, statistical algorithms were conducted on paper well before machines ever took on the title of artificial intelligence.

WHAT IS MACHINE LEARNING?

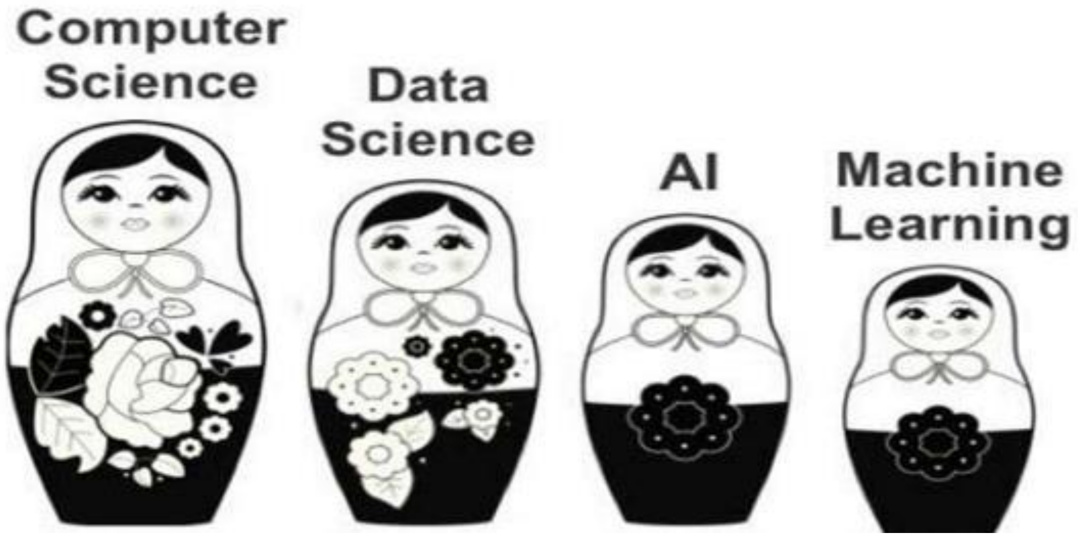
In 1959, IBM published a paper in the IBM Journal of Research and Development with an, at the time, obscure and curious title. Authored by IBM's Arthur Samuel, the paper invested the use of machine learning in the game of checkers "to verify the fact that a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program." [3] Although it was not the first publication to use the term "machine learning" per se, Arthur Samuel is widely considered as the first person to coin and define machine learning in the form we now know today. Samuel's landmark journal submission, Some Studies in Machine Learning Using the Game of Checkers, is also an early indication of homo sapiens' determination to impart our own system of learning to man-made machines.



Arthur Samuel introduces machine learning in his paper as a subfield of computer science that gives computers the ability to learn without being explicitly programmed. [4] Almost six decades later, this definition remains widely accepted. Although not directly mentioned in Arthur Samuel's definition, a key feature of machine learning is the concept of self-learning. This refers to the application of statistical modelling to detect patterns and improve performance based on data and empirical information; all without direct programming commands. This is what Arthur Samuel described as the ability to learn without being explicitly programmed. But he doesn't infer that machine formulate decisions with no upfront programming. On the contrary, machine learning is heavily dependent on computer programming. Instead, Samuel observed that machines don't require a direct input command to perform a set task but rather input data.

Training & Test Data

In machine learning, data is split into training data and test data. The first split of data, i.e. the initial reserve of data you use to develop your model, provides the training data. In the spam email detection example, false positives similar to the PayPal auto-response might be detected from the training data. New rules or modifications must then be added, e.g., email notifications issued from the sending address “payments@paypal.com” should be excluded from spam filtering. After you have successfully developed a model based on the training data and are satisfied with its accuracy, you can then test the model on the remaining data, known as the test data. Once you are satisfied with the results of both the training data and test data, the machine learning model is ready to filter incoming emails and generate decisions on how to categorize those incoming messages. The difference between machine learning and traditional programming may seem trivial at first, but it will become clear as you run through further examples and witness the special power of self-learning in more nuanced situations. The second important point to take away from this chapter is how machine learning fits into the broader landscape of data science and computer science. This means understanding how machine learning interrelates with parent fields and sister disciplines. This is important, as you will encounter these related terms when searching for relevant study materials—and you will hear them mentioned ad nauseam in introductory machine learning courses. Relevant disciplines can also be difficult to tell apart at first glance, such as “machine learning” and “data mining.” Let’s begin with a high-level introduction. Machine learning, data mining, computer programming, and most relevant fields (excluding classical statistics) derive first from computer science, which encompasses everything related to the design and use of computers. Within the all-encompassing space of computer science is the next broad field: data science. Narrower than computer science, data science comprises methods and systems to extract knowledge and insights from data through the use of computers.



ML Categories

Machine learning incorporates several hundred statistical-based algorithms and choosing the right algorithm or combination of algorithms for the job is a constant challenge for anyone working in this field. But before we examine specific algorithms, it is important to understand the three overarching categories of machine learning. These three categories are supervised, unsupervised, and reinforcement.

Supervised Learning

As the first branch of machine learning, supervised learning concentrates on learning patterns through connecting the relationship between variables and known outcomes and working with labeled datasets. Supervised learning works by feeding the machine sample data with various features (represented as “X”) and the correct value output of the data (represented as “y”). The fact that the output and feature values are known qualifies the dataset as “labeled.” The algorithm then deciphers patterns that exist in the data and creates a model that can reproduce the same underlying rules with new data. For instance, to predict the market rate for the purchase of a used car, a supervised algorithm can formulate predictions by analyzing the relationship between car attributes (including the year of make, car brand, mileage, etc.) and the selling price of other cars sold based on historical data. Given that the supervised algorithm knows the final price of other cards sold, it can then work backward to determine the relationship between the characteristics of the car and it

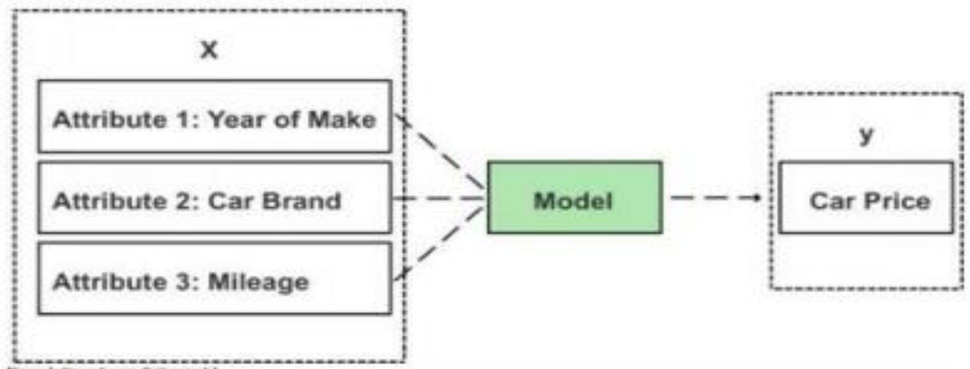


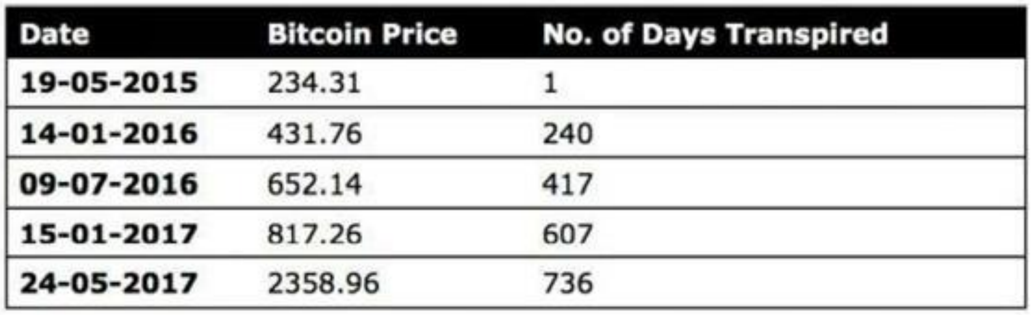
Figure 1: Car value prediction model

s value

After the machine deciphers the rules and patterns of the data, it creates what is known as a model: an algorithmic equation for producing an outcome with new data based on the rules derived from the training data. Once the model is prepared, it can be applied to new data and tested for accuracy. After the model has passed both the training and test data stages, it is ready to be applied and used in the real world. In Chapter 13, we will create a model for predicting house values where y is the actual house price and X are the variables that impact y, such as land size, location, and the number of rooms. Through supervised learning, we will create a rule to predict y (house value) based on the given values of various variables (X). Examples of supervised learning algorithms include regression analysis, decision trees, k nearest neighbours, neural networks, and support vector machines. Examples of supervised learning algorithms include regression analysis, decision trees, k nearest neighbours, neural networks, and support vector machines.

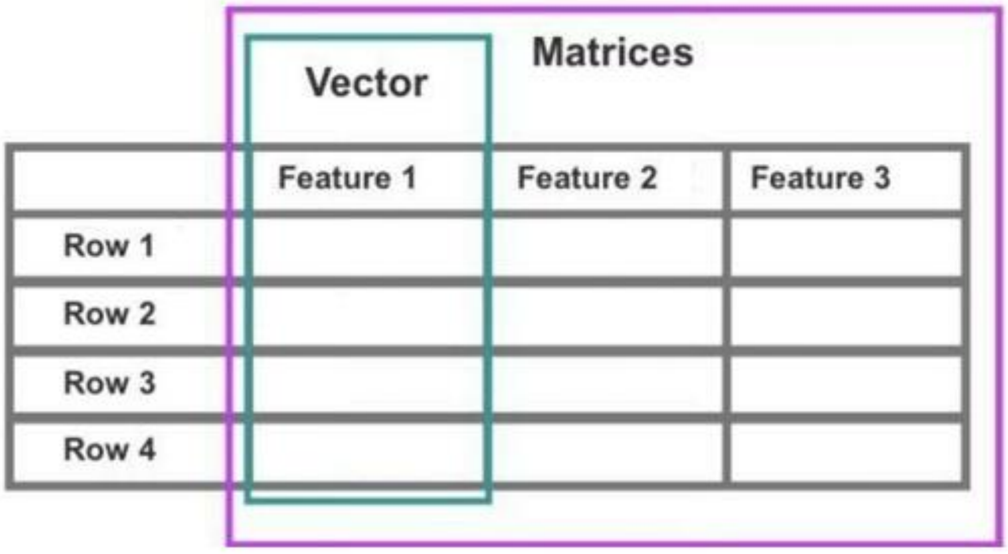
The Machine Learning Tool Box.

A handy way to learn a new subject area is to map and visualize the essential materials and tools inside a toolbox. If you were packing a toolbox to build websites, for example, you would first pack a selection of programming languages. This would include frontend languages such as HTML, CSS, and JavaScript, one or two backend programming languages based on personal preferences, and of course, a text editor. You might throw in a website builder such as WordPress and then have another compartment filled with web hosting, DNS, and maybe a few domain names that you've recently purchased. This is not an extensive inventory, but from this general list, you can start to gain a better appreciation of what tools you need to master in order to become a successful website developer. Let's now unpack the toolbox for machine learning. Compartment 1: Data In the first compartment is your data. Data cons tutes the input variables needed to form a predic on. Data comes in many forms, including structured and non-structured data. As a beginner, it is recommended that you start with structured data. This means that the data is defined and labeled (with schema) in a table, as shown here:

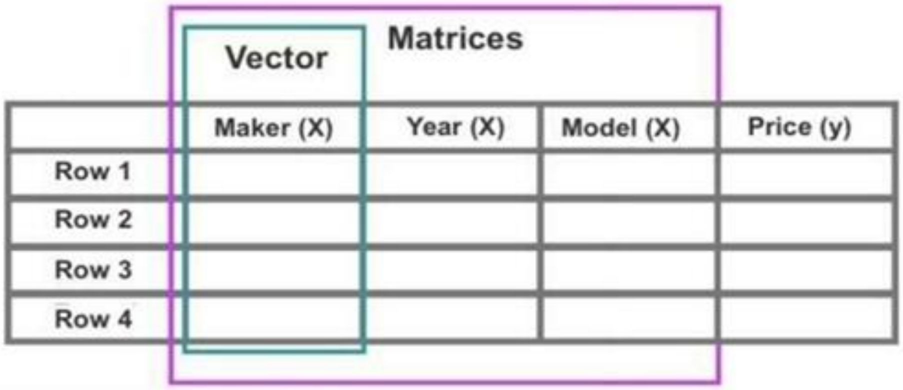


Date	Bitcoin Price	No. of Days Transpired
19-05-2015	234.31	1
14-01-2016	431.76	240
09-07-2016	652.14	417
15-01-2017	817.26	607
24-05-2017	2358.96	736

A tabular (table-based) dataset contains data organized in rows and columns. In each column is a feature. A feature is also known as a variable, a dimension or an attribute—but they all mean the same thing. Each individual row represents a single observation of a given feature/variable. Rows are sometimes referred to as a case or value, but in this book, we will use the term "row."

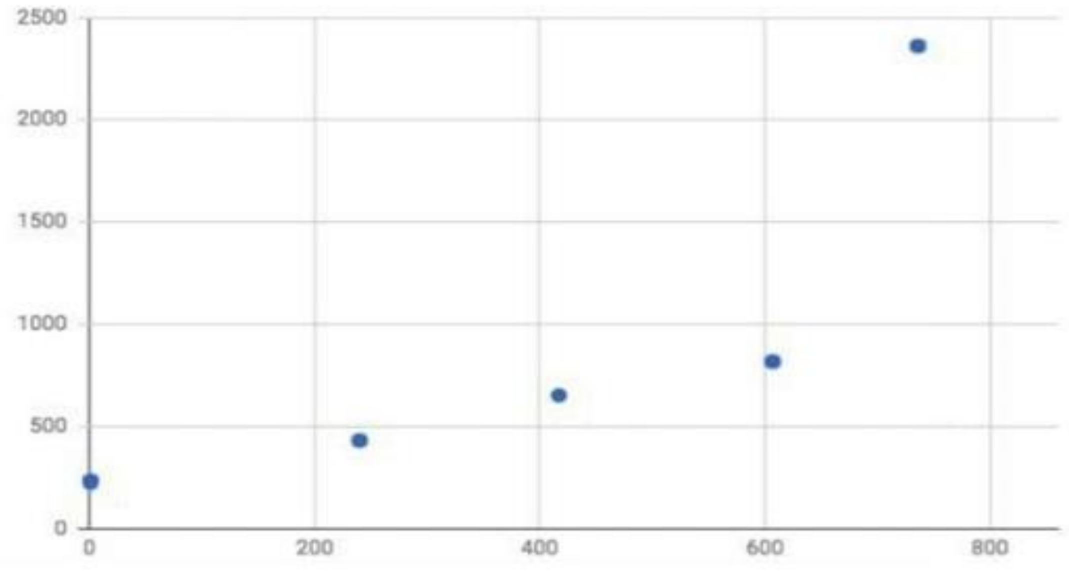


Each column is known as a vector. Vectors store your X and y values and mul ple vectors (columns) are commonly referred to as matrices. In the case of supervised learning, y will already exist in your dataset and be used to iden fy pa erns in rela on to independent variables (X). The y values are commonly expressed in the final column, as shown in Figure 2



	Maker (X)	Year (X)	Model (X)	Price (y)
Row 1				
Row 2				
Row 3				
Row 4				

Next, within the first compartment of the toolbox is a range of sca erplots, including 2-D, 3 D, and 4-D plots. A 2-D sca erplot consists of a ver cal axis (known as the y-axis) and a horizontal axis (known as the x-axis) and provides the graphical canvas to plot a series of dots, known as data points. Each data point on the sca erplot represents one observa on from the dataset, with X values plo ed on the x-axis and y values plo ed on the y-axis.



	Independent Variable (X)	Dependent Variable (y)
Row 1	1	243.31
Row 2	240	431.76
Row 3	417	653.14
Row 4	607	817.26
Row 5	736	2358.96

Logistic Online Ad Click Prediction.

What is It?

Logistic Online Ad Click Prediction refers to using **Logistic Regression**, a statistical machine learning algorithm, to **predict whether a user will click on an online ad (1)** or **not click (0)**.

It is one of the **most widely used and simple models** for binary classification problems like this.

Why Logistic Regression?

Logistic Regression is useful because:

- It gives the **probability** of a click (between 0 and 1).
- It is easy to implement and interpret.
- It works well with **structured data** (like user age, device type, ad time, etc.)
- It is fast and scalable for **large datasets**.

How It Works (Simple Example)

Let's say you have these input features:

Feature	Description
Age	Age of the user
Time_Of_Day	Morning / Evening / Night
Device_Type	Mobile / Desktop
Ad_Category	Tech / Fashion / Food

These features are fed into the **Logistic Regression model**, which calculates:

$$\mathbf{P(click) = 1 / (1 + e\hat{}-(w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n))}$$

Where:

- $x_1, x_2, ...$ = input features
- $w_1, w_2, ...$ = learned weights
- `P(click)` = predicted probability of a click

If **P(click) > 0.5**, it predicts **1** (user will click).
If **P(click) < 0.5**, it predicts **0** (user will not click).

Example Output:

User	**P(Click)**	**Prediction**
User A	0.84	Click (1)
User B	0.35	No Click (0)

Benefits:

- Easy to train and deploy
- Fast prediction time
- Works well with **linear relationships**
- Good baseline model in ad click prediction tasks

Use in Real World:

Logistic regression is used by advertising platforms like:

- Google Ads
- Facebook Ads
- Amazon Ads

It often serves as a **baseline model**, and more complex models (e.g., Random Forest, XGBoost, Deep Learning) are built upon it.

Decision Online Ad Click Prediction.

What is It?

Decision Online Ad Click Prediction refers to using a **Decision Tree algorithm** to predict whether a user will **click** on an online advertisement (**Click = 1**) or **not click** (**Click = 0**).

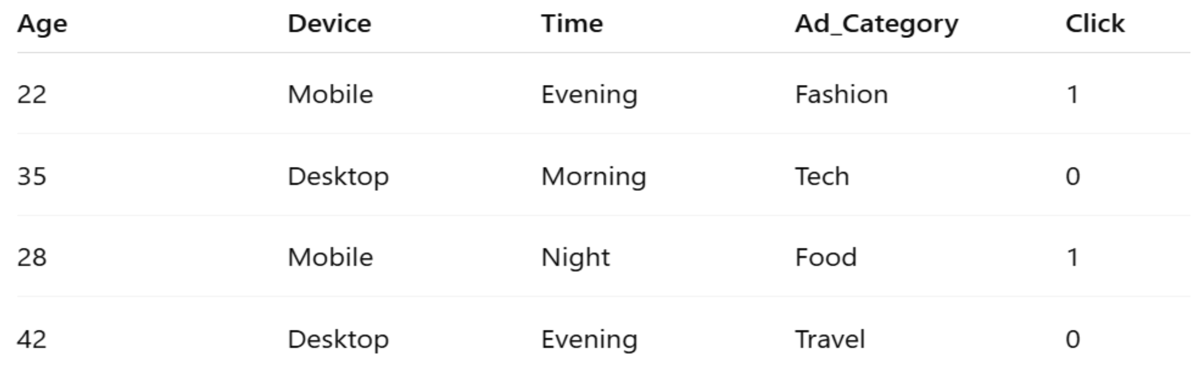
A **Decision Tree** is a machine learning model that makes predictions by **splitting data** based on conditions in a tree-like structure.

How Does a Decision Tree Work?

A Decision Tree works like a series of **if-else questions**.

Example:

IF Age < 25 AND Device = Mobile THEN Click = 1
ELSE Click = 0
It keeps dividing the dataset into smaller parts using the **best features**, until it can make a final prediction (Click or No Click).



Age	Device	Time	Ad_Category	Click
22	Mobile	Evening	Fashion	1
35	Desktop	Morning	Tech	0
28	Mobile	Night	Food	1
42	Desktop	Evening	Travel	0

The Decision Tree will analyze this data and create rules like:

- If **Device = Mobile** and **Time = Night**, then likely **Click = 1**
- If **Age > 40** and **Device = Desktop**, then likely **Click = 0**

Why Use Decision Trees for Ad Click Prediction?

- Easy to **visualize** and understand
- Works with **categorical and numerical** data
- Can handle **non-linear patterns**
- No need for feature scaling
- Good for **initial models or small datasets**

Limitations:

- Can **overfit** on small datasets (memorize instead of generalize)
- Less accurate than ensemble methods like **Random Forest** or **XGBoost**
- Sensitive to noisy data

Example Output:

If a user:

- Is **Age = 21**
- Uses a **Mobile**
- Is browsing during **Evening**
- Ad type is **Fashion**

The Decision Tree might output:

Click = 1 (High chance user will click the ad)

Real-World Applications:

- Used in platforms like **Facebook Ads**, **Google Ads**, and **YouTube** to:
    - Predict ad clicks
    - Choose the best ad to show
    - Personalize the user experience

Conclusion:

Decision Online Ad Click Prediction is an intuitive and effective way to classify user behavior. It gives interpretable rules for businesses and serves as a **foundation model** in digital marketing analytics.

KNN Algorithm Online Ad Click Prediction.

What is KNN?

K-Nearest Neighbors (KNN) is a **supervised machine learning algorithm** used for **classification and regression**. In **Online Ad Click Prediction**, it is used to classify whether a user will **click** (1) or **not click** (0) on an advertisement.

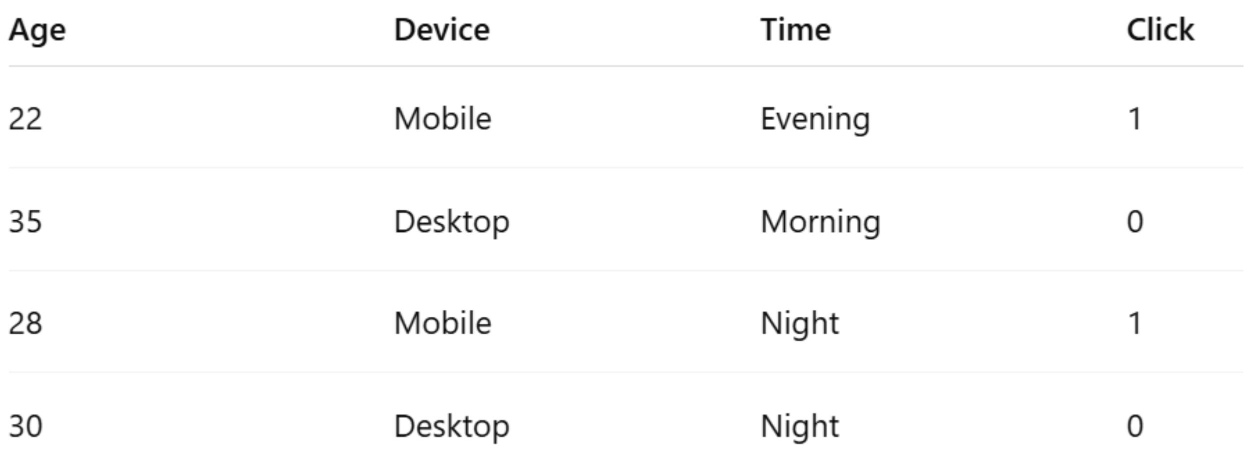
KNN works on a simple idea:

"A user is likely to behave like their nearest neighbors (similar users)."

How Does KNN Work in Ad Click Prediction?

1. **Step 1: Input Features**
   - Age, Gender, Device Type, Time of Day, Ad Type, etc.
2. **Step 2: Choose K**
   - Decide how many neighbors (K) to look at — common values: 3, 5, 7
3. **Step 3: Measure Distance**
   - Calculate distance (e.g., Euclidean distance) between the new user and other users in the dataset.
4. **Step 4: Find Nearest Neighbors**
   - Select the **K users** most similar to the new user.
5. **Step 5: Predict Class**
   - If most of the K neighbors **clicked**, predict **Click = 1**
   - If most **did not click**, predict **Click = 0**

Example Dataset:



Age	Device	Time	Click
22	Mobile	Evening	1
35	Desktop	Morning	0
28	Mobile	Night	1
30	Desktop	Night	0

If a new user is:

- Age: 29
- Device: Mobile
- Time: Night

KNN will:

- Find the 3 nearest users (e.g., users 1, 3, and 4)
- Since majority (2 out of 3) **clicked**, predict: **Click = 1**

Advantages of KNN:

- Simple and intuitive
- No need to train the model
- Works well with small to medium-sized datasets
- Captures non-linear patterns

Disadvantages:

- Slow with large datasets
- Requires good **feature scaling** (e.g., normalization)
- Sensitive to irrelevant features or noise

Real-World Use Case:

In online advertising:

- Platforms can use KNN to recommend ads to users who are **similar** to those who clicked on a certain type of ad.
- Especially useful when you want **quick predictions** without complex models.

Conclusion:

KNN Algorithm offers a simple yet powerful way to perform **Online Ad Click Prediction** based on user similarity. Though not ideal for large-scale systems, it is an excellent starting point for **understanding user behavior** in digital marketing.

K means Algorithm Online Ad Click Prediction.

What is K-Means?

K-Means is an **unsupervised machine learning algorithm** used for **clustering** data into groups based on similarity.

In the context of **Online Ad Click Prediction**, K-Means is not used to directly predict clicks (since it doesn't use labeled data), but it helps to **discover patterns or group similar users** — which can **indirectly improve ad targeting**.

Use Case in Online Advertising:

K-Means can be used to:

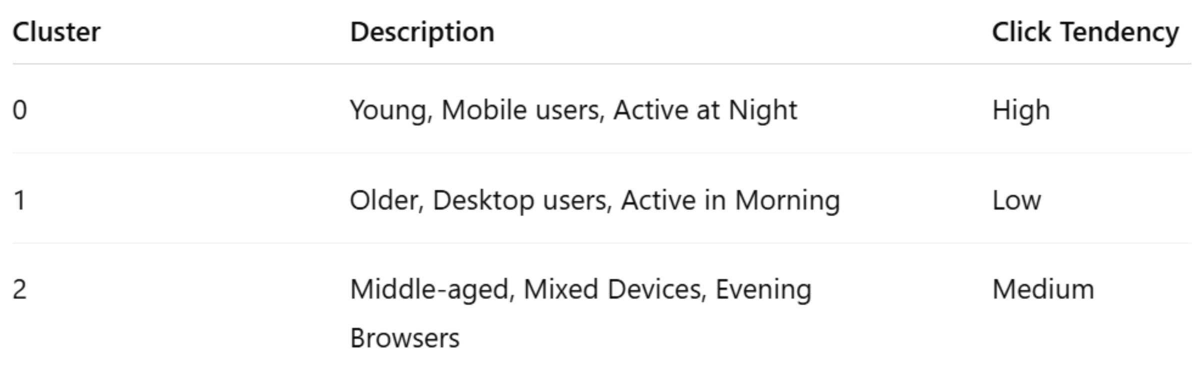
- Segment users into **clusters** based on behavior (e.g., time, age, location, device type, etc.)
- Identify clusters of users that are more likely to **click on ads**
- Personalize ad campaigns for each **user group**

Example:

Let’s say we have user data with:

- Age
- Time of activity
- Device type
- Number of previous ad clicks

K-Means may create clusters like:



Cluster	Description	Click Tendency
0	Young, Mobile users, Active at Night	High
1	Older, Desktop users, Active in Morning	Low
2	Middle-aged, Mixed Devices, Evening Browsers	Medium

Now, marketers can:

- Target **Cluster 0** users more aggressively

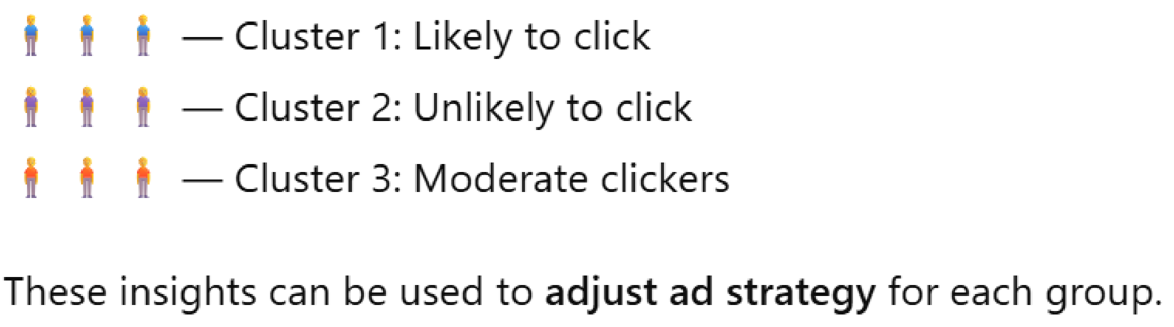
- Reduce spend on **Cluster 1**
- Customize ads based on **cluster behavior**

How K-Means Works (Simplified):

1. **Choose K** = Number of clusters (e.g., K = 3)
2. **Initialize K centroids randomly**
3. **Assign each user** to the nearest centroid (based on similarity)
4. **Recalculate centroids** by averaging the data in each cluster
5. **Repeat** steps 3-4 until the clusters stop changing

Visual Example:

Imagine plotting users on a graph based on their age and number of past clicks. K-Means will group users into **clusters** like this:



🧍 🧍 🧍 — Cluster 1: Likely to click

🧍 🧍 🧍 — Cluster 2: Unlikely to click

🧍 🧍 🧍 — Cluster 3: Moderate clickers

These insights can be used to **adjust ad strategy** for each group.

Advantages:

- Easy to understand and implement
- Finds **natural user segments**
- Helps in **personalized marketing**

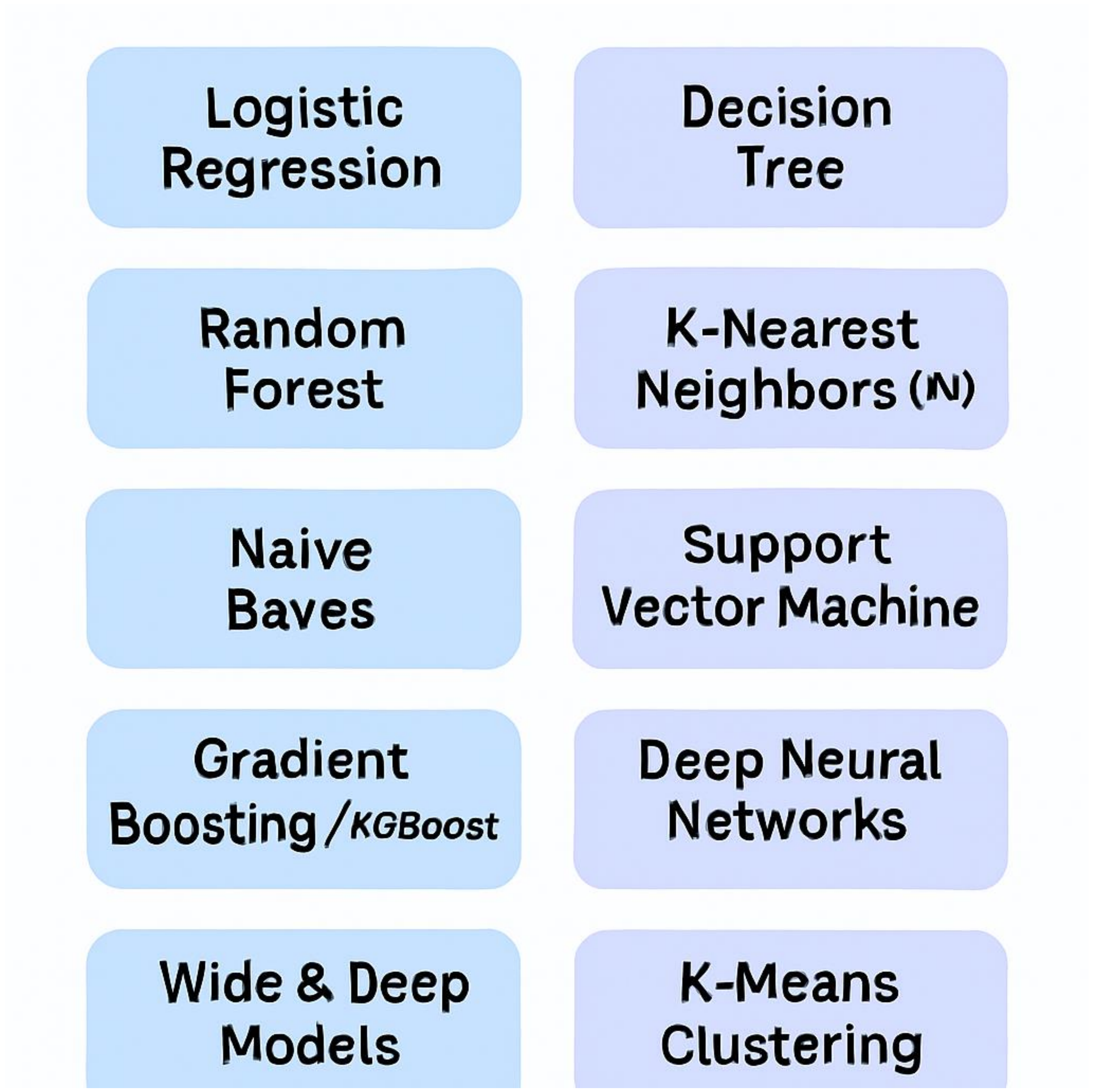
Limitations:

- Doesn’t directly predict click = 1 or 0
- You have to **choose K manually**
- Results depend on **initial centroid positions**
- Not suitable for very large or high-dimensional data without optimization

Conclusion:

K-Means Algorithm is a powerful tool in Online Ad Click Prediction pipelines — not for direct classification, but for **clustering users based on behavior**. These clusters help in creating **targeted, cost-efficient, and relevant** ad campaigns.

Algorithms of Online Ad Click Prediction.



1. Logistic Regression

- **Type:** Supervised, Linear Model
- **Usage:** Predicts probability of click
- **Advantages:** Simple, fast, interpretable
- **Limitation:** Assumes linear relationship

2. Decision Tree

- **Type:** Supervised, Tree-based
- **Usage:** Splits data using rules (e.g., age < 30 → click)
- **Advantages:** Easy to visualize
- **Limitation:** Can overfit on small datasets

3. Random Forest

- **Type:** Supervised, Ensemble (many decision trees)
- **Usage:** Combines predictions from multiple trees
- **Advantages:** Higher accuracy than a single tree
- **Limitation:** Slower than simple models

4. K-Nearest Neighbors (KNN)

- **Type:** Supervised, Instance-based
- **Usage:** Predicts based on behavior of similar users
- **Advantages:** Simple and effective for small data
- **Limitation:** Slow with large datasets

5. Naive Bayes

- **Type:** Supervised, Probabilistic
- **Usage:** Based on Bayes' theorem with independence assumption
- **Advantages:** Fast, works well with categorical data
- **Limitation:** Assumes all features are independent

6. Support Vector Machine (SVM)

- **Type:** Supervised, Classification
- **Usage:** Finds best boundary between click and no-click
- **Advantages:** Works well with high-dimensional data
- **Limitation:** Not ideal for very large datasets

7. Gradient Boosting / XGBoost

- **Type:** Supervised, Ensemble
- **Usage:** Builds multiple trees sequentially to improve accuracy
- **Advantages:** High accuracy, used in competitions
- **Limitation:** More complex and slower to train

8. Deep Neural Networks (DNNs)

- **Type:** Supervised, Deep Learning
- **Usage:** Learns complex patterns in user behavior
- **Advantages:** Great for large-scale systems (e.g., Google Ads)
- **Limitation:** Requires large data and computational power

9. Wide & Deep Models

- **Type:** Hybrid (Linear + Deep Learning)
- **Usage:** Combines memorization (wide) and generalization (deep)
- **Used By:** Google, Facebook for click prediction

10. K-Means Clustering *(for Preprocessing or Segmentation)*

- **Type:** Unsupervised
- **Usage:** Groups users into clusters (e.g., high/low clickers)
- **Advantages:** Improves personalization
- **Limitation:** Not a direct prediction model

Choosing the Right Algorithm Depends On:

Factor	**Suitable Algorithms**
Small & clean data	Logistic, KNN, Naive Bayes
Medium-sized data	Random Forest, SVM
Large-scale data	XGBoost, Deep Learning
Need for explainability	Decision Trees, Logistic Regression
Need for personalization	K-Means (clustering) + classifier combo

Conclusion:

Online Ad Click Prediction combines **statistics, machine learning, and deep learning** to accurately target ads. The choice of algorithm depends on the **data**, **performance needs**, and **business goals**. Often, **multiple algorithms are compared**, and the best one is selected based on accuracy, precision, recall, and AUC score.

CONCLUSION Online Ad Click Prediction

Online Ad Click Prediction is a powerful application of machine learning and data science in the digital advertising industry. It enables advertisers to predict whether a user will click on an advertisement based on various behavioral and contextual factors such as age, gender, device type, time of day, browsing history, and more.

The rise of digital platforms like Google, Facebook, and Instagram has created a massive demand for accurate targeting of ads. Traditional methods of advertisement, which were based on assumptions and mass broadcasting, are no longer sufficient in today's data-driven world. Online Ad Click Prediction models use historical data and modern algorithms to make intelligent predictions that help companies **optimize their marketing strategies**, **reduce costs**, and **increase user engagement**.

During this project, we explored the complete workflow for building a predictive system:

1. **Dataset Collection** from sources like Kaggle
2. **Data Preprocessing** to clean, transform, and prepare the data
3. **Model Building** using ML algorithms like Logistic Regression, Decision Tree, KNN, Random Forest, and K-Means for clustering
4. **Performance Evaluation** using metrics such as Accuracy, Precision, Recall, and F1 Score
5. **User Interface** using tools like Streamlit or dashboards to present results

Among the tested models, advanced algorithms like Random Forest, Gradient Boosting, and Neural Networks performed well on large datasets due to their ability to capture complex patterns. However, simpler models like Logistic Regression still provide good results with smaller or linear datasets and are easier to interpret.

This project also highlighted the importance of **feature selection**, **data balancing (handling clicks vs no-clicks)**, and **real-time prediction**. Real-world ad datasets often suffer from imbalance (very few actual clicks), and models need to be carefully trained to avoid bias.

From a business perspective, click prediction can significantly:

- **Improve ad relevance** by showing users what they are likely to engage with
- **Reduce budget waste** by eliminating low-performing ads
- **Increase Return on Investment (ROI)** by focusing on high-performing segments

In the future, more advanced deep learning models such as **Wide & Deep Networks**, **Reinforcement Learning**, and **Real-time AI systems** will further improve accuracy and personalization. With privacy and user data regulations becoming stricter (e.g., GDPR), ethical use of data in training such models is also an important consideration.

Final Thought:

Online Ad Click Prediction is not just a technical challenge — it’s a key part of modern digital business. By combining data science, machine learning, and ethical practices, companies can create more relevant and successful advertising campaigns that benefit both users and businesses.

Code

```
  import pandas as pd
```

```
data = {
    'Daily Time Spent on Site': [68.95, 80.23, 69.47, 74.15, 68.37],
    'Age': [35, 31, 26, 29, 35],
    'Area Income': [61833.90, 68441.85, 59785.94, 54806.18, 73889.99],
    'Daily Internet Usage': [256.09, 193.77, 236.50, 245.89, 225.58],
    'Male': [0, 1, 0, 1, 0],
    'Clicked on Ad': [0, 0, 1, 0, 1]
}

df = pd.DataFrame(data)
print("□ Dataset Preview:")
print(df.head())
```

```
□ Dataset Preview:
   Daily Time Spent on Site  Age  Area Income  Daily Internet Usage  Male  \
0                     68.95   35     61833.90                256.09     0
1                     80.23   31     68441.85                193.77     1
2                     69.47   26     59785.94                236.50     0
3                     74.15   29     54806.18                245.89     1
4                     68.37   35     73889.99                225.58     0

   Clicked on Ad
0              0
1              0
2              1
3              0
4              1
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

X = df.drop("Clicked on Ad", axis=1)
y = df["Clicked on Ad"]
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y, test_size=0.2,
random_state=42)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
```

```
lr = LogisticRegression()
rf = RandomForestClassifier()

lr.fit(X_train, y_train)
rf.fit(X_train, y_train)
```

RandomForestClassifier
?i

```
RandomForestClassifier()
```

```
y_pred_lr = lr.predict(X_test)
print("\n Logistic Regression:")
print("Accuracy:", accuracy_score(y_test, y_pred_lr))
print(classification_report(y_test, y_pred_lr))
```

```
 Logistic Regression:
Accuracy: 0.0
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       1.0
           1       0.00      0.00      0.00       0.0

    accuracy                           0.00       1.0
   macro avg       0.00      0.00      0.00       1.0
weighted avg       0.00      0.00      0.00       1.0

/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
```

```
Recall is ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
Recall is ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-
packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
Recall is ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
y_pred_rf = rf.predict(X_test)
print("\n□ Random Forest:")
print("Accuracy:", accuracy_score(y_test, y_pred_rf))
print(classification_report(y_test, y_pred_rf))
```

```
□ Random Forest:
Accuracy: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1

    accuracy                           1.00         1
   macro avg       1.00      1.00      1.00         1
weighted avg       1.00      1.00      1.00         1
```

```
import joblib
joblib.dump(rf, "model.pkl")
joblib.dump(scaler, "scaler.pkl")
```

```
['scaler.pkl']
```