**PROFESSIONAL TRAINING REPORT**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

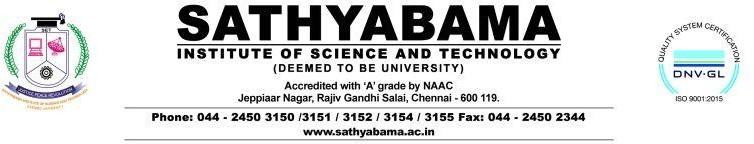
**(DEEMED UNIVERSITY)**

Submitted in partial fulfilment of the requirements for the award,

Bachelor of Engineering Degree in Computer Science and Engineering.

Navamshu Ram PV

(39110683)



**SCHOOL OF COMPUTING**

**BONAFIDE CERTIFICATE**

This is to certify that this Professional Training Report is the Bonafede work of **NAVAMSHU RAM PV** (REG NO.39110683) who underwent the professional training in “MACHINE LEARNING” under our supervision from SEPTEMBER 2021 to OCTOBER 2021.

**Internal Guide**

**Ms.S.R SRIVIDHYA**

**Head Of The Department**

**Dr.S. VIGNESHWARI, M.E., Ph.D.**

**Dr. LAKSHMAN L, M.E, Ph.D.**

**Submitted for Viva voice Examination held on**

**Internal Examiner External Examiner**

**DECLARATION**

I, **NAVAMSHU RAM PV (Reg. No 39110683)** hereby declare that the Professional Training Report on “ MACHINE LEARNING ” done by me under the guidance **Ms.S.R SRIVIDHYA** at Satyabhama University is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

**Date:**

**Place: Chennai Signature of the student**

**ACKNOWLEDGEMENT**

The satisfaction and elation that accompany the successful completion of any task would be incomplete without the mention of the people who have made it possible. It is our great privilege to express our gratitude and respect to all those who have guided me during the course of my Professional Training.

I convey my thanks to **Dr LAKSHMAN L,** Head of the Department, Department of Computer Science and Engineering for providing necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide  **MS SR SRIVIDHYA,** for her valuable guidance, suggestions and constant encouragement which paved my pay to successful completion of my project.

I convey my thanks to the management and all the teaching staffs and non-teaching staffs of the Computer Science and Engineering department who were helpful in many ways for the completion of the project.

Also, I thank the Almighty and my Parents for supporting me in the completion of the Professional Training.

**TRANING CERTIFICATE**



**Abstract**

# IN-VEHICLE-COUPON-RECOMMENDATION

*NAVAMSHU RAM PV*

In the past years, research in the fields of big data analysis, machine learning and decision tree techniques is getting more frequent.

This thesis describes, decision tree is a popular application of supervised learning. Using decision tree, recommend coupon for customers to potential user base.

This data was collected via a survey on Amazon Mechanical Turk. The survey describes different driving scenarios including the destination, current time, weather, passenger, etc., and then ask the person whether he will accept the coupon if he is the driver.

The decision tree method discussed in this paper is based on the classification dataset. We undergo three different relationships between user and product. User-product relationship occurs when some users have an affinity or preference towards specific products that they need. Product-product relationships occur when items are similar in nature, either by appearance or description. User-user relationships occur when some customers have similar taste with respect to a particular product or service.

table of contents

[1.Introduction 7](#_Toc87551666)

[1.1 Background 7](#_Toc87551667)

[1.2 Project Description 8](#_Toc87551668)

[1.3 Questions 9](#_Toc87551669)

[1.4 Report Outline 9](#_Toc87551670)

[2.Aim & Scope 10](#_Toc87551671)

[2.1 Recommender System 10](#_Toc87551672)

[3.Methods & Material 12](#_Toc87551673)

[3.1 Test/Train Split Algorithm 12](#_Toc87551674)

[3.2 RS-Adapted Decision Tree 13](#_Toc87551675)

[3.3 Data 13](#_Toc87551676)

[3.4 Attribute Selection Measures 14](#_Toc87551677)

[3.5 Accuracy & Precision 14](#_Toc87551678)

[4.Result & Discussion 15](#_Toc87551679)

[4.1 Data Pre-processing: 16](#_Toc87551680)

[4.2 Data Analysis 16](#_Toc87551681)

[5.Implementation 19](#_Toc87551682)

[5.1 Decision Tree Classifier 19](#_Toc87551683)

[5.2 System Architecture / Ideation Map 20](#_Toc87551684)

[5.3 Mathematical operations 21](#_Toc87551685)

[6. Source Code 23](#_Toc87551686)

[7. References 40](#_Toc87551687)

**1 Introduction**

* 1. **Background**

In this data science python project, we will perform one of the most essential applications of machine learning – decision tree. In this project, we will implement decision tree in python. Whenever you need to predict the best output for user, decision tree is the ideal methodology.

Decision Tree is the most important support tool with a tree-like structure that models probable outcomes, cost of resources, utilities and possible consequences of supervised learning.

Decision trees are one of the best forms of learning algorithms based on various learning methods. They boost predictive models with accuracy, ease in interpretation, and stability. The tools are also effective in fitting non-linear relationships since they can solve data-fitting challenges, such as regression and classifications. Major applications like assessing prospective growth opportunities, using demographic data to predict prospective clients, serving as a support tool in several fields.

* 1. **Project Description**

The main goal of this project was to recommend and deliver a component that can be used by the user effectively. Given behavioural data of the users from the recommender database, the component should produce a set of users groups based on user similarity. Using these groups, other desired aspect was to recommend customers to target the potential user base.

This project was practical in its nature, but it was based on a comprehensive research study on recommender system, decision tree, test/train algorithms and split classifier method.

The project was split into three phases:

* Study of the research field to identify common algorithms and methods used to solve this kind of problem.
* Implementation and comparison of the chosen algorithms integrated with the provided dataset.
* Evaluation of the trained algorithm and integration with the recommender system for decision tree making.

Recommender Systems (RS) propose useful and interesting items to users to in-crease both seller’s profit and buyer’s satisfaction. They contribute to the commercial success of many on-line ventures such as Amazon.com or Netflix and are a very active research area. Examples of recommended items include movies, web pages, books, news items and more. Often a RS attempts to predict the rating a user will give to items based on her past ratings and the ratings of other (similar) users.

Decision Trees have been previously used as a model-based approach for recommender systems. The use of decision trees for building recommendation models offers several benefits, such as: efficiency and interpretability and flexibility in handling a variety of input data types (ratings, demographic, contextual )

The decision tree forms a predictive model which maps the input to a predicted value based on the input’s attributes. Each interior node in the tree corresponds to an attribute and each arc from a parent to a child node represents a possible value or a set of values of that attribute. The construction of the tree begins with a root node and the input set. An attribute is assigned to the root and arcs and sub-nodes for each set of values are created. The input set is then split by the values so that each child node receives only the part of the input set which matches the attribute value as specified by the arc to the child node.

The process then repeats itself recursively for each child until splitting is no longer feasible. Either a single classification (predicted value) can be applied to each element in the divided set, or some other threshold is reached.

* 1. **Questions**

This kind of problem poses unique challenges that needed to be answered prior to the project. Firstly, the data is always changing since items in the system are removed whenever they are grouped. Therefore, grouping people using popular items within the group as a similarity measure is not possible. Rather, the way a user behaves in the application needs to be the focus of the decision algorithm.

The main questions that arise from this are:

* What is a reasonable accuracy and precision?
* How can the created tree be used in the application to target users with specialized content?
* How can the data be pre-processed to correctly split users into groups of similar people?
* What all other supervised learning that can be applied in solving this problem?

These questions will be answered thoroughly in this report.

**1.4 Report Outline**

1. Introduces the project’s research area and the project’s goals.
2. Presents some research papers that are related to this project.
3. Gives a thorough background about decision tree and test/train methods. In this section, a set of chosen methods are chosen and presented.
4. Describes the recommender system more in depth.
5. Gives an evaluation of the chosen method.
6. The system implementation is described.
7. The results are discussed, and some future work is presented.

**2** **Aim & Scope**

In the field of supervised learning, it is important to choose an algorithm well suited for the available dataset. To make this choice, projects that tackle similar problems and use similar techniques were analysed during the first step of the project. In this section, systems using different algorithms will be presented. The survey describes different driving scenarios including the destination, current time, weather, passenger, etc., and then ask the person whether he will accept the coupon if he is the driver.

**2.1 Recommender System**

In 2015, a master’s student from Uppsala University created a recommender system for using user-based collaborative filtering. This method is largely used in the recommender system field used in renowned systems such as YouTube and Amazon. States that due to the unstable nature of the item data in the system, it is more beneficial to use users to create recommendations rather than using items. This type of recommender system is similar to the decision-making problem that this project is centred around. Recommending items to a user with user-based collaborative filtering is a two step- process where each user is first given a neighbourhood of similar users. Secondly, the data in the computed neighbourhood is used to find a set of popular items within the neighbourhood based on the behaviour of the users in the neighbourhood. In order to compare the performance of the trained algorithm made by test/train algorithm with other dataset solutions, the first part of the user-based collaborative filtering algorithm that can be used.

Decision trees can be used for different recommender systems approaches:

• Collaborative Filtering - Breese et al. [BHK98] used decision trees for building a collaborative filtering system. Each instance in the training set refers to a single customer. The training set attributes refer to the feedback provided by the customer for each item in the system. In this case a dedicated decision tree is built for each item. For this purpose, the feedback provided for the targeted item (for instance like/dislike) is considered to be the decision that is needed to be predicted, while the feedback provided for all other items is used as the input attributes (decision nodes). Figure 1 (left) illustrates an example of such a tree, for movies.

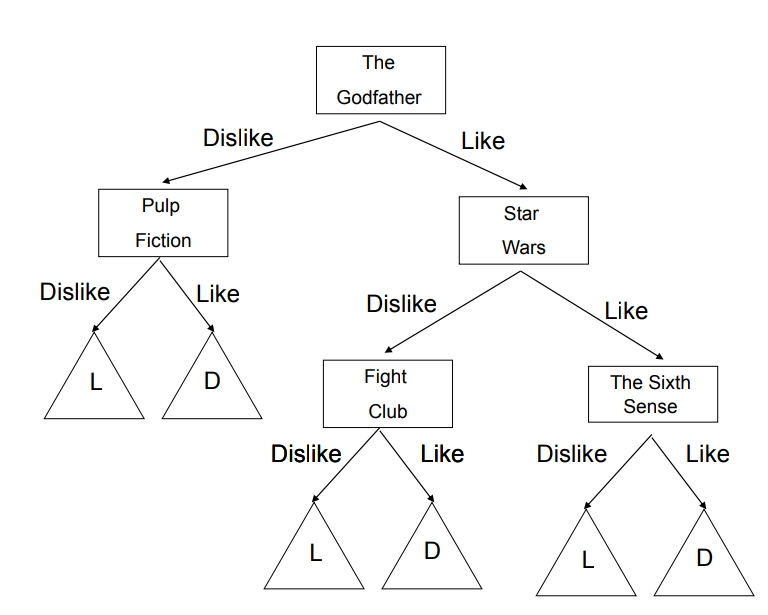
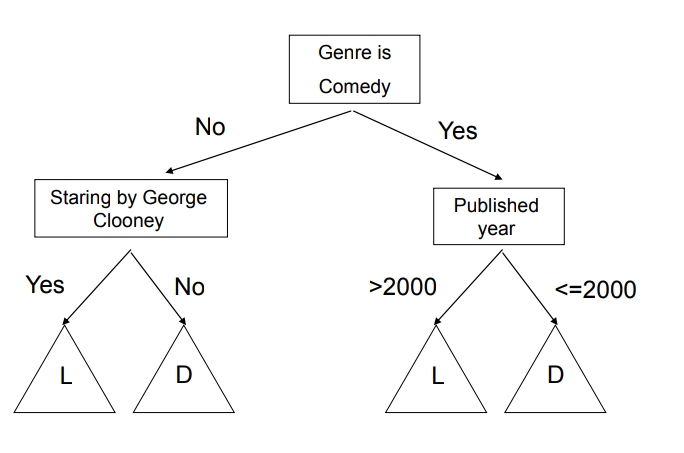
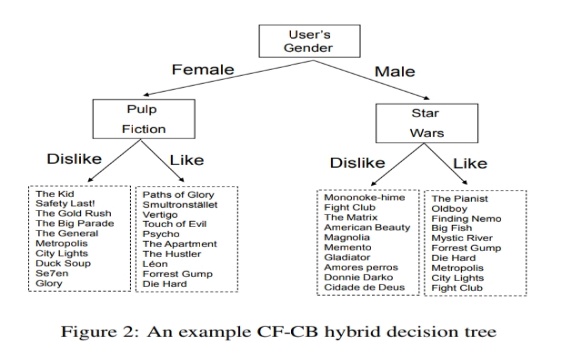
 

Figure 1: Left: A CF decision tree for whether users like the movie” The Usual Suspects” based on their preferences to other movies such as The Godfather, Pulp Fiction etc. A leaf labelled with” L” or ” D” correspondingly indicates that the user likes/dislikes the movie” The Usual Suspects”. Right: A CB decision tree for Bob.

• Content-Based Approach - Li and Yamda [LY04] and Bouza et al. [BRBG08] propose to use content features to build a decision tree. A separate decision tree is built for each user and is used as a user profile. The features of each of the items are used to build a model that explains the user’s preferences. The information gain of every feature is used as the splitting criteria. Figure 1 (right) illustrates Bob’s profile. It should be noted that although this approach is interesting from a theoretical perspective, the precision that was reported for this system is worse than that of recommending the average rating.

• Hybrid Approach - Hybrid decision tree can also be constructed. Only a single tree is constructed in this approach. The tree is similar to the collaborative approach, in that it takes user’s attributes as attributes to split by (such as her liking/disliking of a certain movie) but the attributes it uses are general attributes that represent the user’s preference for the general case, based on the content of the items. The attributes are constructed based on the user’s past ratings and the content of the items. For example, a user who rated negatively all movies of genre comedy is assigned a low value in a” degree of liking comedy movies” attribute. Similarly, to the collaborative approach, the tree constructed is applicable to all users. However, it is now also applicable to all items since the new attributes represent the user’s preferences for all items and not just a single given item. Figure 2 illustrates such a hybrid tree.



1. **Methods & Material**

In this section, the research study that was conducted during this project is presented providing a basic knowledge of RS-adapted decision tree, test/train split algorithm, use of data in algorithm, accuracy and precision as result of model.

**3.1 Test/Train Split Algorithm**

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

1. Train Dataset: Used to fit the machine learning model.
2. Test Dataset: Used to evaluate the fit machine learning model.

The objective is to estimate the performance of the machine learning model on new data, data not used to train the model. There is no optimal split percentage, common split percentages include:

1. Train: 80%, Test: 20%
2. Train: 67%, Test: 33%
3. Train: 50%, Test: 50%

**3.2 RS-Adapted Decision Tree**

In recommender systems the input set for building the decision tree is composed of ratings. Ratings can be described as a relation < ItemID, UserID, Rating > (in which < ItemID, UserID > is assumed to be a primary key). The attributes can describe the users, such as the user’s age, gender, occupation. Attributes can also describe the items, for example the weight, price, dimensions. Rating is the target attribute which the decision tree classifies based on the training set, the system attempts to predict the rating of items the user does not have a rating for and recommends to the user the items with the highest predicted rating. The construction of a decision tree is performed by a recursive process. The process starts at the root node with an input set (training set). At each node an item attribute is picked as the split attribute. For each possible value (or set of values) child-nodes are created and the parent’s set is split between child-nodes so that each child-node receives as input-set all items that have the appropriate value(s) that correspond to this child-node. Picking the split-attribute is done heuristically since we cannot know which split will produce the best tree (the tree that produces the best results for future input), for example the popular C4.5 algorithm ([Qui93]) uses a heuristic that picks the split that produces the largest information gain out of all possible splits. One of the attributes is pre-defined as the target attribute. The recursive process continues until all the items in the node’s set share the same target attribute value or the number of items reaches a certain threshold. Each leaf node is assigned a label (classifying its set of items), this label is the shared target attribute value or the most common value in case there is more than one such value.

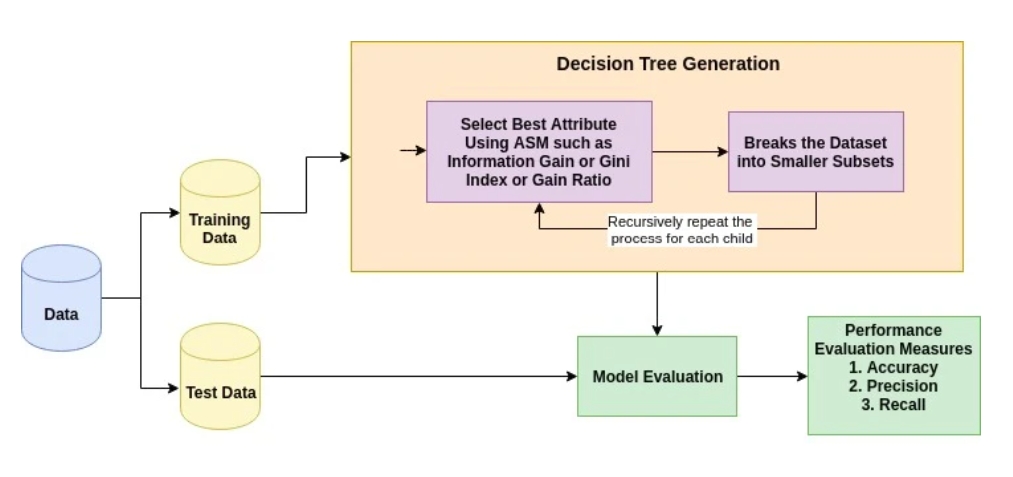
**Data**

The available data is the most vital part of any test/train algorithm. The most important aspects are the quality and amount of the available data. To run some sort of similarity function to split items or users in a system, the data needs to be arranged into feature vectors with a set of feature values. To achieve the best results, a large amount of data is needed and more importantly the absence of data points needs to be minimal. The amount of data in most cases is not a problem nowadays since companies store all kinds of user and item data in large databases.

Another important aspect in test/train algorithm is to understand the available data. In a system where items are rated using some sort of scale, e.g. a rating from zero to five, it is easy to interpret a user’s preferences. However, in systems where the set of items is not predefined, as in E-marketplace where users upload items which are removed when sold, it is much harder to determine a user’s preference. In this type of 6 market, it is usual to use data of the type of user-item interaction as a measure for preference. These interactions can be e.g. “liking” on items, time spent looking at an item, a comment on an item. This would mean that users that view, “like” and comment the same set of items are like each other. However, for new users that have a small amount of item interaction, it is impossible to determine in which group they should be placed. An analysis of the available data points should be made to pick a suitable splitting algorithm.

* 1. **How does Decision Tree Classifier work?**

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the conditions will match:
4. All the tuples belong to the same attribute value.
5. There are no more remaining attributes.
6. There are no more instances.



**3.4 Attribute Selection Measures**

Attribute selection measure is a heuristic for selecting the splitting criterion that partition data into the best possible manner. It is also known as splitting rules because it helps us to determine breakpoints for tuples on a given node. ASM provides a rank to each feature (or attribute) by explaining the given dataset. Best score attribute will be selected as a splitting attribute (Source). In the case of a continuous-valued attribute, split points for branches also need to define. Popular selection measures are Information Gain, Gain Ratio, and Gini Index.

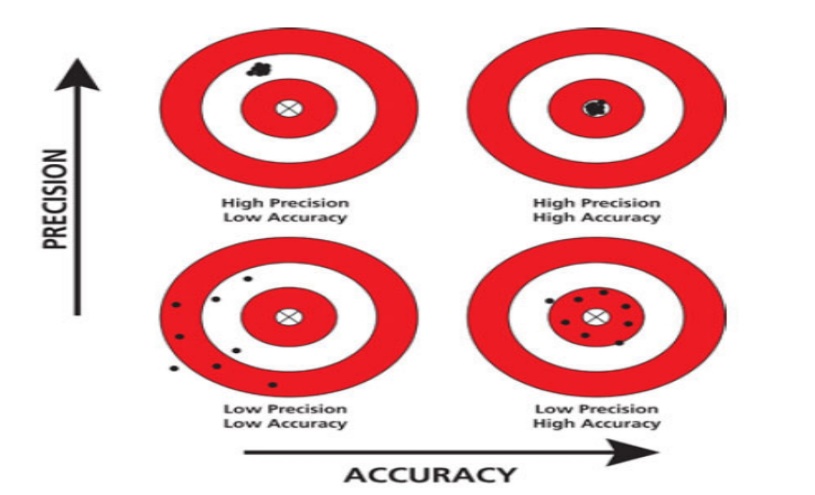
**3.5 Accuracy & Precision**

What does accuracy mean?

If a measurement is accurate, it means that it agrees closely with the accepted standard for that measurement. For example, if we estimate a project's size to x and the actual size of the finished project is equal to or very close to x, then it is accurate, but it might not be precise. The closer a system's measurements to the accepted value, the more accurate the system is considered to be. Humans make errors all the time, but if you're using a project management software to help you scope, you will begin to have a more accurate project measurement and refined process.

What does precision mean?

A measurement that is precise means that it agrees with other measures of the same thing. In the sense of project scoping, let's take an estimation of workload as an example. If we estimate the size of several projects and they, in the end, are all close to or equal to what we predicted, then we can start to get a sense of the precision of our estimates. But first and foremost, each project needs to be as accurate as possible.



When scoping a project, you want to be as close to the actual workload as possible. Defining the scope means that you and your client are figuring out and documenting a list of specific project goals. That could be features, functionalities, deliverables, deadlines, and ultimately costs of the project. Project scope helps with resource planning and time management of the project. Accuracy and precision are used in the context of measurement, e.g. the size of a project and therefore are both helpful when defining the scoping.

Accuracy and precision are alike only in the fact that they both refer to the quality of measurement, but they are very different indicators of measurement. Accuracy is the degree of closeness to true value. Precision is the degree to which an instrument or process will repeat the same value. In other words, accuracy is the degree of veracity while precision is the degree of reproducibility.

**4** **Result & Discussion**

In this section the data analysis is presented, and a method is chosen. This choice is motivated using results drawn from various tests done on the data.

**4.1 Data Pre-processing:**

[Evaluation of Model Analysis]

1. Firstly, to compare users, a rating system needs to be defined. In the case of user base’s, a user’s interest in an item can be defined by several weighted variables. The rating is therefore defined as an aggregated score which is derived from the data points discussed. To be able to analyse the data and compare users, these scores need to be standardized into the same scale. The primary data points used in the analysis of the user base during this project were item views and item likes. A visualization of the resulting user-item ratings matrix for n users and m items is shown below. In the data set there is a column called Temperature which is int datatype model, we changed the temperature into category datatype as it is an ordinal datatype.

Graphical user interface, text

Description automatically generated

**4.2 Data Analysis**

Since no clear split separations can be deduced, a deeper data analysis was made to find underlying properties that can be used to get a clearer picture of the data. These properties are based on the segmentation system described above.

The focus during this phase of the project was to figure out if and how the spending score and annual income split going to change over time. To answer this question two methods were tested. Firstly, we use classifier algorithm, where we couldn’t be able to increase accuracy, so we are using supervised algorithms to group the information according to user’s choice.

The upcoming graph is about gender where the destitution of male and female, where that contains huge number of females than male users.

Chart

Description automatically generated with medium confidence

Data set contains more female users than male

**Chart, bar chart, funnel chart

Description automatically generated**

The above graph says about the income of the individual, people earning 25,000 - 38,000 has highest customer count and significantly 12,500 - 25,000 can be considered for market.

Chart

Description automatically generated

The above graph says about the weather report which is convenient for the users. As per the dataset we find sunny region propose to maximum user base. Product should focus mainly on this region as recommended above.

Chart, funnel chart

Description automatically generated

The above graph tends to say age group of 20 – 30 (years) has the highest customer count. The product often needs to be designed based on their interests will be considered for better market.

These are few criteria, used from dataset in getting better knowing on the user interest. There are many more section in the dataset will be used as data in the source code for better output decision tree making. Some of the criteria excluded in this analysis are [Destination, Count] [Passenger, Count] [Temperature, Count], [Occupation and Education, Count], [Expenses, Count] [Time, Count] [Coupon, Count] etc

**5. Implementation**

In this section, the implementation of the final system is described.

**5.1 Decision Tree Classifier**

The first part of this project consisted of gathering and analysing the data of the Decision Tree. To achieve this,the programming language Python was used combination with the libraries of numPy, pandas, matplotlib, seaborn and sckit learn classifier we imported decision tree classifier, used for scientific computing. This program is responsible of three major parts of the system:

1. Understanding the customer potential using supervised learning. This part of the system is also responsible for the data pre-processing and importing the python libraries, in the data set there is a column called Temperature which is int datatype model, we changed the temperature into category datatype as it is an ordinal datatype.
2. After the pre-processing the model we consider various factors in our data set that is age, gender, annual income and spending score through this data we visualize and fed into the decision tree classifier. This was done using seaborn. The latter was also used in the development stages for splitting data given some split algorithm such as test/train split method. The **Test/Train Split Method** is one of the most popular methods to train the model. At the initial stage of the development, all the plotting and data visualization was made with the python plotting library matplotlib.
3. Finally, the understanding the trained model potential is researched through the accuracy and precision. The better accuracy and precision model can be next implemented and showcase as decision tree.

**5.2 System Architecture / Ideation Map**

Import all the necessary libraries (numPy, pandas,

matplotlib)

Analyze the Result

Running the test cases of the program

Importing the dataset (CSV) file to Jupiter Notebook

Training model using test/train split algorithm

Performing exploring data analysis (EDA)

Using decision tree classifier method for coupon recommendation

In this project, I will demonstrate how to build a model predicting recommender system using decision tree in Python using the following steps:

1. Data preparation
2. Exploratory Data Analysis
3. Analysing the correlation to numerical features
4. Encode categorical features
5. Using Machine Learning Decision Tree Algorithm for getting output

**Hardware & Software Requirements:**

4GB RAM

Operating System

Windows 10, Core i5

Anaconda Navigator along with Python 3.8

**About Dataset:**

This data was collected via a survey on Amazon Mechanical Turk. The survey describes different driving scenarios including the destination, current time, weather, passenger, etc., and then ask the person whether he will accept the coupon if he is the driver.

* 1. **Mathematical operations**

Classification accuracy is defined as the ratio of the number of correctly classified cases and is equal to the sum of TP and TN divided by the total number of cases (TN + FN + TP + FP).

**Accuracy=TP+TN/TN + FN + TP + FP**

Precision is defined as the number of true positives (TP) over the number of true positives plus the number of false positives (FP).

**Precision=TP/TP+FP**

Recall is defined as the number of true positives (TP) over the number of true positives plus the number of false negatives (FN).

**Recall=TP/TP+FN**

Sensitivity refers to the rate of correctly classified positive and is equal to TP divided by the sum of TP and FN. Sensitivity may be referred as a True Positive Rate

**Sensitivity=TP/FN+TP**

Specificity refers to the rate of correctly classified negative and is equal to the ratio of TN to the sum of TN and FP

**Specificity=TN/TN+FP**

True Positive (TP) is the number of correct predictions that an instance is true, or in other words; it is occurring when the positive prediction of the classifier coincided with a positive prediction of target attribute. True Negative (TN) is presenting a number of correct predictions that an instance is false, (i.e.) it occurs when both the classifier, and the target attribute suggests the absence of a positive prediction. The False Positive (FP) is the number of incorrect predictions that an instance is true. Finally, False Negative (FN) is the number of incorrect predictions that an instance is false.

**Using Decision Tree:**

Decision trees are very easy as compared to the random forest. A decision tree combines some decisions, whereas a random forest combines several decision trees. Thus, it is a long process, yet slow. Whereas a decision tree is fast and operates easily on large data sets, especially the linear one. A decision tree is fast and operates easily on large data sets, especially the linear one. The random forest model needs rigorous training. When you are trying to put up a project, you might need more than one model. Thus, more random forests, more the time.

**Conclusion & Future work:**

The metrics we used are calculated using sklearn wrapper so can be trusted for the model performance. Our end goal was to have a tuned model that could beat the untuned benchmark which it did by a very fine margin. So, the solution described below is satisfactory to our initialexpectations. We generated a final model with above list of tuned parameters.

Machine learning techniques are widely used data analysis methods in various business and industrial sectors. The main reason for that because ML can build predictive models to produce better predictions and achieve the desired level of accuracy, leading to better outcomes. Building the models is an easy and straight forward process. The main challenges in data analysis are data preparation and cleaning, the selection of appropriate models and attributes used in their implementation.

The aim of the project is how to use supervised learning techniques for analysis and making predictions using existing dataset in Recommender system. To find how they can be used together in a process of converting raw data to effective decision-making knowledge. Thus, in this project are used decision tree algorithms to predict a person will accept the coupon if he is the driver. As per the model, model score is 0.9987.

**Appendix**

**Source Code:**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

**Graphical user interface

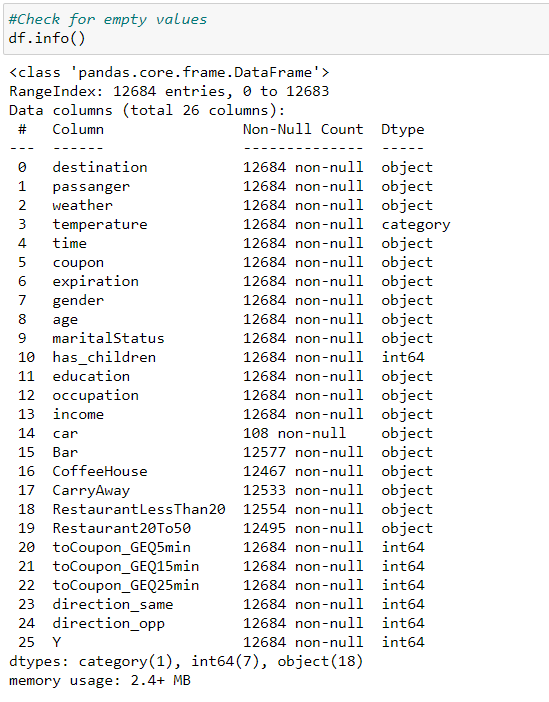
Description automatically generated**

**Table

Description automatically generated with low confidence**

**Table

Description automatically generated**

****

**Graphical user interface, text, application, email

Description automatically generated**

**Table

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

**A picture containing graphical user interface

Description automatically generated**

**Text

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, waterfall chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Text

Description automatically generated**

**A picture containing table

Description automatically generated**

**A picture containing text, white, screenshot

Description automatically generated**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Table

Description automatically generated**

**References**

* Francesco Ricci, Lior Rokach and Bracha Shapira. Introduction to Recommender Systems handbook. Springer, 2011.
* "How to split data into three sets (train, validation, and test) And why? | by Samarth Agrawal | Towards Data Science" <https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c>
* "Machine Learning in Coupon Recommendation | by João Felipe Guedes | Towards Data Science" <https://towardsdatascience.com/machine-learning-in-coupon-recommendation-2bdae281d840>
* Cognibot Machine learning.
* Siraj Ravel You tube.