DataMiningandDataWarehousing Laboratory (CSPC-328)

B.TechVIthSemester
(January–June2024)

<u>Submittedby</u>

Kirtan Gohil(21103073) Group-G3

<u>Submittedto</u>

Dr.SamayveerSingh

DepartmentofComputerScience&Engineering
Dr.B.R.AmbedkarNationalInstituteofTechnology Jalandhar
-144008,Punjab,India TableofContent

Sr.No.	PracticalName	Date	Page No.	Remarks
1.	DesigningDatabaseUsingERModelling a)HospitalManagementSystem b)LibraryManagementSystem	29-01-2024	3-6	
2.	NormalizingaDatabase	5-02-2024	7-19	
3.	ProgramstoimplementProcedures, CursorsandTriggersinadatabase	12-02-2024	20-23	
4.	Writeprogramstoimplementand understandusageofDatamarts.	19-02-2024	24-26	
5.	Feature Selection and Variable Filtering.			
6.	Perform Associative Mining In Weka and Orange on large datasets			
7.	Perform K-Nearest Neighbour Classification in Weka and Orange			
8.	Perform DBSCAN Clustering in Weka and Orange			
9.	Perform Heirarchial Clusrtering			
10.				

Practical1

Aim:-DesigningDatabaseUsingERModelling

Que1CreatedatabasedesignforHospitalManagementSystemusingER Modelling

The patient, physician, department, room, and appointment are the entities that make up the hospital administration system.
The following is a relationship between these entities areas:

An appointment is for one patient and one doctor. A patient may have one or more appointments. A doctor may schedule many appointments with various patients.
One department is assigned to a doctor.
A department may employ several physicians.
One patient can be assigned to one room, and one or more patients can be housed in a room.
A doctor is in charge of each room, however they can oversee more than one. These relationships allow us to develop the subsequent ER model:

1.Entities:

- Patient with attributes (Name, Age, Room Number, and Patient ID).
- Physician with the following attributes: DepartmentID, Name, Specialty, DoctorID.
- Department including features like DepartmentName, DepartmentID.

Room has the following attributes: bed count, supervising doctor ID, room number.

- Appointment with the following attributes: PatientID, DoctorID, Date, Time, Appointment ID.

2. Relationships:
A patient's relationship with an appointment is symbolized by a "has" relationship.
A doctor-patient connection is based on a "conducts" relationship.
A department and a doctor are associated, represented by a "assignedto" relationship.
Multiple doctors are associated with a department through the "employs" relationship.
A patient and a room are connected through a "assignedto" relationship.
A room can have a relationship with numerous patients, represented by a "houses" relationship.
A room has a relationship with a doctor, which is represented by a "supervisedby" relationship. An diagram representing things as boxes and relationships as lines linking these boxes—often with additional symbols to signify the kind and cardiacality of the interactions—would be the visual representation of the ER model.
The relationships and entities within the hospital management system are shown in Fig. 1.1.

The patient, doctor, department, room, and appointment are the five main entities that are included. Patients may schedule many appointments, with a doctor and a single patient at each visit.Physicians are assigned to departments, and each department may have more than one physician on staff.Patients are assigned to rooms, and each room can accommodate several patients under a single doctor's care.The ER graphic also shows how a doctor is able to oversee many rooms.The entities are linked together by a number of links, including "has," "conducts," "assigned to," "employees," "houses," and "supervisedby," which illustrate the many relationships and interactions that exist in a medical setting.The diagram shows the relationships between the various components of the system and acts as a visual representation of the data model.

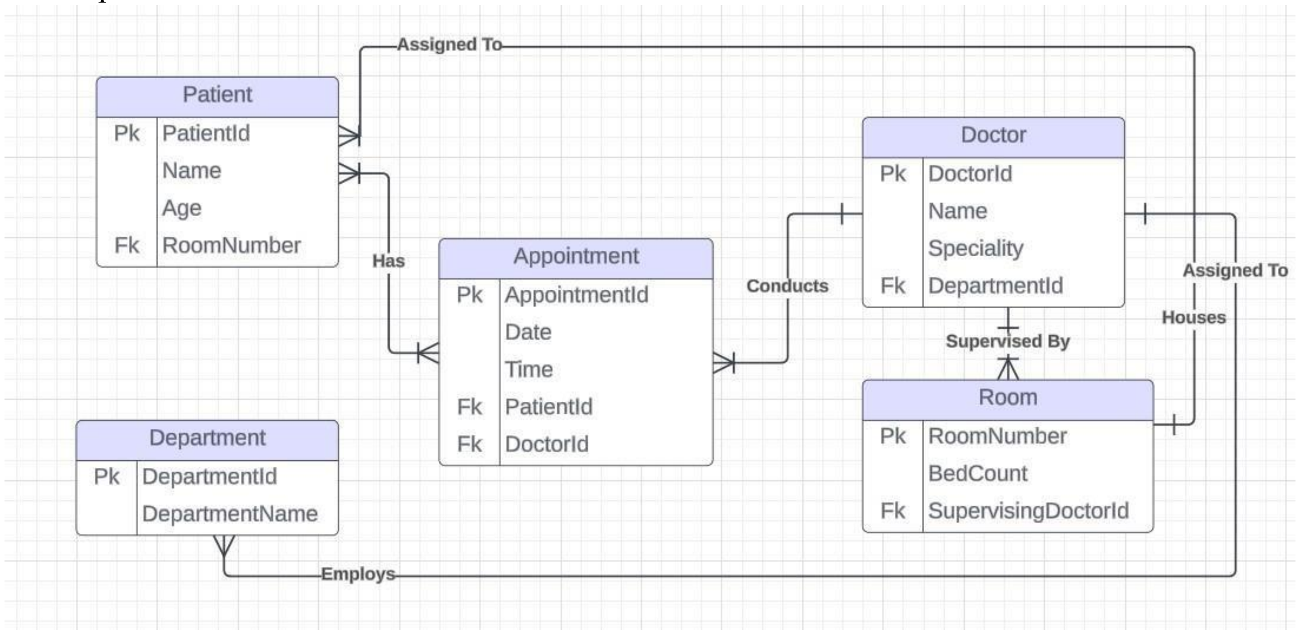


Fig.1.1:ERdiagramforHospitalManagementSystem

Que2CreatedatabasedesignforLibraryManagementSystemusingER Modelling

The following entities are included in the library management system: book, author, borrower, genre, and loan.The following is a relationship between these entities areas: A book is authored by one or more writers. • A writer can pen one or more books.
• A borrower may check out many books, but a book may be checked out by just one borrower. •in real time.
• A book falls into a specific genre.
• A genre can be connected to more than one book.
• The loan specifies when a book was checked out and when it must be returned.

These relationships led us to derive the subsequent ER model:

1. Entities
• Book with attributes: Title, ISBN, BookID, GenreID.
•Author with attributes: Name, BirthDate, and AuthorID.
• Borrower with properties: Name, Address, Phone, and Borrower ID.
• Genre with attributes (GenreName, GenreID).
• Loan with attributes: BookID, BorrowerID, Borrow Date, Due Date, Loan ID.
2. Relationships:
• A book is linked to its author(s) by means of a "writtenby" relationship.
One or more books are associated with an author via a "writes" relationship.

• A "borrows" relationship connects a borrower with books.
• A book and borrower have a relationship thanks to the "isborrowedby" connection.
• A book and a genre are connected by a "belongsto" relationship.
• Aloani is related to a borrower and a book through a "issued for" relationship. • A genre is connected to many books through a "encompasses" relationship.

To visualize the ER model, entities would be shown as boxes with relationships between them shown as lines or arrows. The types and cardinality of each link would be represented by annotations or symbols.

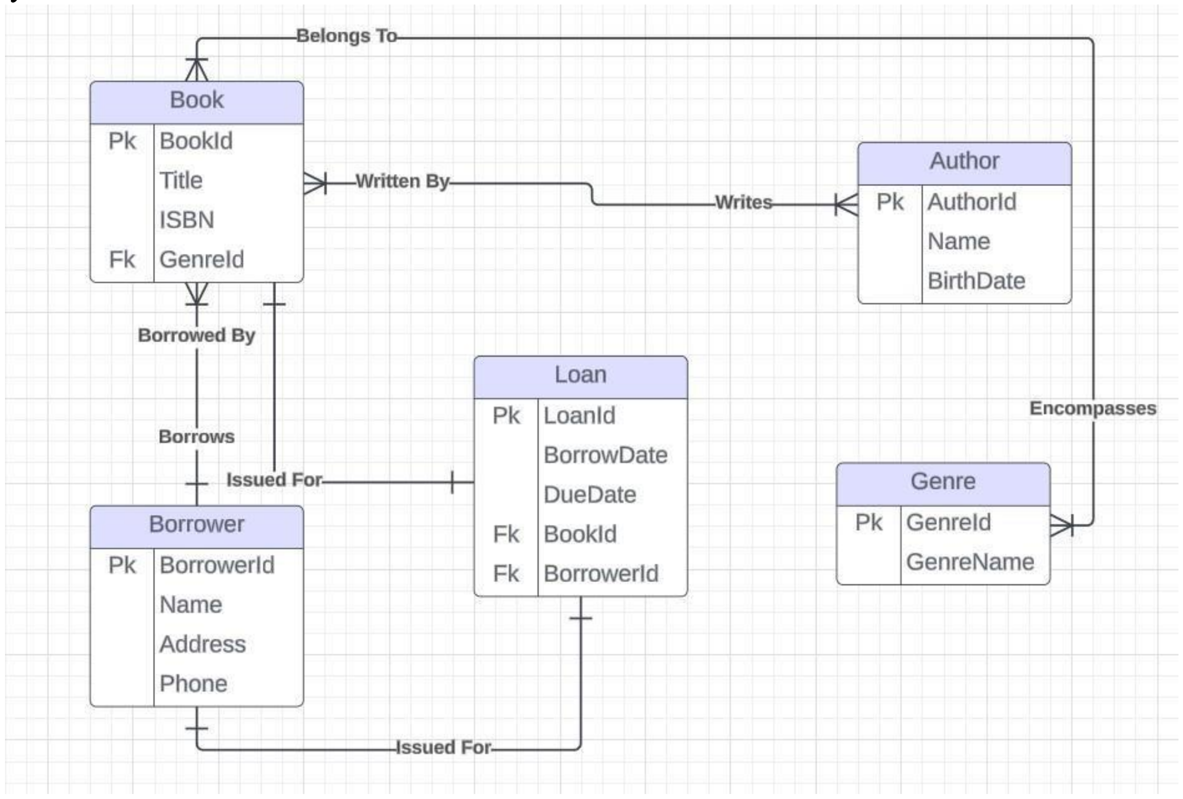


Fig.1.2:ERdiagramforLibraryManagementSystem

Figure 1.2 illustrates the connections and entities in the Library Management System.There are five main components to it: Book, Author, Borrower, Genre, and Loan.The graphic shows how a book is linked to one or more writers by a "written by" relationship, enabling numerous authors to contribute to a single work.Books are linked to authors by a "writes" relationship, meaning that an author is able to write more than one book.The relationship "borrows" links borrowers to books; this means that one borrower may check out numerous books at once, but only one borrower may check out a book at a time.Books are grouped by genres using a "belongsto" relationship, which indicates that a given book is part of a particular genre. Genres might include more than one book.The "issuedfor" relationships bind loans to both borrowers and books, indicating the date a book was borrowed and the return deadline.

Practical2

Aim:-NormalisingaDatabaseUsingGriffithNormalisation Tool

Que1Understandthefunctionaldependenciesandnormalizeeach functional dependencyupto2NF,3NF,andBCNFusingnormalizationtoolfrom GriffithUniversity.
Foreachquestion:

•Findtheminimalcover.
•Identifythecandidatekey(s)orprimarykey.
•Checkforpartialdependenciestodetermineiftherelationisin2NF.
•Checkfortransitivedependenciestoassessiftherelationisin3NF.
•CheckfortransitivedependenciestoassessiftherelationisinBCNF.

A.StudentDatabase:
Giventherelation:
StudentCourses(StudentID,CourseName,Instructor,CourseCredits)
andthefunctionaldependencies: StudentID,CourseName→Instructor
CourseName→CourseCredits

PreviousFunctionalDependencies

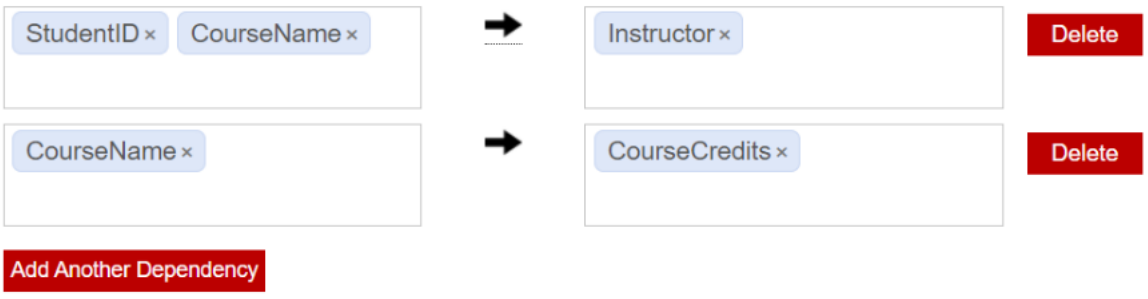
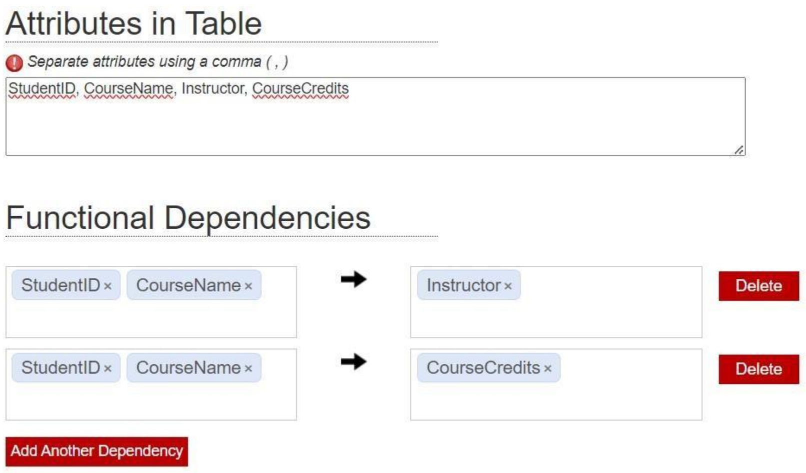


Fig1.A.1 ModifiedFunctionalDependencies


Attributes in Table

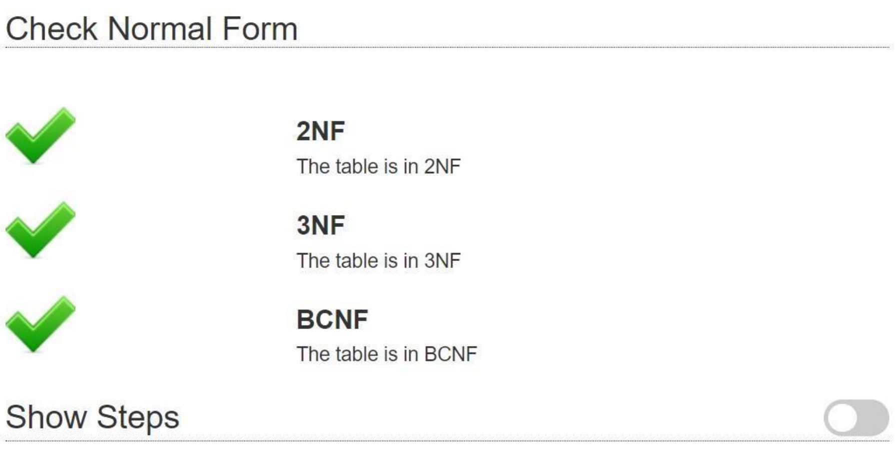
Separate attributes using a comma (,)

StudentID, CourseName, Instructor, CourseCredits

Functional Dependencies

Fig1.A.2

Result


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF

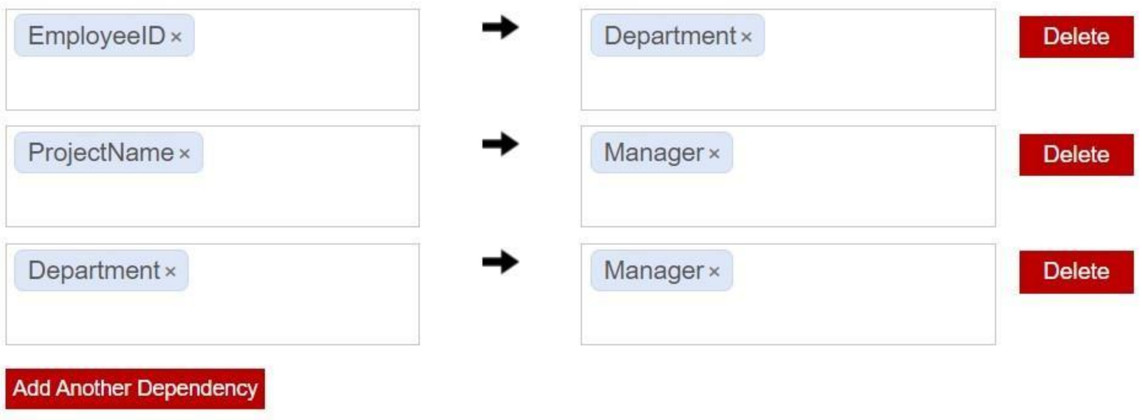
Show Steps

Fig1.A.3

Fig1.A.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.A.2 showsnewFunctionalDependencieswhichshowsIfyouknowastudent'sIDand thenameofthecoursethey'retaking,youcandeterminetheinstructorwhoteaches thatcourseandhowmanycreditsthatcoursecarries.Fig1.A.3showstheresultthat newFDsareinBCNF.

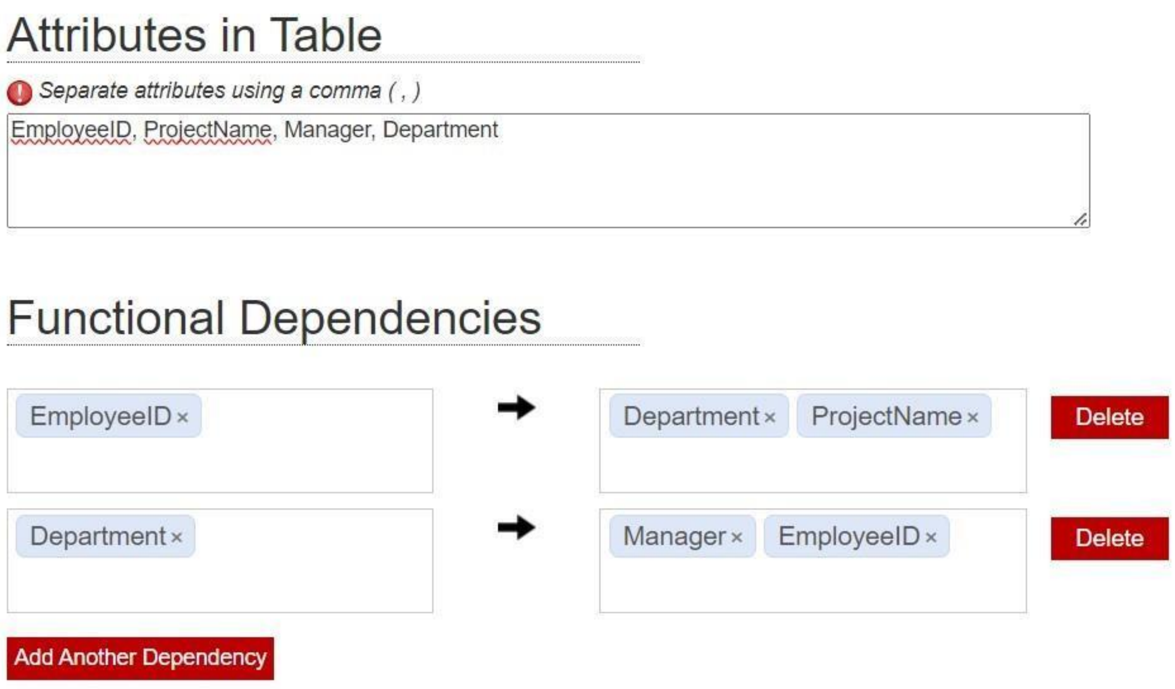
B.EmployeeManagement:
Giventherelation:
EmployeeProjects(EmployeeID,ProjectName,Manager,Department)
withthefunctionaldependencies:
EmployeeID→Department
ProjectName→Manager Department→Manager

PreviousFunctionalDependencies



EmployeeID ×	→	Department ×	Delete
ProjectName ×	→	Manager ×	Delete
Department ×	→	Manager ×	Delete

Fig1.B.1 ModifiedDependencies


Attributes in Table

Separate attributes using a comma (,)

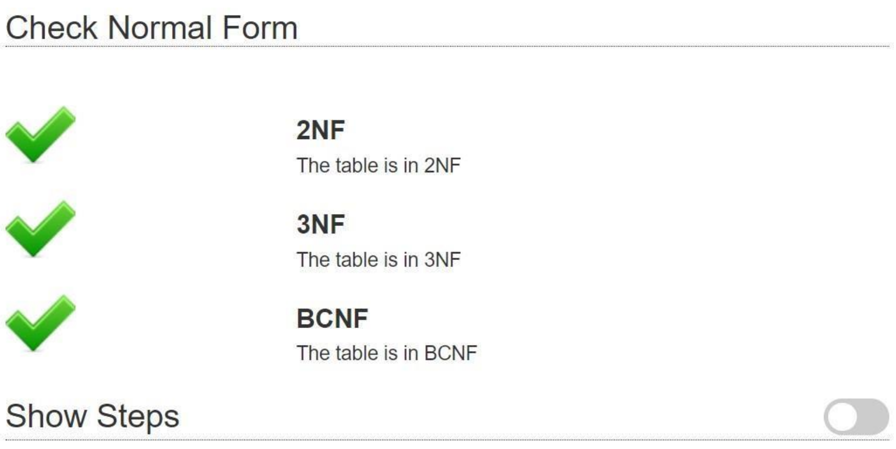
EmployeeID, ProjectName, Manager, Department

Functional Dependencies

EmployeeID → Department, ProjectName

Department → Manager, EmployeeID

Fig1.B.2 Result


Check Normal Form

2NF
The table is in 2NF

3NF
The table is in 3NF

BCNF
The table is in BCNF

Fig1.B.3

Fig1.B.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.B.2 showsnewFunctionalDependencieswhichshowsGivenanEmployeeID,wecan determinetheProjectNameandDepartmentassociatedwiththatemployee. GivenaDepartment,wecandeterminetheManagerandEmployeeIDassociated withthatdepartment.Fig1.B.3showstheresultthatnewFDsareinBCNF.

C.LibrarySystem:

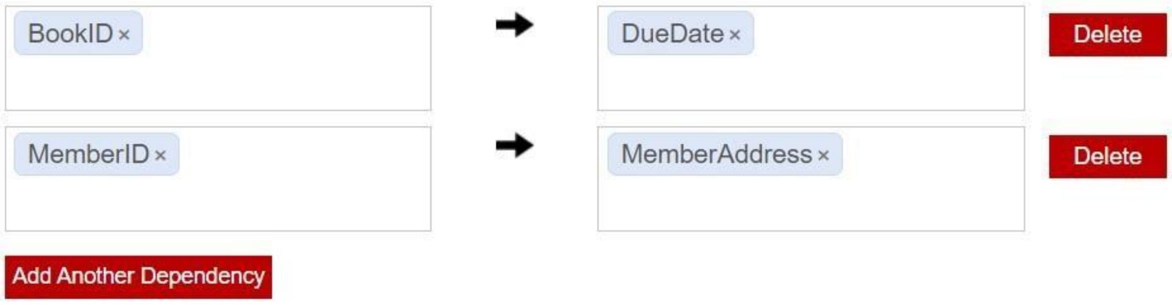
Considertherelation:

BookLending(BookID,MemberID,BorrowDate,DueDate,MemberAddress)

andthefunctionaldependencies: BookID→DueDate

MemberID→MemberAddress

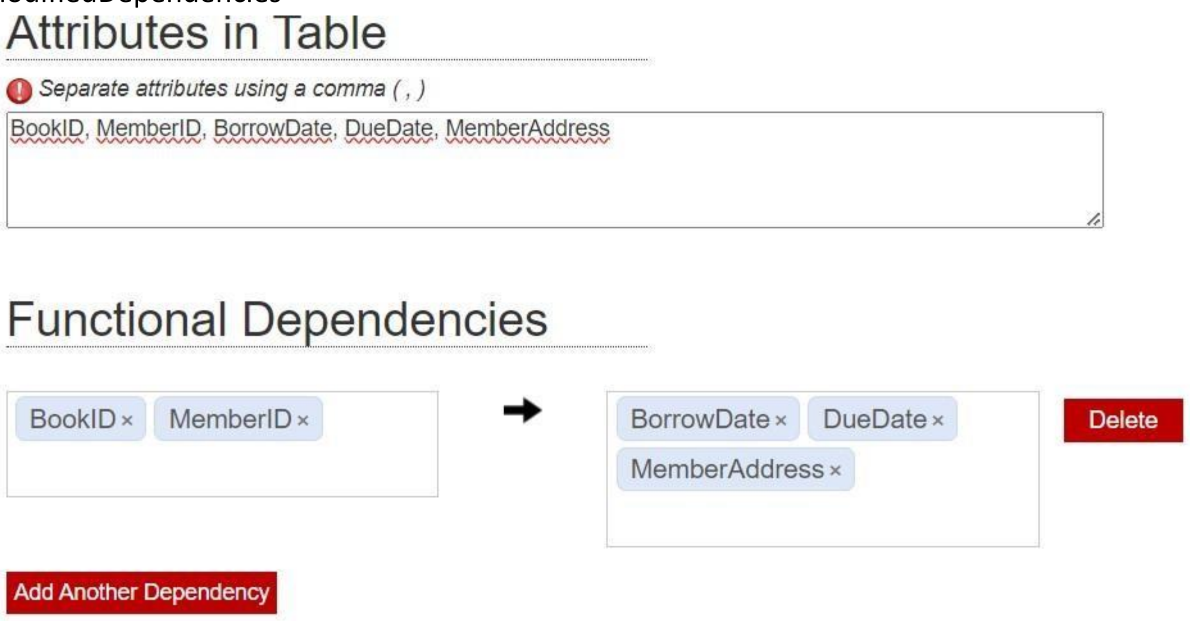
PreviousFunctionalDependencies


BookID × → DueDate × Delete

MemberID × → MemberAddress × Delete

Fig1.C.1

ModifiedDependencies


Attributes in Table

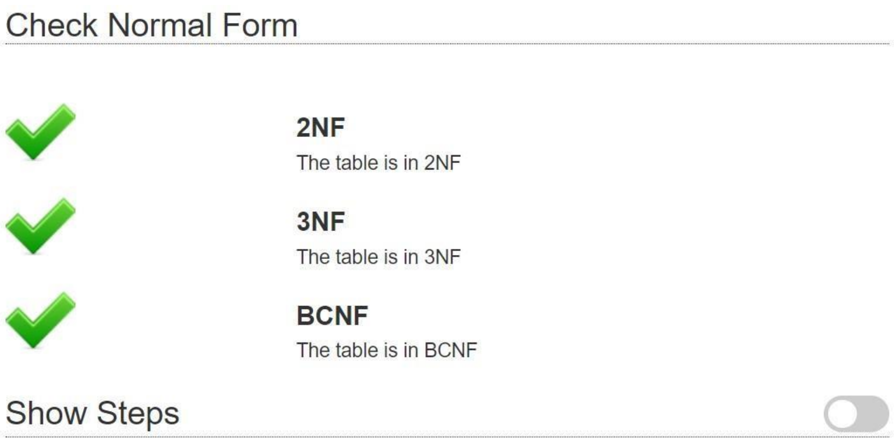
Separate attributes using a comma (,)

BookID, MemberID, BorrowDate, DueDate, MemberAddress

Functional Dependencies

BookID × MemberID × → BorrowDate × DueDate × MemberAddress × Delete

Fig1.C.2 Result


Check Normal Form

✓ **2NF** The table is in 2NF

✓ **3NF** The table is in 3NF

✓ **BCNF** The table is in BCNF

Show Steps

Fig1.C.3

Fig1.C.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.C.2 showsnewFunctionalDependencieswhichshowsIfyouknowwhichbookis borrowedbywhichmember,youcandeterminethemember'saddress,theduedate ofthebook,andthedateitwasborrowed.Fig1.C.3showstheresultthatnewFDs areinBCNF.

D.HospitalManagement:

-Fortherelation:

PatientTreatment(PatientID,Treatment,Doctor,DoctorSpecialization)
withthefunctionaldependencies: Doctor→DoctorSpecialization PatientID,Treatment→Doctor

PreviousFunctionalDependencies

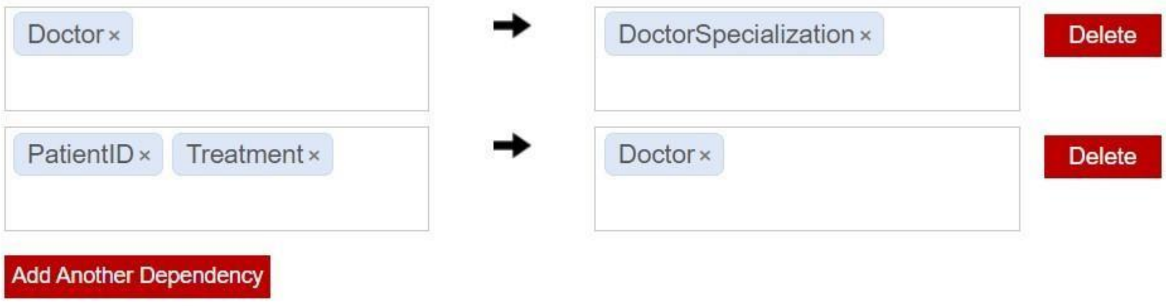
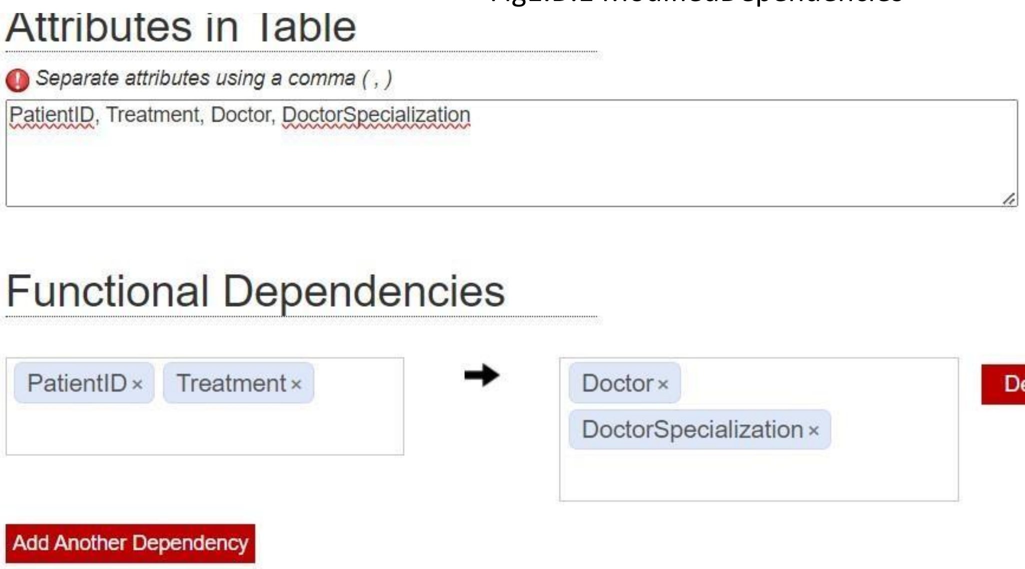
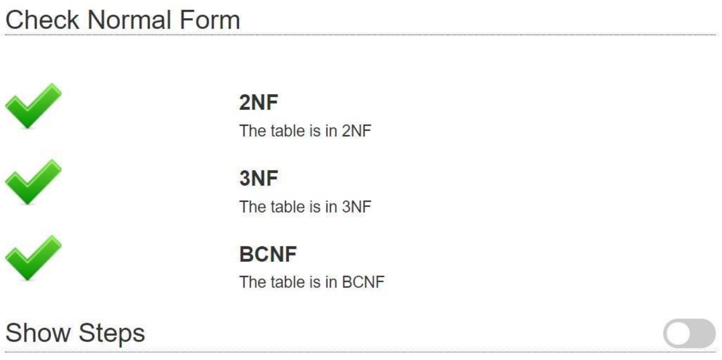


Fig1.D.1 ModifiedDependencies



PatientID, Treatment, Doctor, DoctorSpecialization

Fig1.D.2 Result



2NF
The table is in 2NF

3NF
The table is in 3NF

BCNF
The table is in BCNF

Fig1.D.3

Fig1.D.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.D.2 showsnewFunctionalDependencieswhichshowsIfyouknowthePatientIDandthe treatmenttheyareundergoing,youcandeterminewhichdoctorisresponsiblefor providingthattreatment,alongwiththedoctor'sspecialization.Fig1.D.3showsthe resultthatnewFDsareinBCNF.

E.AirlineReservationSystem:

-Giventherelation:
FlightReservations(FlightNumber,Date,PassengerID,SeatNumber,ClassType,
Price,DepartureTime,ArrivalTime,DepartureCity,ArrivalCity) -Functionaldependenciesare:
FlightNumber,Date→DepartureTime,ArrivalTime,DepartureCity,ArrivalCity
SeatNumber,Date,FlightNumber→PassengerID,ClassType,Price
ClassType→Price
PassengerID→DepartureCity

PreviousFunctionalDependencies

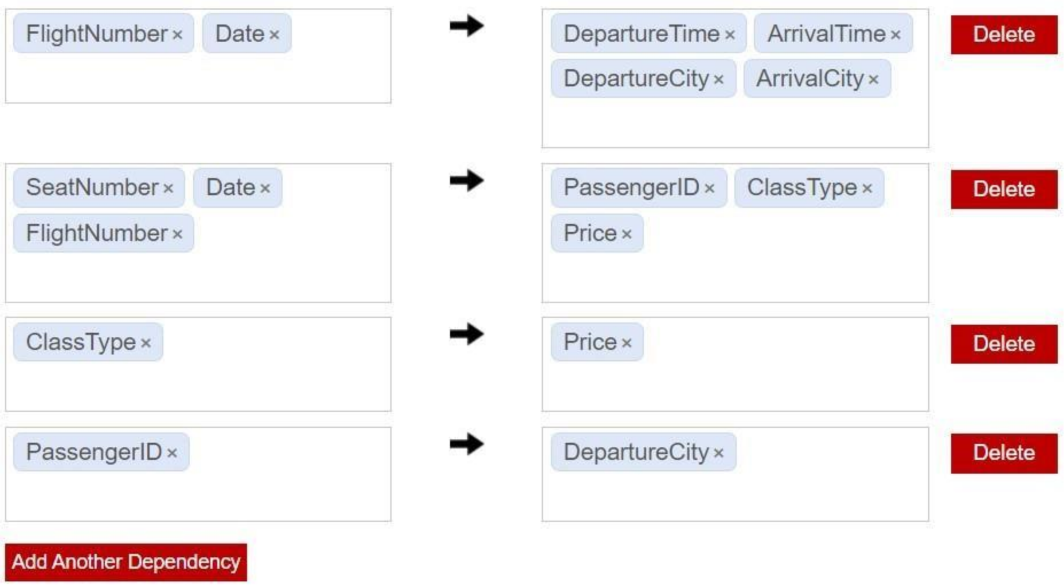
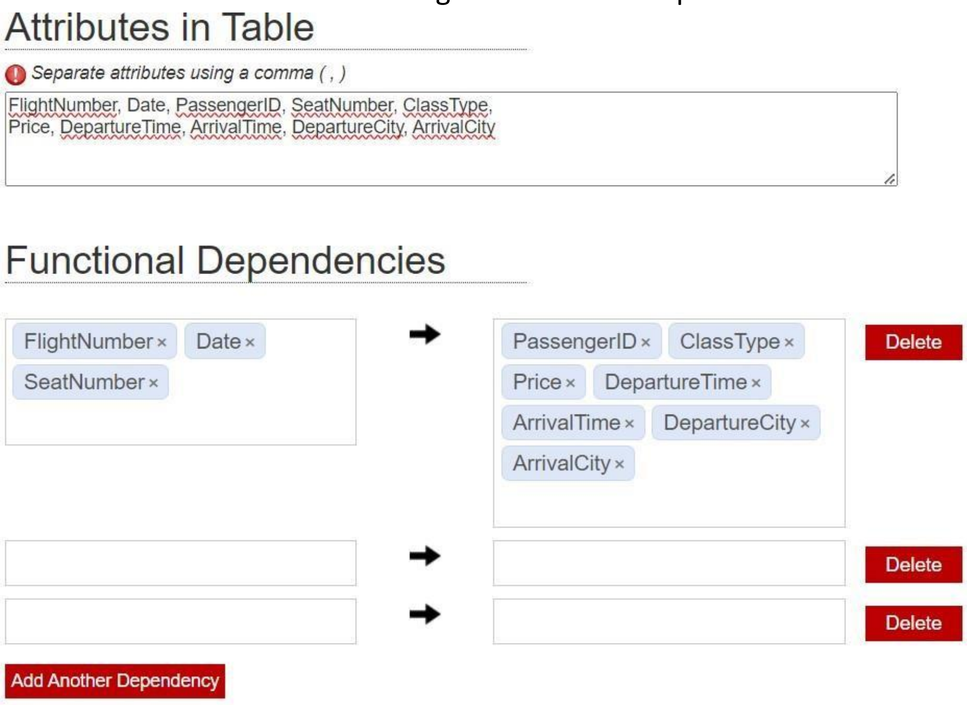

FlightNumber, Date → DepartureTime, ArrivalTime, DepartureCity, ArrivalCity
SeatNumber, Date, FlightNumber → PassengerID, ClassType, Price
ClassType → Price
PassengerID → DepartureCity

Fig1.E.1 ModifiedDependencies


Attributes in Table

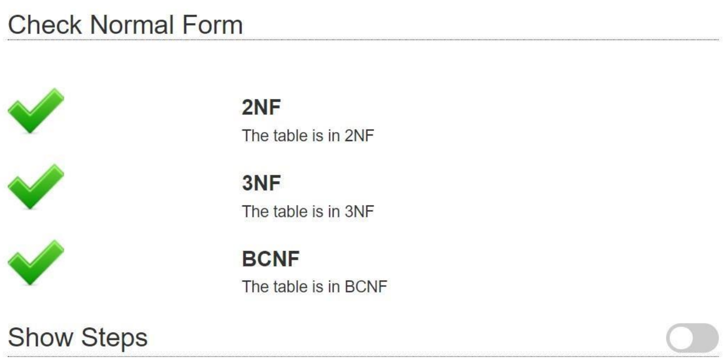
Separate attributes using a comma (,)

FlightNumber, Date, PassengerID, SeatNumber, ClassType, Price, DepartureTime, ArrivalTime, DepartureCity, ArrivalCity

Functional Dependencies

FlightNumber, Date, SeatNumber → PassengerID, ClassType, Price, DepartureTime, ArrivalTime, DepartureCity, ArrivalCity

Fig1.E.2 Result


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF

Show Steps

Fig1.E.3

Fig1.E.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.E.2 showsnewFunctionalDependencieswhichshowsthatifyouhaveinformation abouttheflightnumber,date,andseatnumber,youcandeterminethedetailsrelated tothatspecificbooking,includingthedepartureandarrivaltimes,cities,passenger ID,classtype,andpriceassociatedwiththatbooking.Fig1.E.3showstheresultthat newFDsareinBCNF.

F.6.UniversityEnrolmentSystem:

-Giventherelation:

Enrollments(StudentID,CourseCode,Semester,Grade,InstructorID,CourseName, CourseCredits,Department) -Functionaldependenciesare:

StudentID,CourseCode,Semester→Grade,InstructorID

CourseCode→CourseName,CourseCredits,Department

InstructorID,CourseCode→Department

InstructorID→Department

PreviousFunctionalDependencies

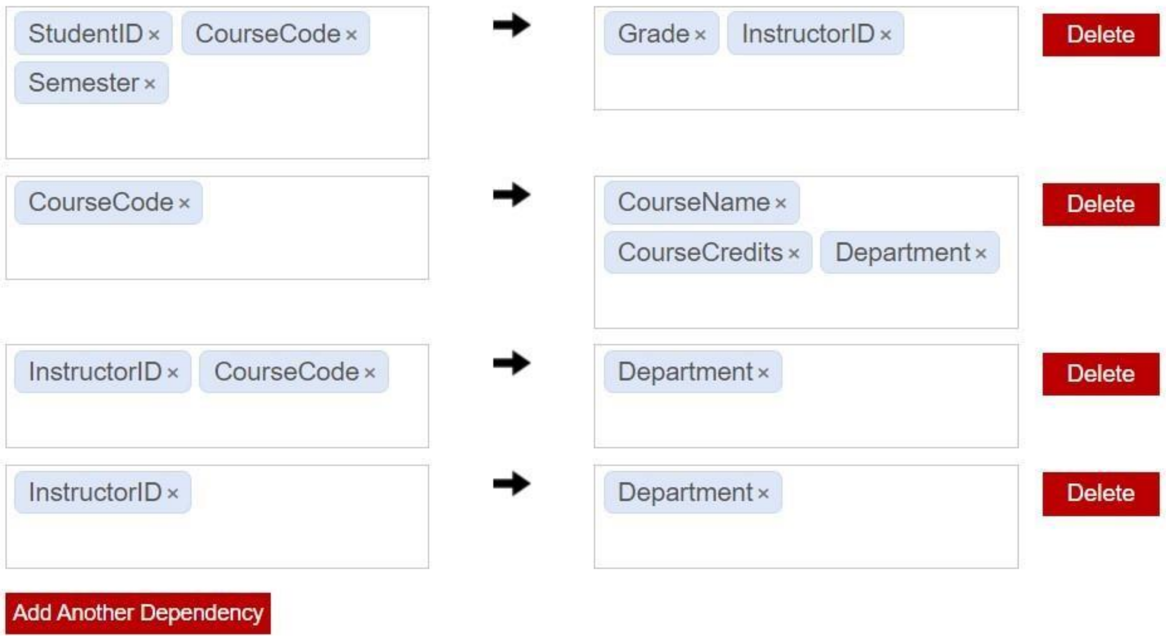
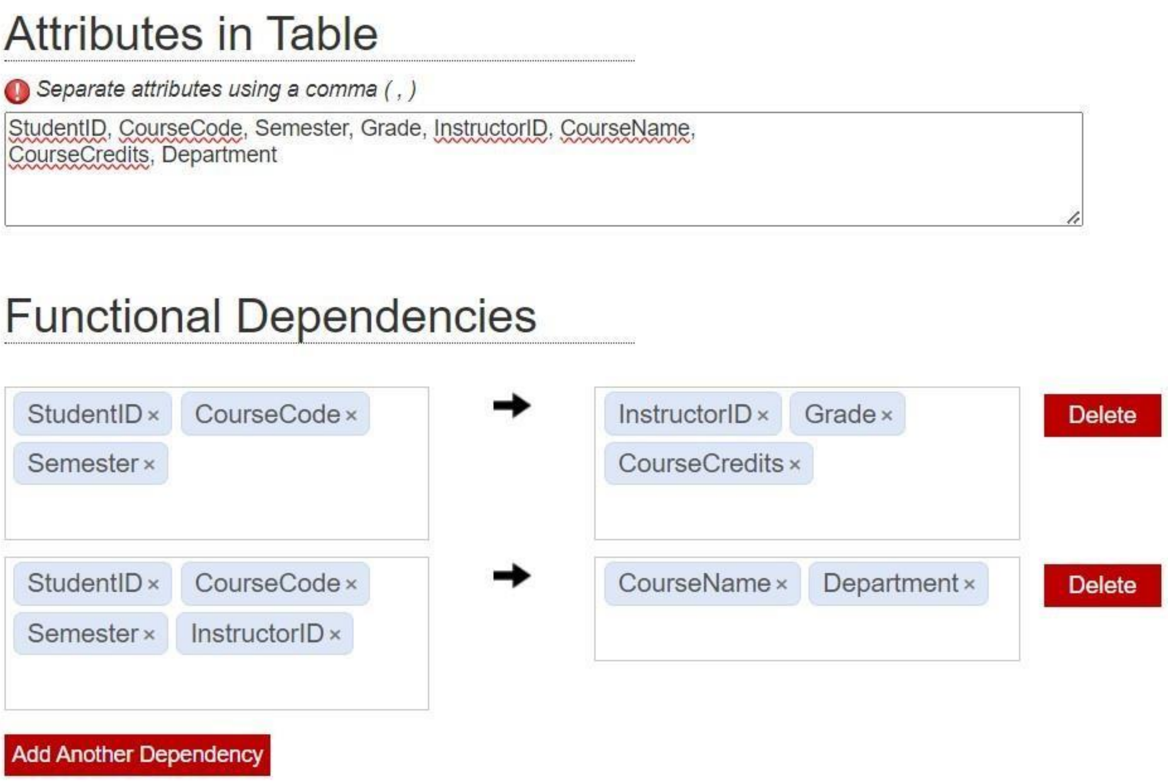

StudentID × CourseCode × Semester ×	→	Grade × InstructorID ×	Delete
CourseCode ×	→	CourseName × CourseCredits × Department ×	Delete
InstructorID × CourseCode ×	→	Department ×	Delete
InstructorID ×	→	Department ×	Delete

Fig1.F.1

ModifiedDependencies


Attributes in Table

Separate attributes using a comma (,)

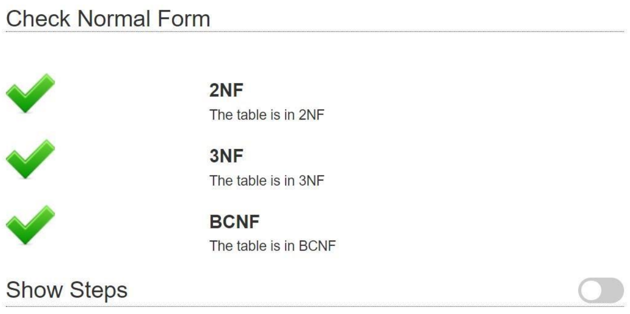
StudentID, CourseCode, Semester, Grade, InstructorID, CourseName, CourseCredits, Department

Functional Dependencies

StudentID × CourseCode × Semester × → InstructorID × Grade × CourseCredits ×

StudentID × CourseCode × Semester × InstructorID × → CourseName × Department ×

Fig1.F.2 Result


Check Normal Form

2NF
The table is in 2NF

3NF
The table is in 3NF

BCNF
The table is in BCNF

Fig1.F.3

Fig1.F.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.F.2 showsnewFunctionalDependencieswhichshowsthatforagivenstudent,a specificcourseinaparticularsemesteruniquelydeterminesthegradereceivedby thestudent,theinstructorteachingthecourse,andthenumberofcreditsassociated withthecourse. Itmeansthatforagivenstudenttakingaspecificcourseinaparticularsemester withaparticularinstructor,thereisonlyonedepartmenttowhichthecoursebelongs andonespecificnameforthecourse.Fig1.F.3showstheresultthatnewFDsarein BCNF.

G.MusicStreamingPlatform:

-Fortherelation:

UserPlays(UserID,SongID,Date,ArtistName,Album,Genre,PlayCount, SubscriptionType)

-Functionaldependenciesare:

UserID,SongID,Date→PlayCount

SongID→ArtistName,Album,Genre

UserID→SubscriptionType

ArtistName,Album→Genre

PreviousFunctionalDependencies

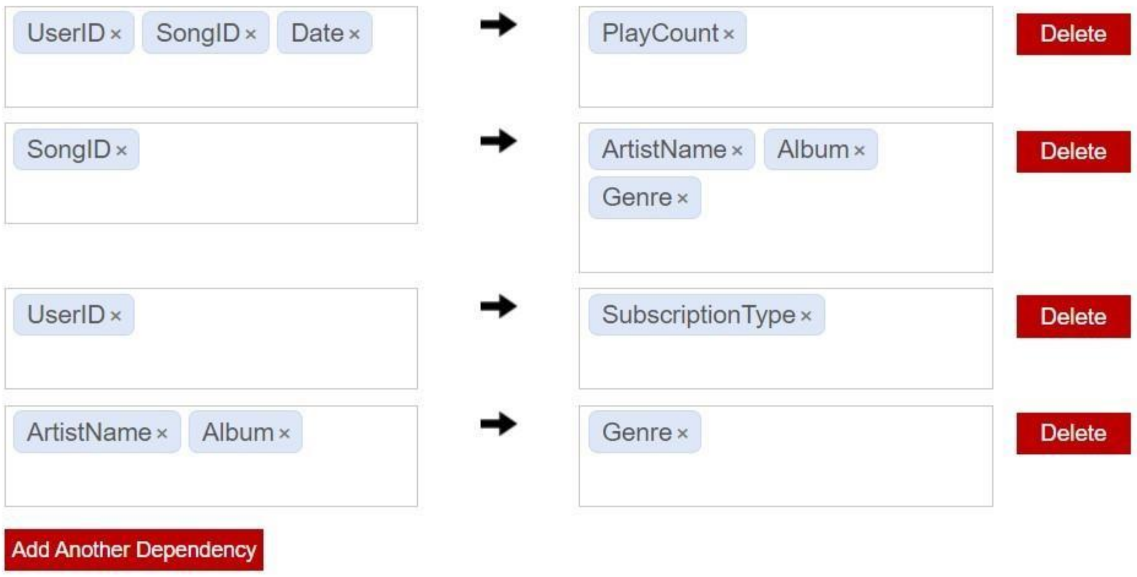
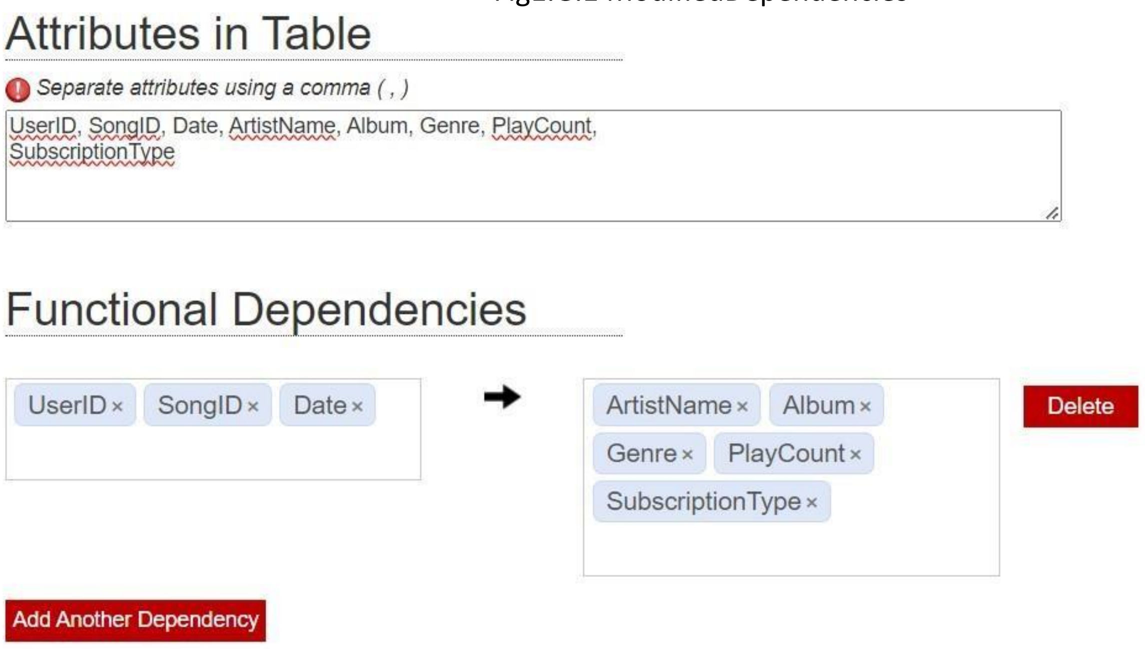

UserID × SongID × Date × → PlayCount × Delete
SongID × → ArtistName × Album × Genre × Delete
UserID × → SubscriptionType × Delete
ArtistName × Album × → Genre × Delete
Add Another Dependency

Fig1.G.1 ModifiedDependencies



Attributes in Table

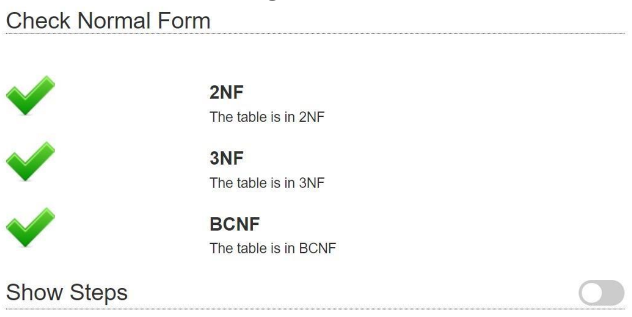
Separate attributes using a comma (,)

UserID, SongID, Date, ArtistName, Album, Genre, PlayCount, SubscriptionType

Functional Dependencies

UserID × SongID × Date × → ArtistName × Album × Genre × PlayCount × SubscriptionType × Delete
Add Another Dependency

Fig1.G.2 Result



Check Normal Form

✓ **2NF** The table is in 2NF

✓ **3NF** The table is in 3NF

✓ **BCNF** The table is in BCNF

Show Steps

Fig1.G.3

Fig1.G.1showspreviousFunctionalDependencieswhicharenotinBCNF.Fig1.G.2 showsnewFunctionalDependencieswhichshowsthatforagivenuser,listeningto aspecificsongonaparticulardateuniquelydeterminesvariousattributesrelatedto thatlisteningevent,suchashowmanytimesthesongwasplayed(PlayCount),the typeofsubscriptiontheuserhas(SubscriptionType),thenameoftheartist,the album,andthegenreofthesong.Fig1.G.3showstheresultthatnewFDsarein BCNF.

H.RealEstateSystem:

-Fortherelation:

PropertyListings(PropertyID,OwnerID,AgentID,Price,Location,HouseType, NumberOfRooms,AgentName,CommissionRate) -Functionaldependenciesare:

PropertyID→Price,Location,HouseType,NumberOfRooms,OwnerID,AgentID

AgentID→AgentName,CommissionRate HouseType→NumberOfRooms

PreviousFunctionalDependencies

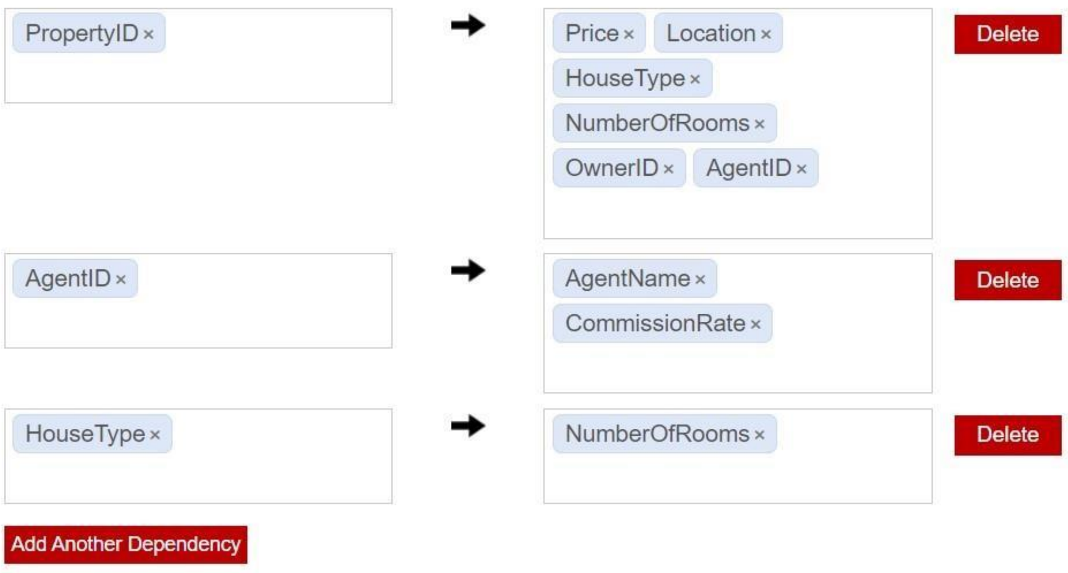
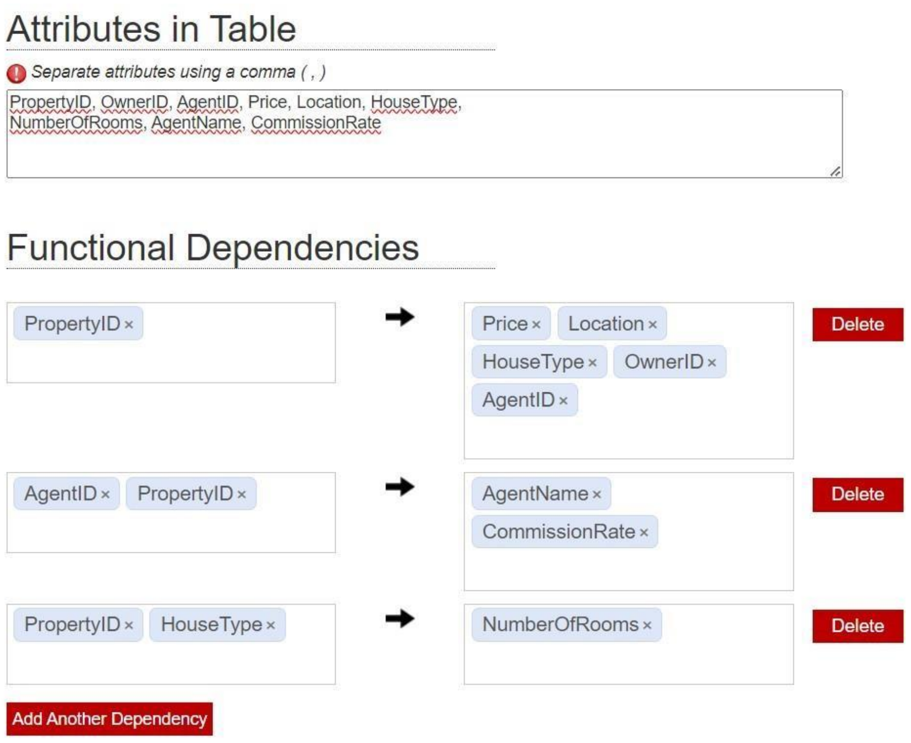

PropertyID ×
Price × Location × HouseType × NumberOfRooms × OwnerID × AgentID ×
Delete
AgentID ×
AgentName × CommissionRate ×
Delete
HouseType ×
NumberOfRooms ×
Delete
Add Another Dependency

Fig1.H.1 ModifiedDependencies



Attributes in Table

Separate attributes using a comma (,)

```
PropertyID, OwnerID, AgentID, Price, Location, HouseType,
NumberOfRooms, AgentName, CommissionRate
```

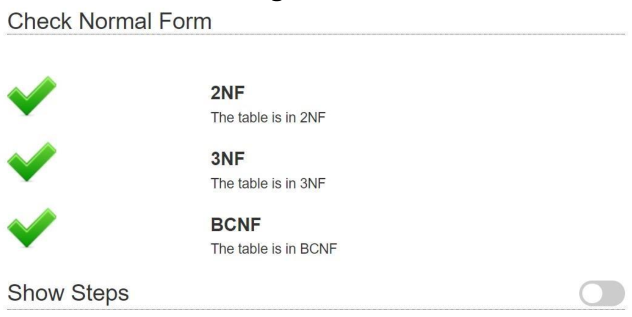
Functional Dependencies

PropertyID → Price, Location, HouseType, OwnerID, AgentID

AgentID, PropertyID → AgentName, CommissionRate

PropertyID, HouseType → NumberOfRooms

Fig1.H.2 Result



Check Normal Form

2NF
The table is in 2NF

3NF
The table is in 3NF

BCNF
The table is in BCNF

Fig1.H.3

Fig1.H.1showspreviousFunctionalDependencieswhic harenotinBCNF.Fig1.H.2 showsnewFunctionalDependencieswhichshowsthat eachpropertyinthetableis uniquelyidentifiedbyitsPropertyID,andforeachPrope rtyID,thereisafixedprice, location,housetype,ownerID,andagentIDassociated withit.

Itmeansthateachagentassignedtoaspecificpropertyisuniquelyidentifiedby theirAgentID,andforeachcombinationofAgentIDandPropertyID,thereisafixed namefortheagentandafixedcommissionrateassociatedwiththatagent's involvementinthatpropertytransaction.

Itmeansthatthenumberofroomsinapropertyisuniquelydeterminedbythe combinationofitsPropertyIDandHouseType.Fig1.H.3showstheresultthatnew FDsareinBCNF.

Que2DesignaBCNFNormalizedDatabaseandverifyusingGriffithTool.

AnsDatabaseisFlightReservationSystem.

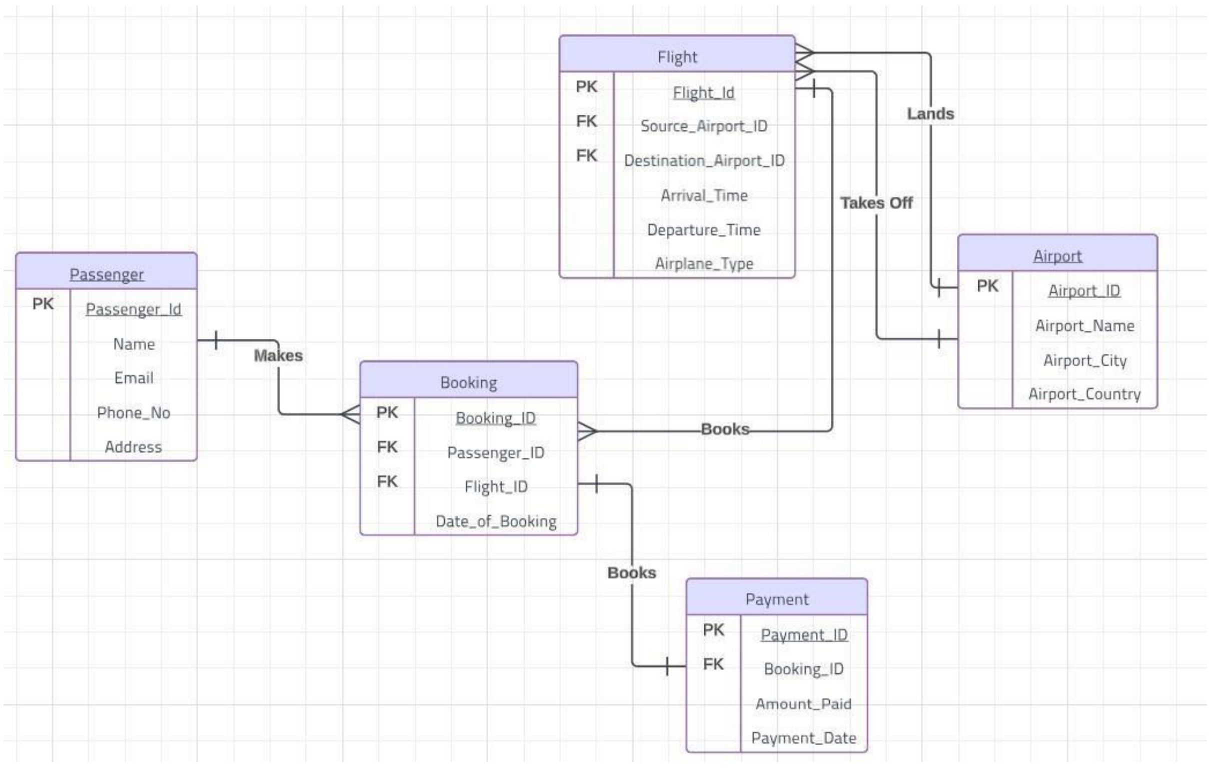


Fig2.1

Fig2.1showsthedesignofairlinereservationsystemdatabase.

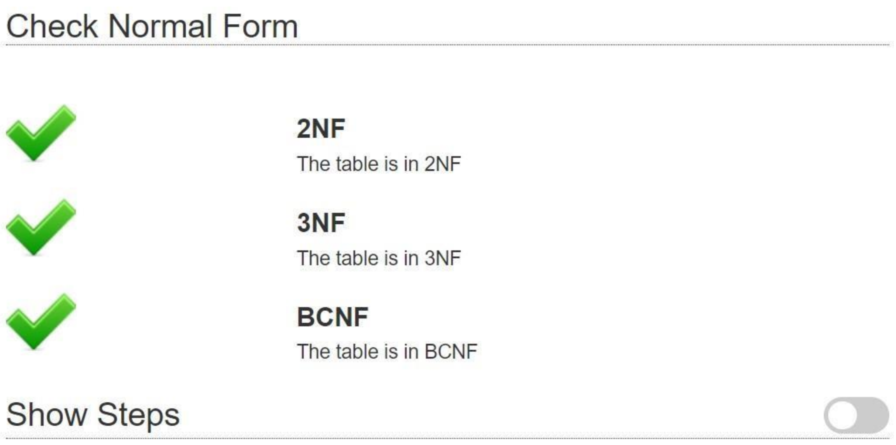
FunctionalDependenciesare:

FlightsTable:

- Flight_ID->Source_Airport_ID
- Flight_ID->Destination_Airport_ID
- Flight_ID->Departure_Time
- Flight_ID->Arrival_Time
- Flight_ID->Airplane_Type AirportsTable: Airport_Code->Airport_Name
- Airport_Code->Airport_City
- Airport_Code->Airport_Country PassengerTable:
- Customer_ID->Name
- Customer_ID->Email
- Customer_ID->Phone_No
- Customer_ID->Address BookingsTable:
- Booking_ID->Flight_ID
- Booking_ID->Passenger_ID
- Booking_ID->Date_of_Booking PaymentsTable:
- Payment_ID->Booking_ID
- Payment_ID->Amount_Paid

- Payment_ID->Payment_Date

VerificationUsingGriffithTool


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF

Fig2.2

Result

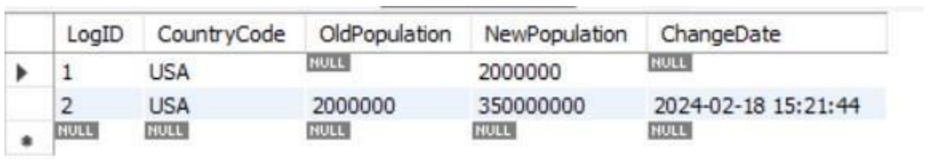
Fig2.2showsthatEachTableisinBCNF.

Practical-3

Aim:-CreateProcedures,TriggersandCursors

Que1WriteastoredprocedurenamedUpdateCountryPopulationthat updatesthepopulationofagivencountrybasedonaprovidedcountry codeandnewpopulationvalue.Additionally,theprocedureshouldlog theoldandnewpopulationvaluestoapopulation_change_logtable. Ans

```
DELIMITER//
CREATEPROCEDUREUpdateCountryPopulation(INCountryCodeCHAR(3),IN
NewPopulationINT)
BEGIN
   DECLAREOldPopulationINT;

   --Gettheoldpopulation
   SELECTPopulationINTOOldPopulation
   FROMcountry
   WHERECode=CountryCode;

   --Updatethepopulation
   UPDATEcountry
   SETPopulation=NewPopulation WHERECode=CountryCode;

   --Logthepopulationchange
   INSERTINTOpopulation_change_log(CountryCode,OldPopulation,
NewPopulation,ChangeDate)
   VALUES(CountryCode,OldPopulation,NewPopulation,NOW());--NOW()isused
forthecurrenttimestampinMySQL
END//
DELIMITER;

CALLUpdateCountryPopulation('USA',350000000);
```



	LogID	CountryCode	OldPopulation	NewPopulation	ChangeDate
▶	1	USA	NULL	2000000	NULL
	2	USA	2000000	350000000	2024-02-18 15:21:44
*	NULL	NULL	NULL	NULL	NULL

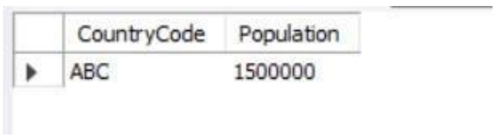
Fig3.1

Fig3.1showspopulation_change_logtablewhichhasoldpopulation,new populationanddateofchange.

Que2Developatriggernamedafter_country_insertthatchecksifthe insertedcountry'spopulationexceeds1million.Ifitdoes,inserta recordintoahigh_population_countriestable.

Ans

```
CREATETRIGGERafter_country_insert
AFTERINSERTONcountry
FOREACHROW
BEGIN
   DECLARECountryPopulationINT;

   --Getthepopulationoftheinsertedcountry
   SELECTPopulationINTOCountryPopulation
   FROMcountry
   WHERECode=NEW.Code;

   --Checkifpopulationexceeds1million
   IFCountryPopulation>1000000THEN
      --Insertintohigh_population_countriestable
      INSERTINTOhigh_population_countries(CountryCode,Population)
      VALUES(NEW.Code,CountryPopulation);
ENDIF; END//
DELIMITER;

INSERTINTOcountry(Code,Population)VALUES('ABC',1500000);

select*fromhigh_population_countries;
```



	CountryCode	Population
▶	ABC	1500000

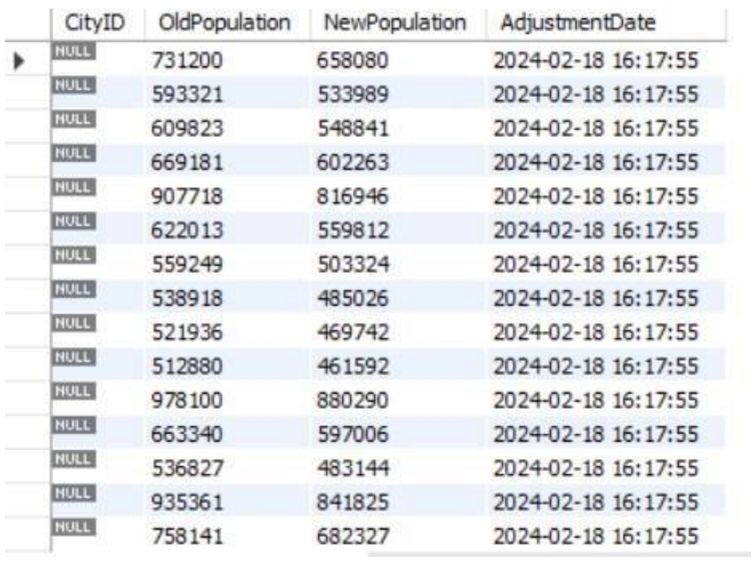
Fig3.2

Fig3.2showshigh_population_countriestablewithcountrycodeandpopulation.

Que3DevelopaprocedureAdjustCityPopulationsusingacursorthat decreasesthepopulationby10%forallcitiesinagivencountrycode, providedthecurrentpopulationisbetween500,000and1million. Additionally,logthesechangestoacity_population_adjustmentstable withcityID,oldpopulation,andnewpopulation.

Ans

```
DELIMITER//
          CREATEPROCEDUREAdjustCityPopulations(INCountryCodeCHAR(3))
BEGIN
   DECLAREdoneINTDEFAULTFALSE;
   DECLARECityIDINT;
   DECLAREOldPopulationINT;
   DECLARENewPopulationINT;
   --Declarecursor
   DECLAREcity_cursorCURSORFOR
   SELECTCityID,Population
   FROMcity
   WHERECountryCode=CountryCode
   ANDPopulationBETWEEN500000AND1000000;

   --Declarehandlerfornomorerows
   DECLARECONTINUEHANDLERFORNOTFOUNDSETdone=TRUE;

   --Openthecursor
   OPENcity_cursor;

   --Startloopingthroughthecursor
   adjust_loop:LOOP --Fetchtherow
      FETCHcity_cursorINTOCityID,OldPopulation;

      --Checkifnomorerows IFdoneTHEN
         LEAVEadjust_loop;
      ENDIF;

      --Calculatenewpopulation(decreaseby10%)
      SETNewPopulation=ROUND(OldPopulation*0.9,0);

      --Updatecitypopulation
      UPDATEcity
      SETPopulation=NewPopulation WHERECityID=CityID;

      --Logpopulationadjustment
      INSERTINTOcity_population_adjustment(CityID,OldPopulation,
NewPopulation,AdjustmentDate)
      VALUES(CityID,OldPopulation,NewPopulation,NOW()); ENDLOOPadjust_loop;

   --Closethecursor
   CLOSEcity_cursor;
END//
DELIMITER;
CALLAdjustCityPopulations('USA'); select*fromcity_population_adjustment;
```



CityID	OldPopulation	NewPopulation	AdjustmentDate
NULL	731200	658080	2024-02-18 16:17:55
NULL	593321	533989	2024-02-18 16:17:55
NULL	609823	548841	2024-02-18 16:17:55
NULL	669181	602263	2024-02-18 16:17:55
NULL	907718	816946	2024-02-18 16:17:55
NULL	622013	559812	2024-02-18 16:17:55
NULL	559249	503324	2024-02-18 16:17:55
NULL	538918	485026	2024-02-18 16:17:55
NULL	521936	469742	2024-02-18 16:17:55
NULL	512880	461592	2024-02-18 16:17:55
NULL	978100	880290	2024-02-18 16:17:55
NULL	663340	597006	2024-02-18 16:17:55
NULL	536827	483144	2024-02-18 16:17:55
NULL	935361	841825	2024-02-18 16:17:55
NULL	758141	682327	2024-02-18 16:17:55

Fig3.3

Fig3.3showscity_population_adjustmenttablewhichrecordthepopulation statisticsanddateofchange.

Practical-4

Aim:-Writeprogramstoimplementandunderstandusageof Datamarts.

Question1:Designadatamartforabanktostorethecredithistoryof customersinabank.Usethiscreditprofilingtoprocessfutureloan applications.(Suggestivetables:CustomerProfile,accounts,loans, creditcards,paymenthistorytable,inquiries,Collections,CreditScore History). Ans

createdatabasebank;

createtablecustomer_profile(customer_idintprimarykey,first_name varchar(25),last_namevarchar(25),d_o_bdate,addressvarchar(50),phone_no int,emailvarchar(25),incomeint);

createtableaccounts(account_idintprimarykey,customer_idint,accounttype varchar(25),dateofopendate,accountstatusvarchar(25),foreignkey(customer_id) referencescustomer_profile(customer_id),balanceint);

createtableloans(loan_idintprimarykey,customer_idint,loantype varchar(25),loanamountint,termint,interest_ratedecimal(4,2),loanstatus varchar(25),foreignkey(customer_id)referencescustomer_profile(customer_id));

createtablecreditcards(card_idintprimarykey,customer_idint,cardtype varchar(25),creditlimitdecimal(10,2),cardissuedatedate,foreignkey(customer_id) referencescustomer_profile(customer_id),currentbalancedecimal(10,2));

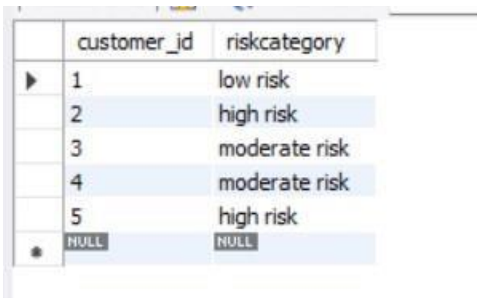
createtablepaymenthistory(payment_idintprimarykey,customer_idint,account_id int,paymentamountdecimal(10,2),paymentdatedate,foreignkey(customer_id) referencescustomer_profile(customer_id),foreignkey(account_id)references accounts(account_id));

createtableinquiries(inquiry_idintprimarykey,customer_idint,inquirydate date,inquirytypevarchar(25),foreignkey(customer_id)references customer_profile(customer_id));

createtablecollections(collection_idintprimarykey,customer_idint,collectiondate date,collectiontypevarchar(25),amountint,foreignkey(customer_id)references customer_profile(customer_id));

createtablecredit_score_history(creditscore_idintprimarykey,customer_id int,creditscoreint,scoredatedate,foreignkey(customer_id)references customer_profile(customer_id)); --DATAMART:

createtablecustomerrisk(customer_idintprimarykey,riskcategoryvar
char(25));
insertintocustomerrisk(customer_id,riskcategory)selectc.customer_id,case
whenc.income>75000andsum(a.balance)>100000then'lowrisk'
whenc.income>50000andsum(a.balance)>60000then'moderaterisk' else'highrisk'
endasriskcategory
fromcustomer_profilecjoinaccountsaonc.customer_id=a.customer_idgroupby c.customer_id;

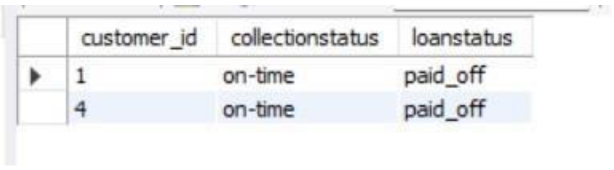


customer_id	riskcategory
1	low risk
2	high risk
3	moderate risk
4	moderate risk
5	high risk
NULL	NULL

Fig4.1

InFig4.1,itshowsthatitdividesthecustomersintodifferentriskcategorybaseon
incomeandbalanceofcustomers.

createtableloanassessmentasselectc.customer_idas
customer_id,c.collectionstatusascollectionstatus,l.loanstatusasloanstatusfrom
collectionscjoinloanslonc.customer_id=l.customer_idwherecollectionstatus='ontime'andloanst
atus='paid_off';

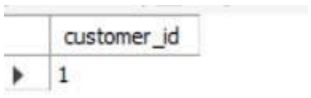


customer_id	collectionstatus	loanstatus
1	on-time	paid_off
4	on-time	paid_off

Fig4.2

InFig4.2itshowstheresultofcustomerswhoseloanstatusispaidoffand collectionstatusisontime.

createtableloanpassasselectl.customer_idfromloanassessmentljoin
customerriskconl.customer_id=c.customer_idjoincredit_score_historychon
ch.customer_id=c.customer_idwherec.riskcategory='lowrisk'and ch.creditscore>750;



customer_id
1

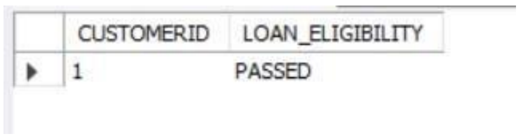
Fig4.3

InFig4.3itshowsthecustomerswhichhaslowriskcategoryhasloanstatusas
paidoffandontimeandcreditscoregreaterthan750.

CREATEPROCEDURELOAN_PASS_RESULT(INCUSTOMERIDINT) BEGIN
DECLAREMESSAGE_TEXTVARCHAR(50);
IFEXISTS(
        SELECT1FROMloanpass

```
        WHEREcustomer_id=CUSTOMERID
    )THEN
        SELECTCUSTOMERID,'PASSED'ASLOAN_ELIGIBILITY;
ELSE
        SELECTCUSTOMERID,'REJECTED'ASLOAN_ELIGIBILITY;
ENDIF;
END//
DELIMITER; callLOAN_PASS_RESULT(1);
```

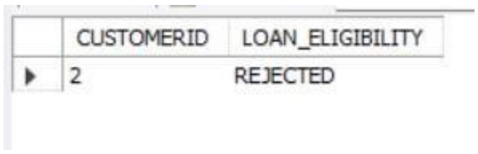
Output1



```
   CUSTOMERID   LOAN_ELIGIBILITY
▶  1            PASSED
```

Fig4.4

callLOAN_PASS_RESULT(2); Output2



```
   CUSTOMERID   LOAN_ELIGIBILITY
▶  2            REJECTED
```

Fig4.5

RESULT:Successfullyimplementedandlearnttheusageof Datamarts.

PRACTICAL#5

Objective: Feature Selection and Variable Filtering.

Question#:

A) Select a dataset that has a minimum of 150 features.

B) Apply 3 Feature Selection Techniques

C) For each feature selection technique apply 3 machine learning models on it.

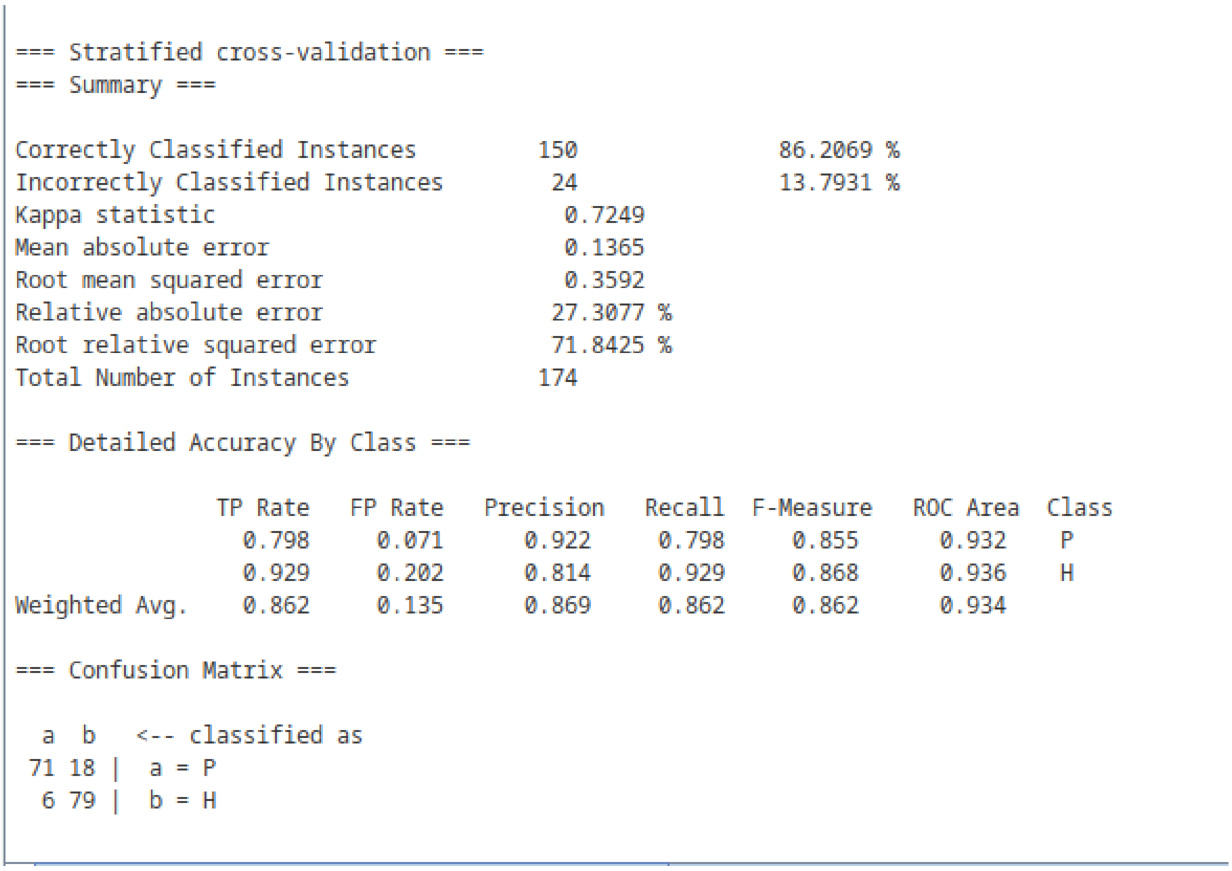
D) Compare the results.

TOOL USED: Weka

Feature Selection Technique->Gain Ratio->The gain ratio is a metric in decision trees that balances the information gain with the intrinsic information of attributes, helping to select the best attribute for splitting nodes.

No. of selelcted attribute-> 20

Algorithm: Naive Bayes->Probabilistic classification algorithm based on Bayes' theorem with an assumption of independence between features



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         150               86.2069 %
Incorrectly Classified Instances        24               13.7931 %
Kappa statistic                          0.7249
Mean absolute error                      0.1365
Root mean squared error                  0.3592
Relative absolute error                 27.3077 %
Root relative squared error             71.8425 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall   F-Measure   ROC Area  Class
                 0.798     0.071     0.922     0.798     0.855      0.932    P
                 0.929     0.202     0.814     0.929     0.868      0.936    H
Weighted Avg.    0.862     0.135     0.869     0.862     0.862      0.934

=== Confusion Matrix ===

  a  b   <-- classified as
 71 18 |  a = P
  6 79 |  b = H
```

Fig 5.1 Naive Bayes with 20 attributes

Algorithm:Random tree->It works by building multiple decision trees during training, where each tree is trained on a random subset of the training data and a random subset of the features.The random trees vote on the final classification or regression output, and the most popular outcome is chosen.Random Trees help reduce overfitting and improve accuracy, especially when dealing with noisy or high-dimensional data

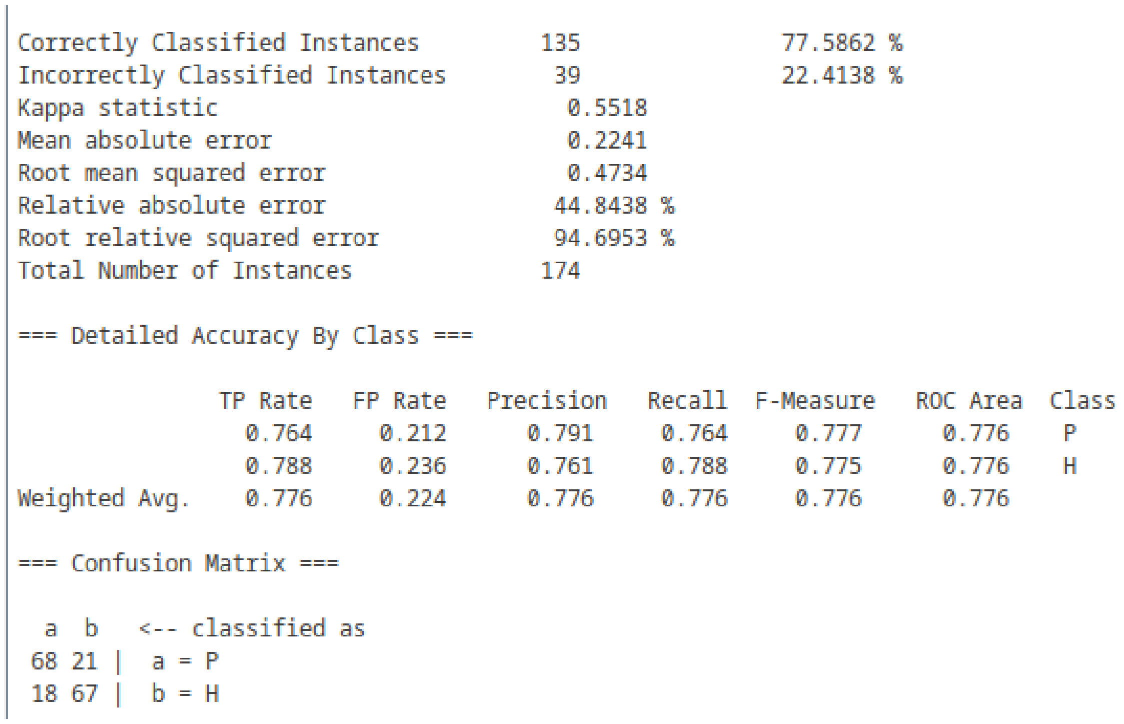
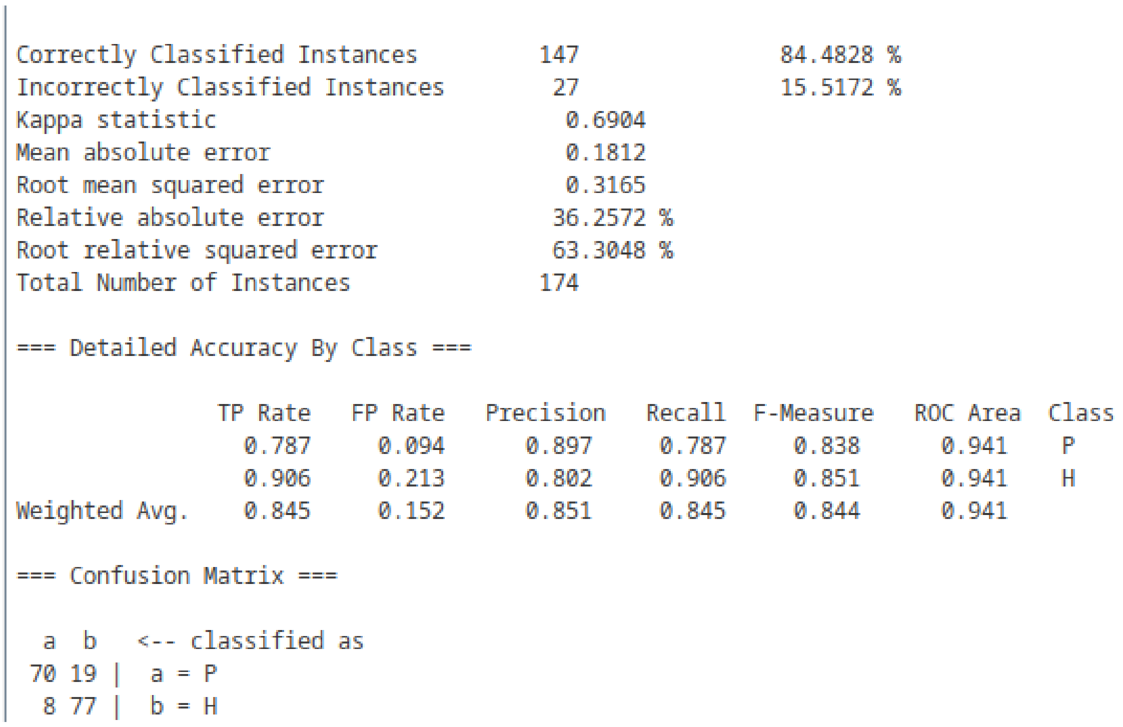

```
Correctly Classified Instances         135               77.5862 %
Incorrectly Classified Instances        39               22.4138 %
Kappa statistic                          0.5518
Mean absolute error                      0.2241
Root mean squared error                  0.4734
Relative absolute error                 44.8438 %
Root relative squared error             94.6953 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.764     0.212     0.791     0.764     0.777      0.776     P
                 0.788     0.236     0.761     0.788     0.775      0.776     H
Weighted Avg.    0.776     0.224     0.776     0.776     0.776      0.776

=== Confusion Matrix ===

  a  b   <-- classified as
 68 21 |  a = P
 18 67 |  b = H
```

Fig 5.2 Random Tree with 20 attributes

Algorithm:AdaBoost->It works by combining multiple weak learners (typically decision trees) to create a strong learner.t begins by assigning equal weights to all training samples. Then, it iteratively trains weak learners, focusing more on incorrectly classified samples in each iteration.The predictions of weak learners are combined through weighted voting, where more accurate learners have higher weights. This process continues until a predetermined number of iterations is reached or until perfect predictions are achieved

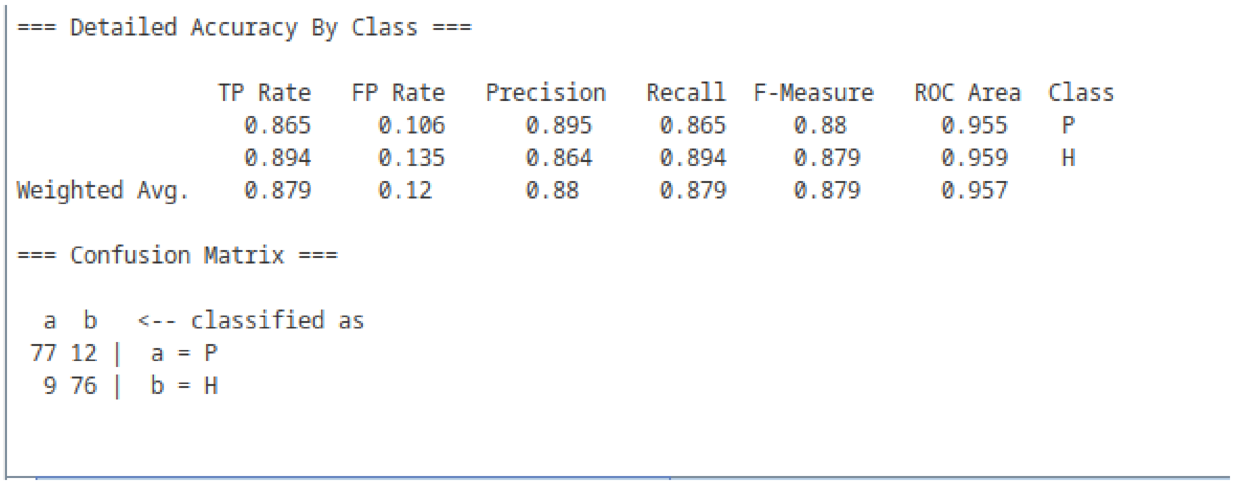


```
Correctly Classified Instances         147               84.4828 %
Incorrectly Classified Instances        27               15.5172 %
Kappa statistic                          0.6904
Mean absolute error                      0.1812
Root mean squared error                  0.3165
Relative absolute error                 36.2572 %
Root relative squared error             63.3048 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.787     0.094      0.897     0.787     0.838      0.941    P
                 0.906     0.213      0.802     0.906     0.851      0.941    H
Weighted Avg.    0.845     0.152      0.851     0.845     0.844      0.941

=== Confusion Matrix ===

  a  b   <-- classified as
 70 19 |  a = P
  8 77 |  b = H
```

Fig 5.3 AdaBoost with 20 attributes

Feature Selection Technique->Gain Ratio

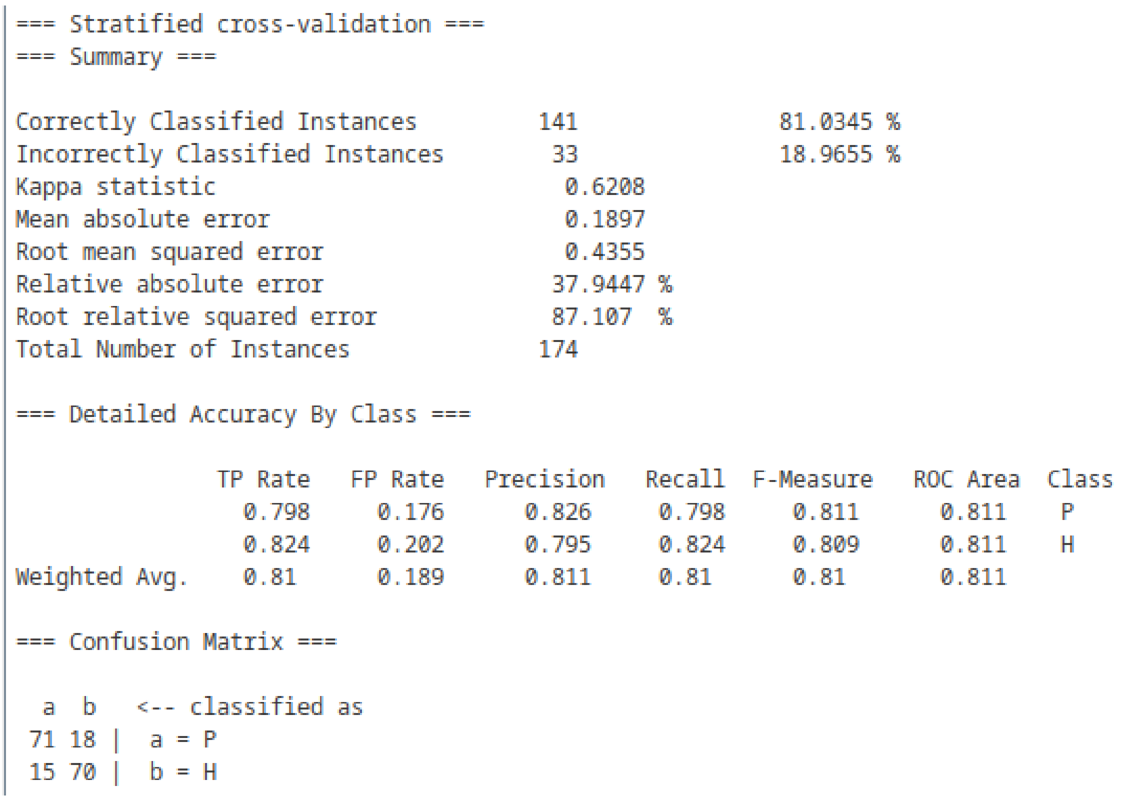
No. of selelcted attribute-> 40
Algorithm: Naive Bayes



```
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.865     0.106      0.895     0.865     0.88       0.955    P
                 0.894     0.135      0.864     0.894     0.879      0.959    H
Weighted Avg.    0.879     0.12       0.88      0.879     0.879      0.957

=== Confusion Matrix ===

  a  b   <-- classified as
 77 12 |  a = P
  9 76 |  b = H
```

Fig 5.4 Naive Bayes with 40 attributes

Algorithm: Random Tree

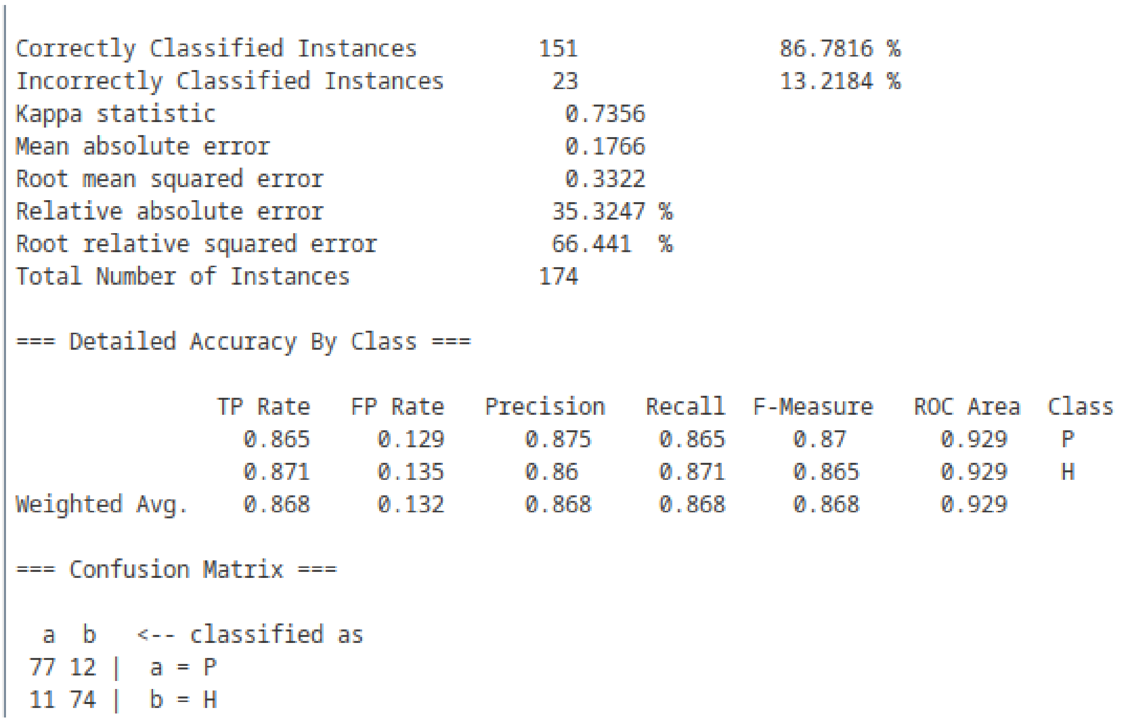


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         141               81.0345 %
Incorrectly Classified Instances        33               18.9655 %
Kappa statistic                          0.6208
Mean absolute error                      0.1897
Root mean squared error                  0.4355
Relative absolute error                 37.9447 %
Root relative squared error             87.107  %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.798     0.176      0.826     0.798     0.811      0.811    P
                 0.824     0.202      0.795     0.824     0.809      0.811    H
Weighted Avg.    0.81      0.189      0.811     0.81      0.81       0.811

=== Confusion Matrix ===

  a  b   <-- classified as
 71 18 |  a = P
 15 70 |  b = H
```

Fig 5.5 Random Tree with 40 attributes

Algorithm: AdaBoost

Fig 5.6

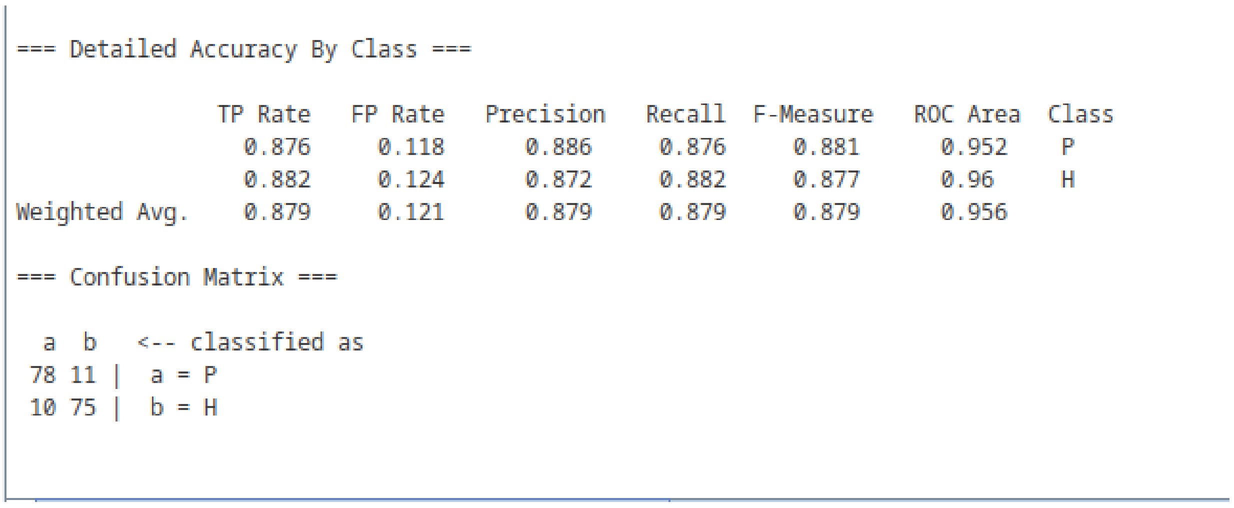


```
Correctly Classified Instances         151               86.7816 %
Incorrectly Classified Instances        23               13.2184 %
Kappa statistic                          0.7356
Mean absolute error                      0.1766
Root mean squared error                  0.3322
Relative absolute error                 35.3247 %
Root relative squared error             66.441  %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.865     0.129      0.875     0.865     0.87       0.929    P
                 0.871     0.135      0.86      0.871     0.865      0.929    H
Weighted Avg.    0.868     0.132      0.868     0.868     0.868      0.929

=== Confusion Matrix ===

  a  b   <-- classified as
 77 12 |  a = P
 11 74 |  b = H
```

Fig 5.6 AdaBoost with 40 attributes

Feature Selection Technique->Gain Ratio

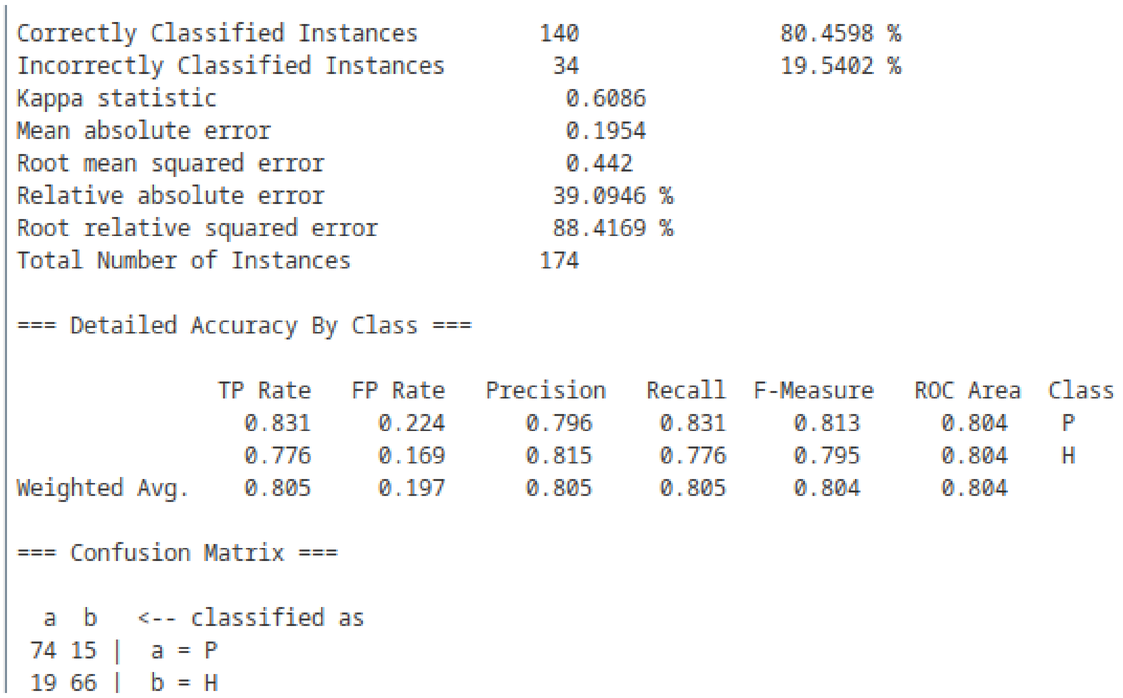
No. of selelcted attribute-> 50
Algorithm: Naive Bayes



```
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.876     0.118      0.886     0.876     0.881      0.952    P
                 0.882     0.124      0.872     0.882     0.877      0.96     H
Weighted Avg.    0.879     0.121      0.879     0.879     0.879      0.956

=== Confusion Matrix ===

  a  b   <-- classified as
 78 11 |  a = P
 10 75 |  b = H
```

Fig 5.7 Naive Bayes with 50 attributes

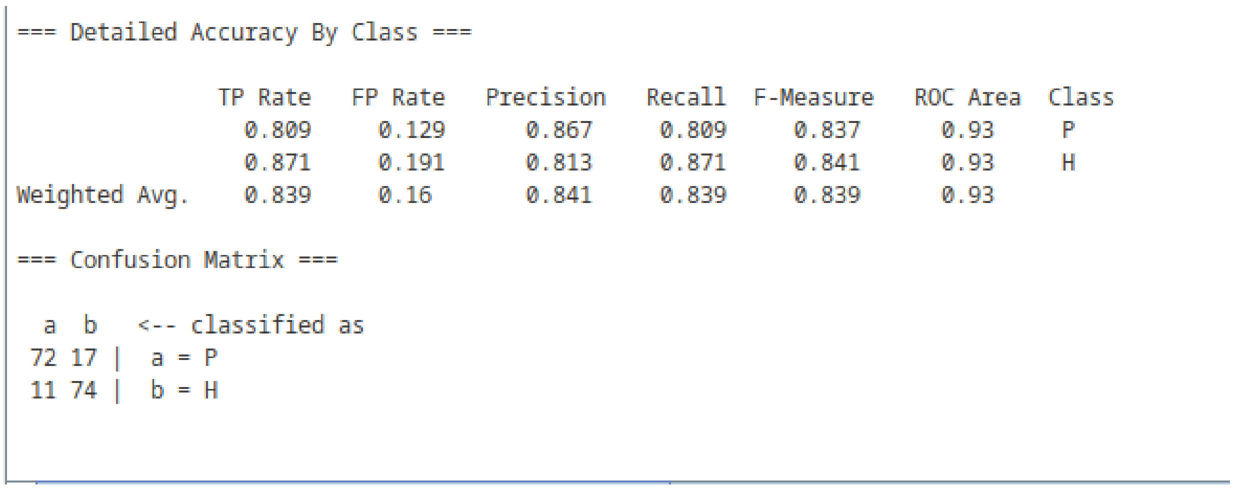
Algorithm: Random Tree



```
Correctly Classified Instances         140               80.4598 %
Incorrectly Classified Instances        34               19.5402 %
Kappa statistic                          0.6086
Mean absolute error                      0.1954
Root mean squared error                  0.442
Relative absolute error                 39.0946 %
Root relative squared error             88.4169 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.831     0.224      0.796     0.831     0.813      0.804    P
                 0.776     0.169      0.815     0.776     0.795      0.804    H
Weighted Avg.    0.805     0.197      0.805     0.805     0.804      0.804

=== Confusion Matrix ===

  a  b   <-- classified as
 74 15 |  a = P
 19 66 |  b = H
```

Fig 5.8 Random Tree with 50 attributes

Algorithm: AdaBoost



```
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.809     0.129      0.867     0.809     0.837      0.93     P
                 0.871     0.191      0.813     0.871     0.841      0.93     H
Weighted Avg.    0.839     0.16       0.841     0.839     0.839      0.93

=== Confusion Matrix ===

  a  b   <-- classified as
 72 17 |  a = P
 11 74 |  b = H
```

Fig 5.9 AdaBoost with 50 attributes

TOOL USED:- ORANGE

Orange is an open-source data visualization, analysis, and machine learning toolkit. It provides a user-friendly interface for data preprocessing, exploration, visualization, and predictive modeling.Orange offers a wide range of machine learning algorithms for classification, regression, clustering, and other tasks. Users can easily compare and evaluate different algorithms using built-in evaluation widgets.

KNN->K-Nearest Neighbors (KNN) is a simple yet effective supervised machine learning algorithm used for both classification and regression tasks. It's based on the idea that similar data points tend to belong to the same class or have similar values.When making predictions for a new data point, KNN calculates the distance between that point and all other points in the training dataset. Common distance metrics include

Euclidean distance, Manhattan distance, or cosine similarity.

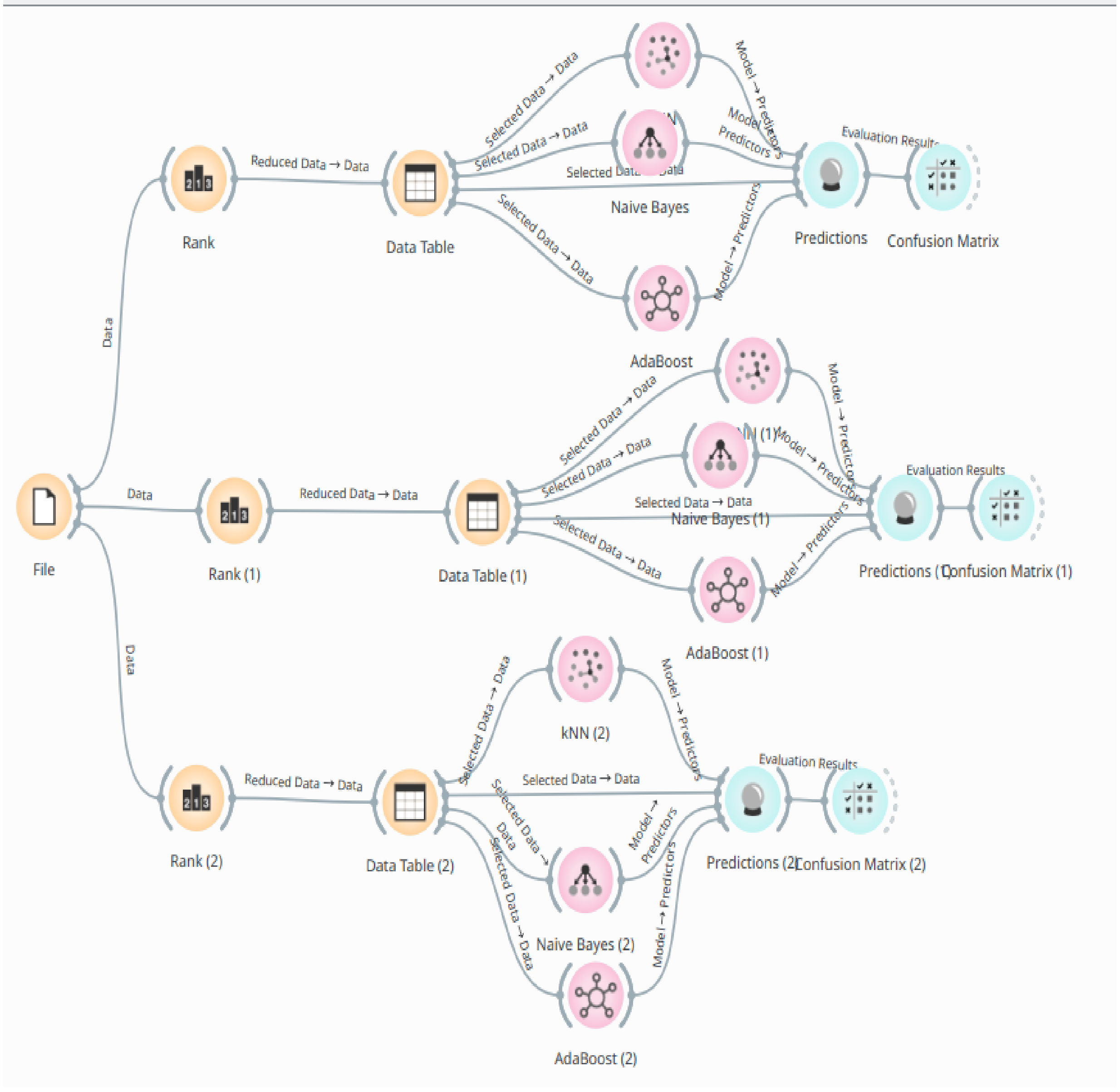


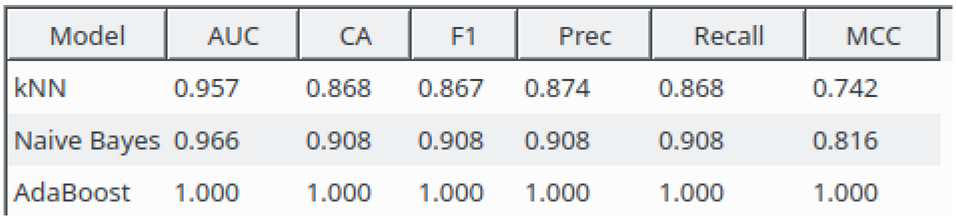
Fig 5.10

Feature Selection Technique->Gain Ratio

No. of selelcted attribute-> 20

No. of

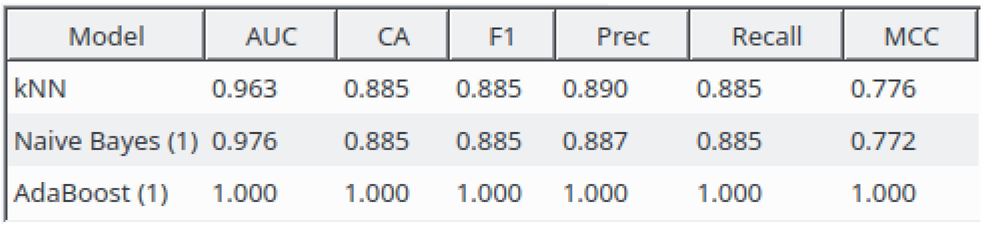
selelcted



Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.957	0.868	0.867	0.874	0.868	0.742
Naive Bayes	0.966	0.908	0.908	0.908	0.908	0.816
AdaBoost	1.000	1.000	1.000	1.000	1.000	1.000

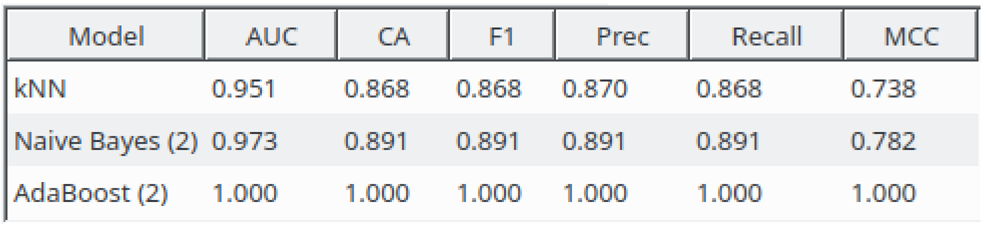
attribute-> 40

No. of



Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.963	0.885	0.885	0.890	0.885	0.776
Naive Bayes (1)	0.976	0.885	0.885	0.887	0.885	0.772
AdaBoost (1)	1.000	1.000	1.000	1.000	1.000	1.000

selelcted attribute-> 50



Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.951	0.868	0.868	0.870	0.868	0.738
Naive Bayes (2)	0.973	0.891	0.891	0.891	0.891	0.782
AdaBoost (2)	1.000	1.000	1.000	1.000	1.000	1.000

PRACTICAL#6

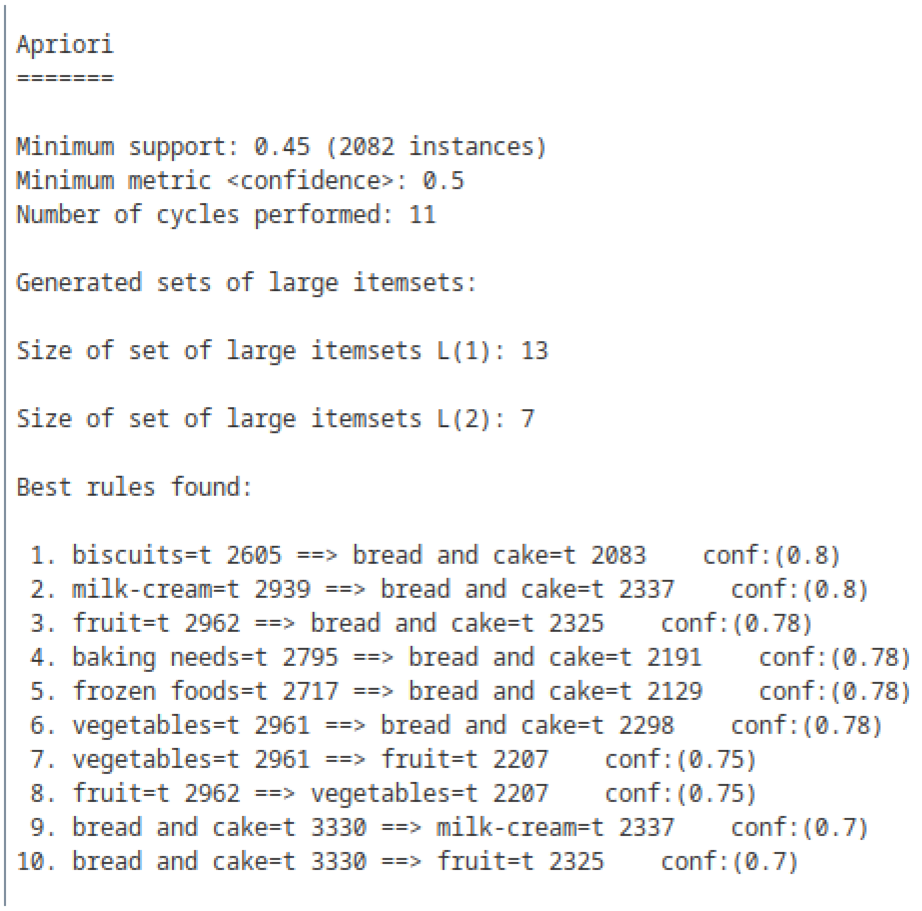
Aim :- Perform Associative Mining In Weka and Orange on large datasets

Theory :- To perform association mining on large datasets, algorithms such as Apriori or FP-growth are employed. These algorithms efficiently extract frequent itemsets by iteratively identifying patterns within transactional data. With the support of these algorithms, associations between items can be discovered, aiding in tasks such as market basket analysis or recommendation systems. Efficient implementation and optimization are crucial for handling the computational complexity posed by large datasets, ensuring scalability and practical applicability in real-world scenarios.

PROCEDURE:

USING WEKA TOOL:

Scenario#1:WITH VALUE OF SUPPORT = 0.3 AND CONFIDENCE = 0.5

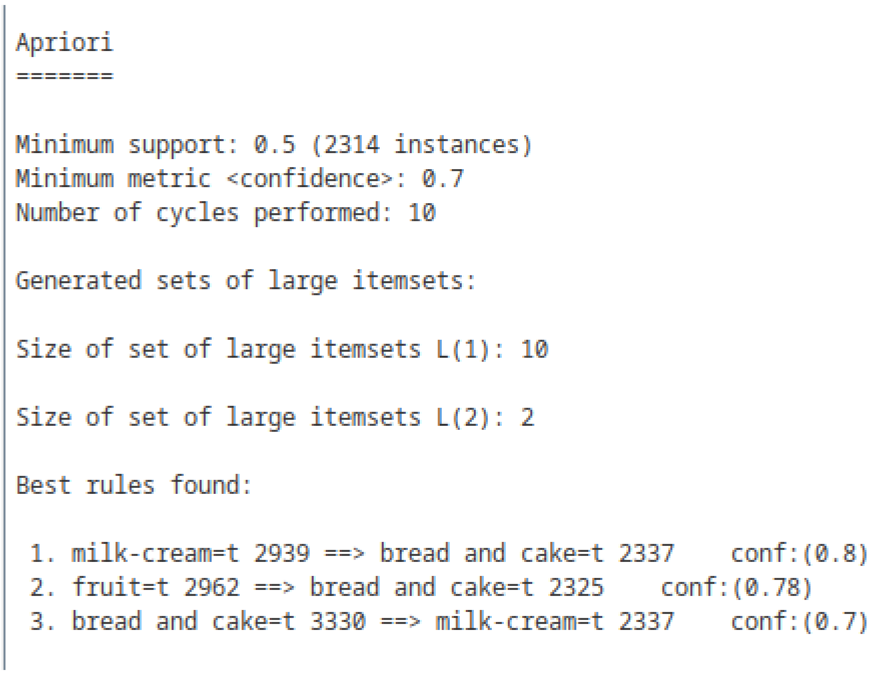


```
Apriori
=======

Minimum support: 0.45 (2082 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 11

Generated sets of large itemsets:

Size of set of large itemsets L(1): 13

Size of set of large itemsets L(2): 7

Best rules found:

 1. biscuits=t 2605 ==> bread and cake=t 2083    conf:(0.8)
 2. milk-cream=t 2939 ==> bread and cake=t 2337    conf:(0.8)
 3. fruit=t 2962 ==> bread and cake=t 2325    conf:(0.78)
 4. baking needs=t 2795 ==> bread and cake=t 2191    conf:(0.78)
 5. frozen foods=t 2717 ==> bread and cake=t 2129    conf:(0.78)
 6. vegetables=t 2961 ==> bread and cake=t 2298    conf:(0.78)
 7. vegetables=t 2961 ==> fruit=t 2207    conf:(0.75)
 8. fruit=t 2962 ==> vegetables=t 2207    conf:(0.75)
 9. bread and cake=t 3330 ==> milk-cream=t 2337    conf:(0.7)
10. bread and cake=t 3330 ==> fruit=t 2325    conf:(0.7)
```

Fig 6.1 he rules found based on Support = 0.3 and Confidence = 0.5

The Apriori method, with a minimum support of 0.3 and a minimum confidence of 0.5 over 11 cycles, produced 2082 instances in Figure 6.1. Large itemset sets were produced by it; L(1) contained 13 sets, while L(2) contained 7. Among the notable rules are those describing associations, such biscuits leading to cake and bread or fruit leading to cake and bread.

Scenario#2:WITH VALUE OF SUPPORT = 0.5 AND CONFIDENCE = 0.7



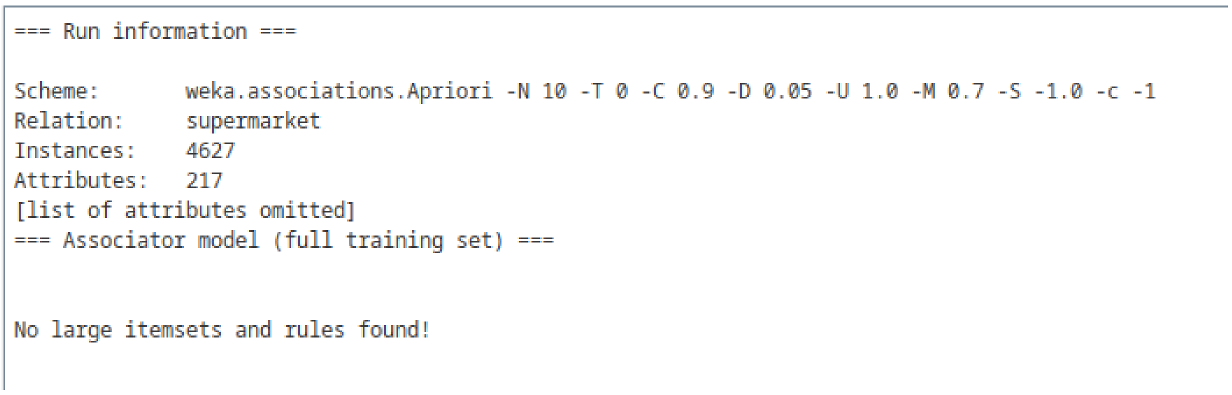
```
Apriori
=======

Minimum support: 0.5 (2314 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 10

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10

Size of set of large itemsets L(2): 2

Best rules found:

 1. milk-cream=t 2939 ==> bread and cake=t 2337    conf:(0.8)
 2. fruit=t 2962 ==> bread and cake=t 2325    conf:(0.78)
 3. bread and cake=t 3330 ==> milk-cream=t 2337    conf:(0.7)
```

Fig 6.2 The rules found based on Support = 0.5 and Confidence = 0.7

The Apriori method, with a minimum support of 0.5 and a minimum confidence of 0.7 over 10 cycles, produced 2314 instances in Figure 6.2. Large itemset sets were produced by it. L(1) contained 10 sets, while L(2) contained 2.

Some Associations are milk-cream to bread and cake or fruit to bread and cake

Scenario#3:WITH VALUE OF SUPPORT = 0.3 AND CONFIDENCE = 0.7



```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.7 -S -1.0 -c -1
Relation:     supermarket
Instances:    4627
Attributes:   217
[list of attributes omitted]
=== Associator model (full training set) ===


No large itemsets and rules found!
```

Fig 6.3 The rules found based on Support = 0.7 and Confidence = 0.9

In Figure 6.3, no rules were found in this iteration of the Apriori method, with a minimum support of 0.7 and a minimum confidence of 0.9 applied.This finding might be explained by the strict confidence and support standards that were established, which might have led to too few examples satisfying these requirements to create meaningful correlations. The lack of rules implies that there might not be frequent itemsets in the dataset that meet the designated confidence and support requirements.

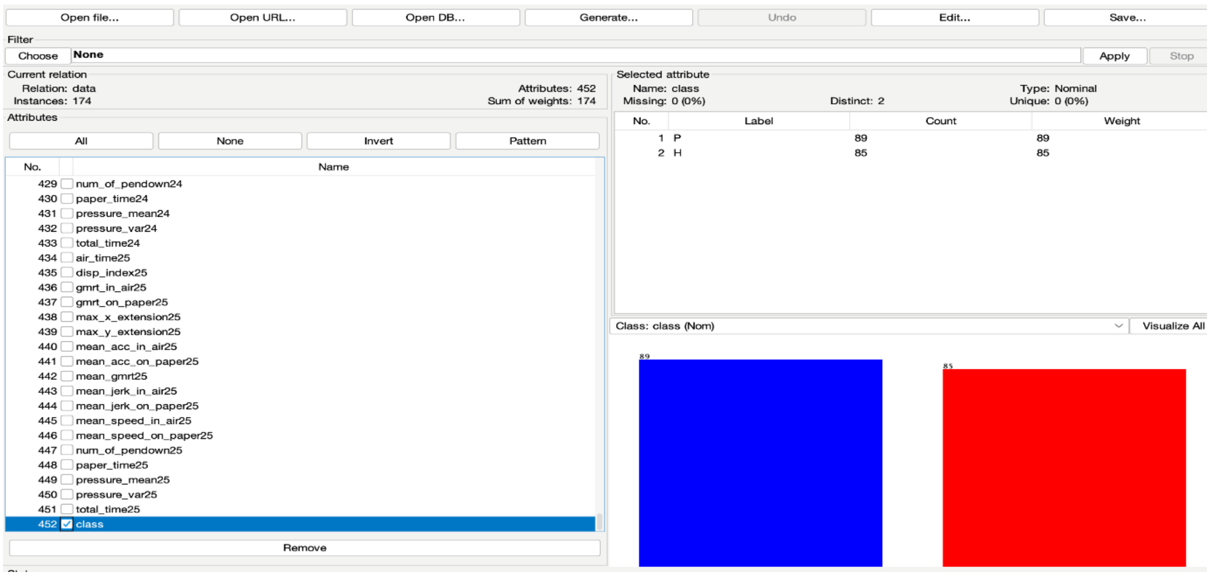
Practical #7

Aim : Perform K-Nearest Neighbour Classification in Weka and Orange

Theory:
K-Nearest Neighbors (KNN) is a simple yet powerful algorithm used for classification and regression tasks in machine learning. It's a non-parametric and instance-based learning algorithm, meaning it doesn't make strong assumptions about the underlying data distribution and it memorizes the entire training dataset.The choice of K (the number of nearest neighbors to consider) is crucial. A smaller K value can lead to more complex decision boundaries, while a larger K value can lead to smoother boundarie

Using : WEKA tool

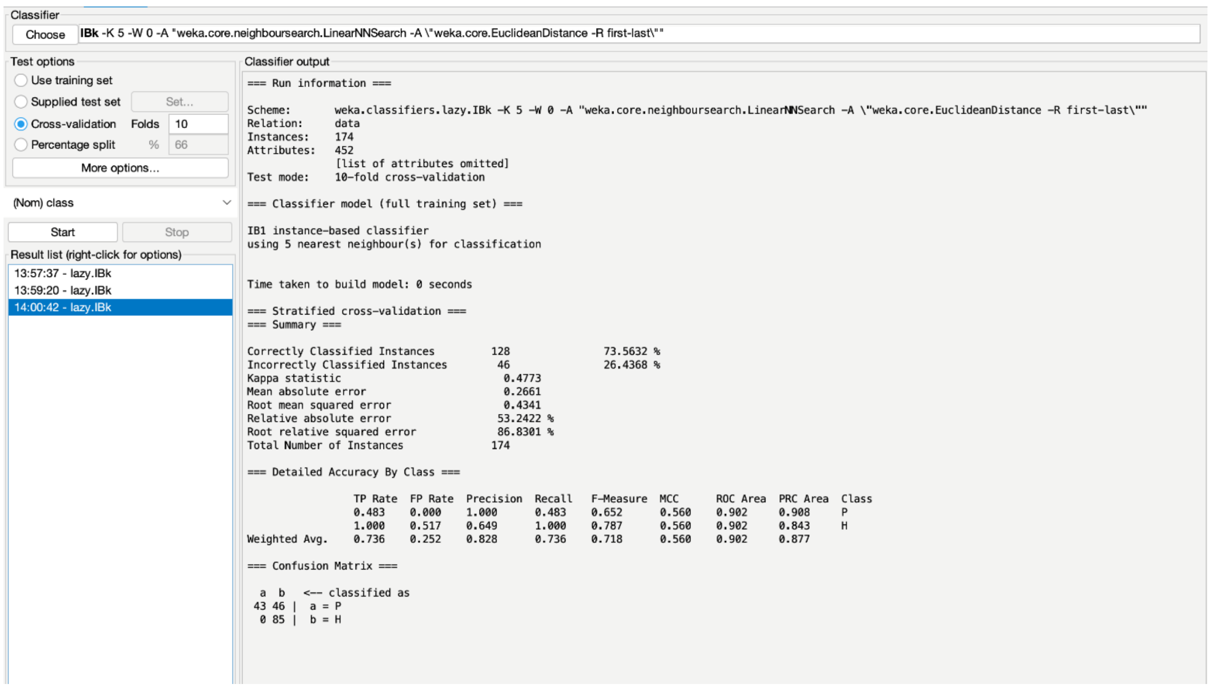
Database used : Darwin



```
Filter: None

Current relation
  Relation: data                          Attributes: 452
  Instances: 174                      Sum of weights: 174

Attributes
 No.   Name
 429 [ ] num_of_pendown24
 430 [ ] paper_time24
 431 [ ] pressure_mean24
 432 [ ] pressure_var24
 433 [ ] total_time24
 434 [ ] air_time25
 435 [ ] disp_index25
 436 [ ] gmrt_in_air25
 437 [ ] gmrt_on_paper25
 438 [ ] max_x_extension25
 439 [ ] max_y_extension25
 440 [ ] mean_acc_in_air25
 441 [ ] mean_acc_on_paper25
 442 [ ] mean_gmrt25
 443 [ ] mean_jerk_in_air25
 444 [ ] mean_jerk_on_paper25
 445 [ ] mean_speed_in_air25
 446 [ ] mean_speed_on_paper25
 447 [ ] num_of_pendown25
 448 [ ] paper_time25
 449 [ ] pressure_mean25
 450 [ ] pressure_var25
 451 [ ] total_time25
 452 [✓] class

Selected attribute
  Name: class                                   Type: Nominal
  Missing: 0 (0%)          Distinct: 2          Unique: 0 (0%)

 No.   Label      Count      Weight
   1   P          89         89
   2   H          85         85

Class: class (Nom)
```

Fig 7.1 uploading labelled data

Linear NN Search



IBk -K 5 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options: Use training set / Supplied test set / [✓ Cross-validation] Folds 10 / Percentage split % 66

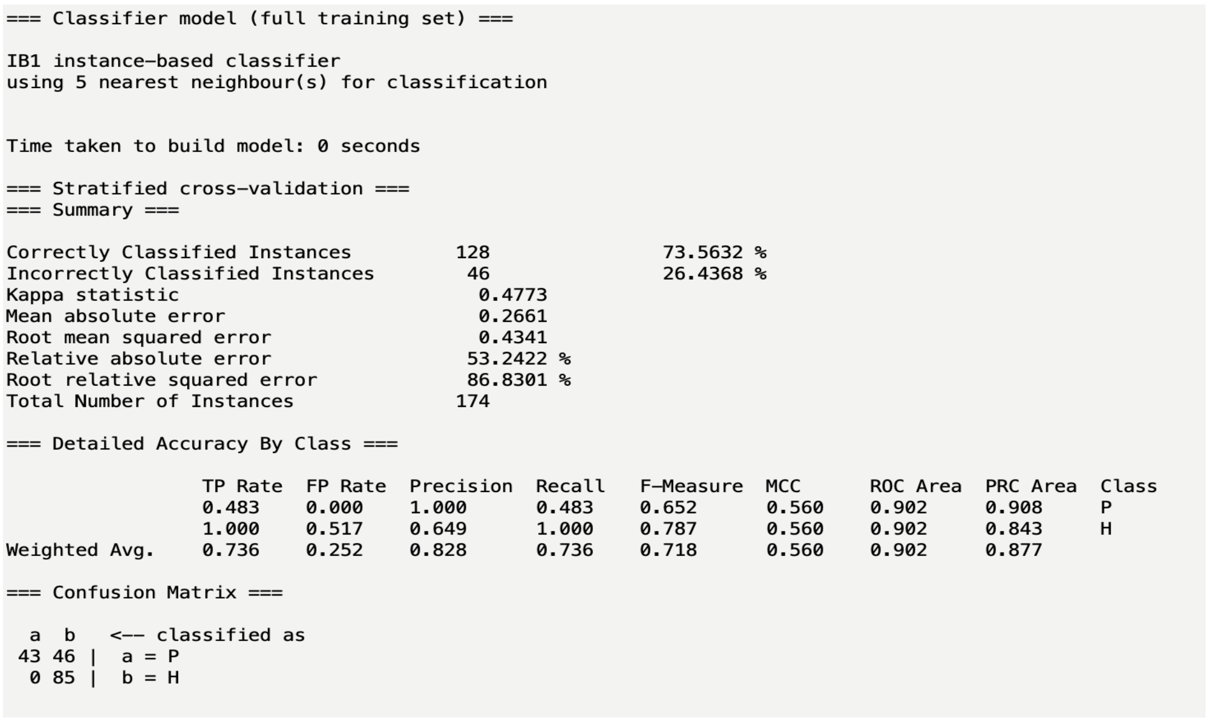
(Nom) class

Result list (right-click for options)
13:57:37 - lazy.IBk
13:59:20 - lazy.IBk
14:00:42 - lazy.IBk

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 5 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     data
Instances:    174
Attributes:   452
              [list of attributes omitted]
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         128               73.5632 %
Incorrectly Classified Instances        46               26.4368 %
Kappa statistic                          0.4773
Mean absolute error                      0.2661
Root mean squared error                  0.4341
Relative absolute error                 53.2422 %
Root relative squared error             86.8301 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.483    0.000    1.000      0.483    0.652      0.560    0.902     0.908     P
                 1.000    0.517    0.649      1.000    0.787      0.560    0.902     0.843     H
Weighted Avg.    0.736    0.252    0.828      0.736    0.718      0.560    0.902     0.877

=== Confusion Matrix ===

  a  b   <-- classified as
 43 46 |  a = P
  0 85 |  b = H
```

Fig 7.2 KNN using linear NN search

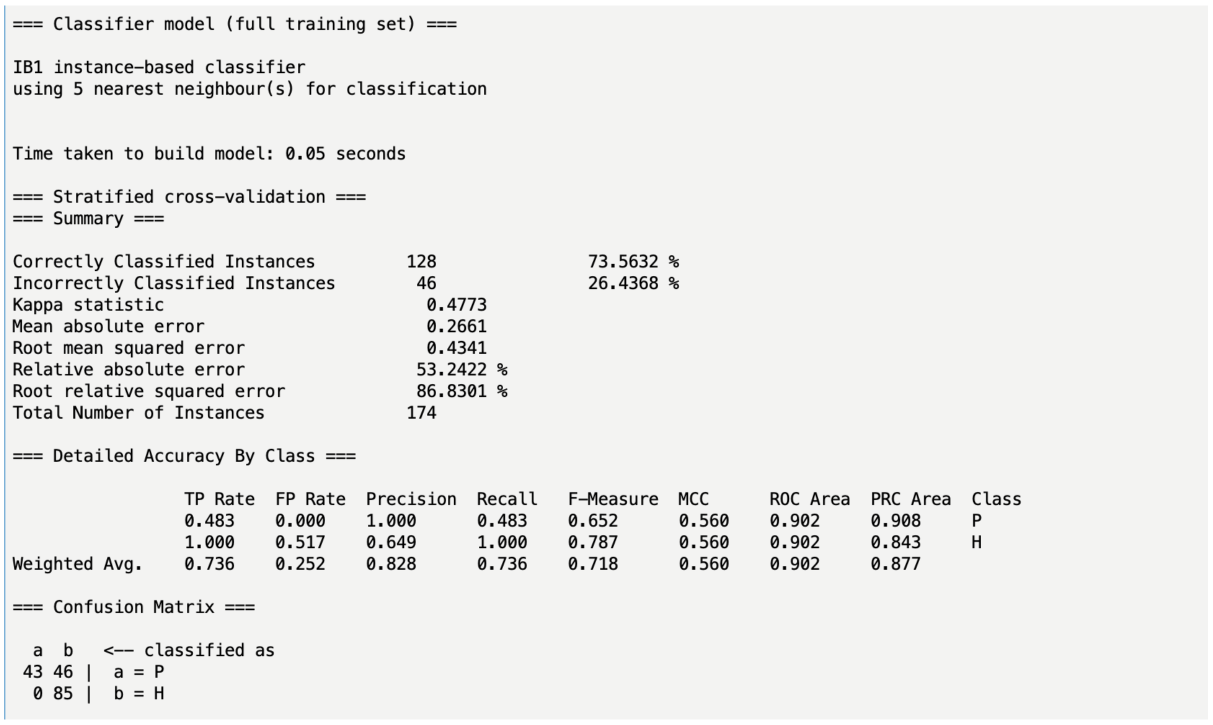
Cover Tree



```
=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         128               73.5632 %
Incorrectly Classified Instances        46               26.4368 %
Kappa statistic                          0.4773
Mean absolute error                      0.2661
Root mean squared error                  0.4341
Relative absolute error                 53.2422 %
Root relative squared error             86.8301 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.483    0.000    1.000      0.483    0.652      0.560    0.902     0.908     P
                 1.000    0.517    0.649      1.000    0.787      0.560    0.902     0.843     H
Weighted Avg.    0.736    0.252    0.828      0.736    0.718      0.560    0.902     0.877

=== Confusion Matrix ===

  a  b   <-- classified as
 43 46 |  a = P
  0 85 |  b = H
```

Fig 7.3 KNN using Cover tree

Ball Tree



```
=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 nearest neighbour(s) for classification


Time taken to build model: 0.05 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         128               73.5632 %
Incorrectly Classified Instances        46               26.4368 %
Kappa statistic                          0.4773
Mean absolute error                      0.2661
Root mean squared error                  0.4341
Relative absolute error                 53.2422 %
Root relative squared error             86.8301 %
Total Number of Instances              174

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.483    0.000    1.000      0.483    0.652      0.560    0.902     0.908     P
                 1.000    0.517    0.649      1.000    0.787      0.560    0.902     0.843     H
Weighted Avg.    0.736    0.252    0.828      0.736    0.718      0.560    0.902     0.877

=== Confusion Matrix ===

  a  b   <-- classified as
 43 46 |  a = P
  0 85 |  b = H
```

Fig 7.4 KNN using Ball tree

Using : Orange tool

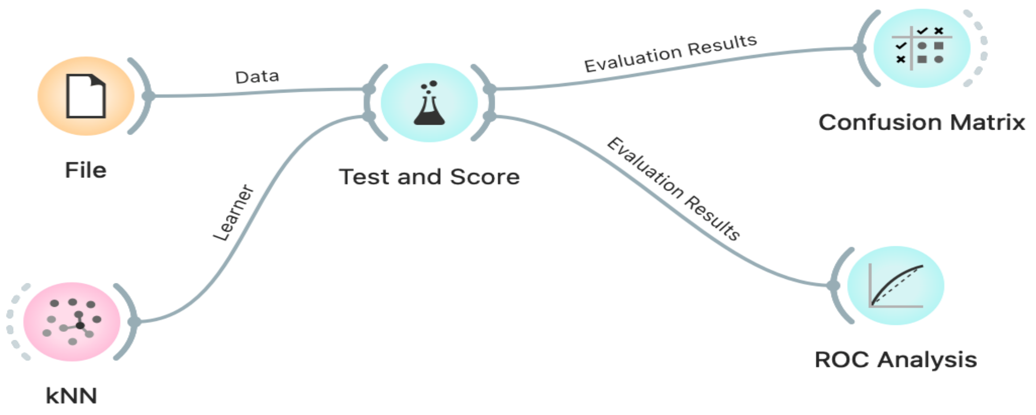
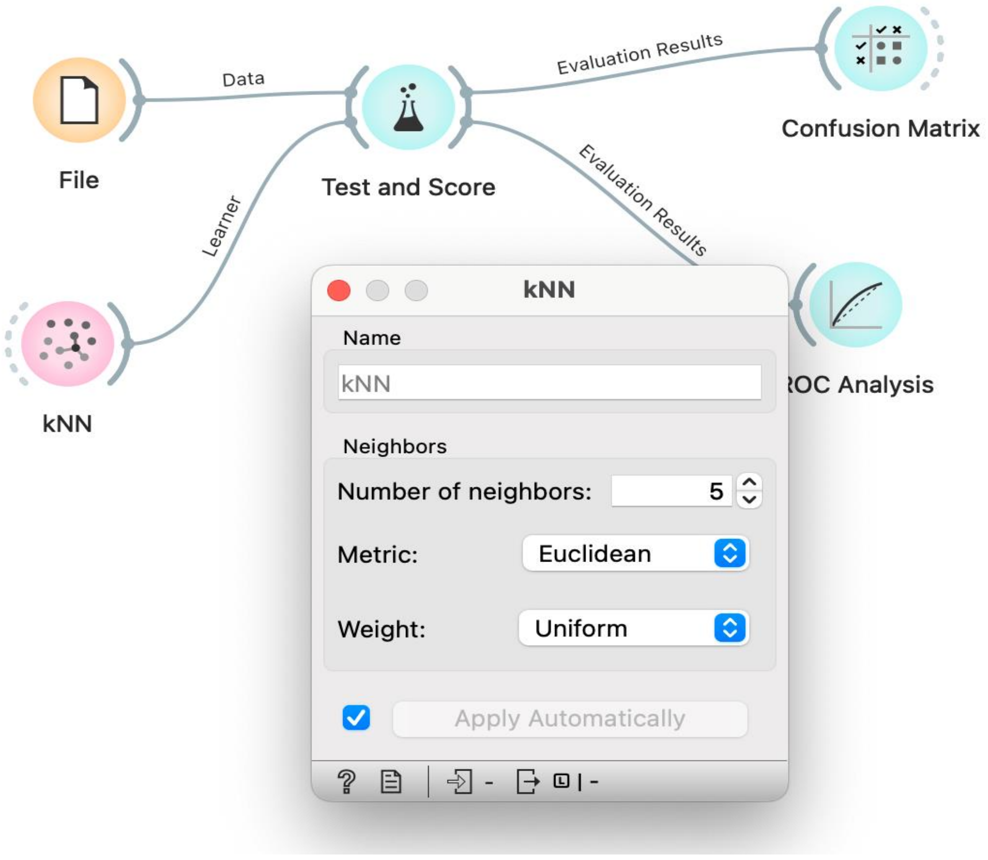


Fig 7.5 KNN using Orange tool

Matric : Euclidean



kNN

Name: kNN

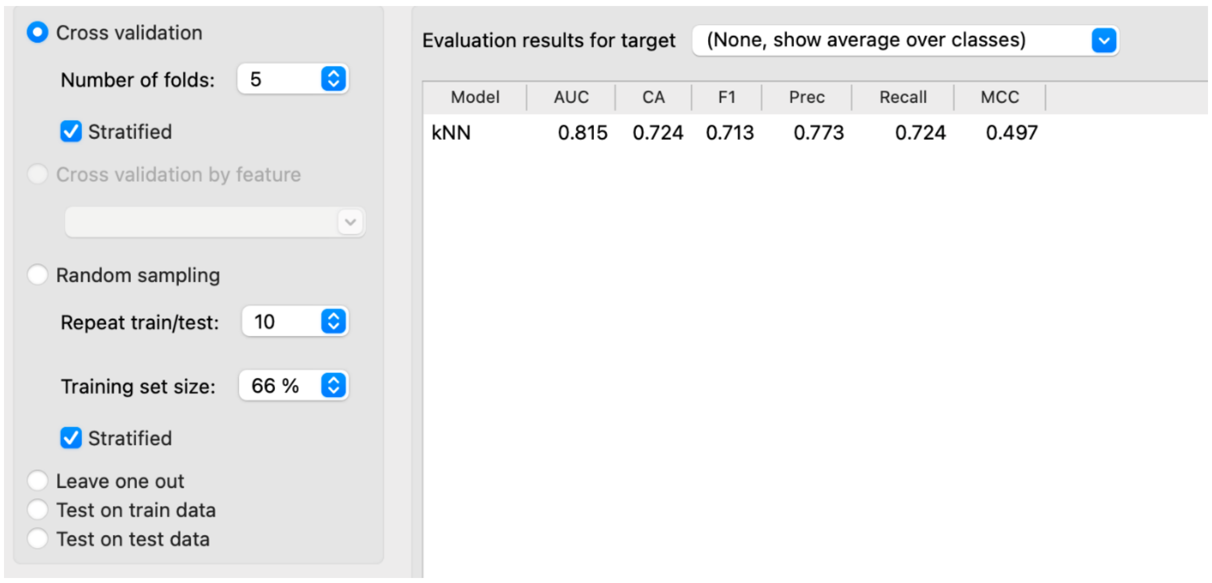
Neighbors

Number of neighbors: 5

Metric: Euclidean

Weight: Uniform

Fig 7.6 using Euclidean matric



Cross validation [✓]

Number of folds: 5

Stratified [✓]

Cross validation by feature

Random sampling

Repeat train/test: 10

Training set size: 66 %

Stratified [✓]

Leave one out

Test on train data

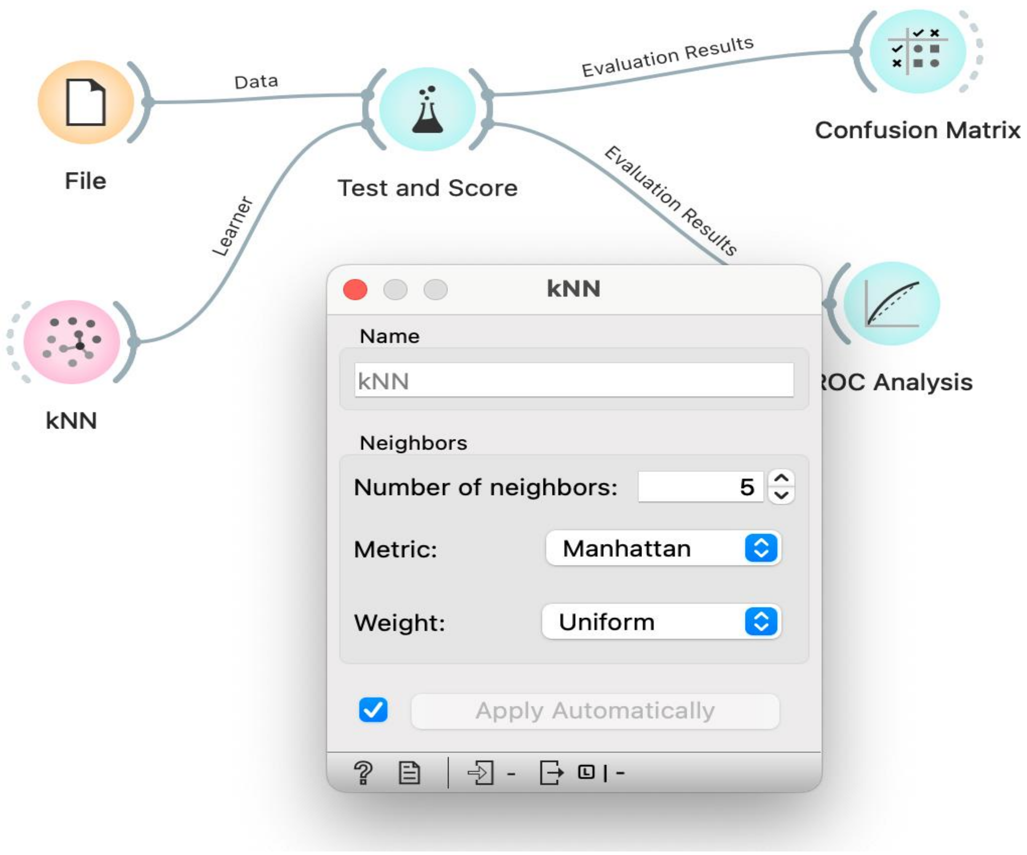
Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.815	0.724	0.713	0.773	0.724	0.497

Fig 7.7 Evaluation result

Matric : Manhattan



kNN

Name: kNN

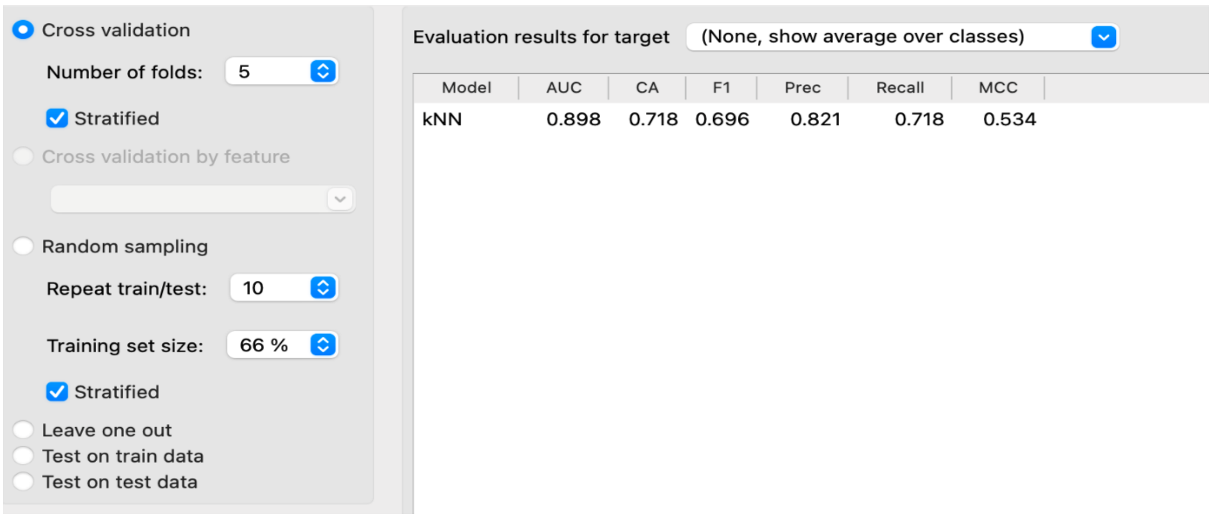
Neighbors

Number of neighbors: 5

Metric: Manhattan

Weight: Uniform

Fig 7.8 KNN using Manhattan matric



[✓ Cross validation]

Number of folds: 5

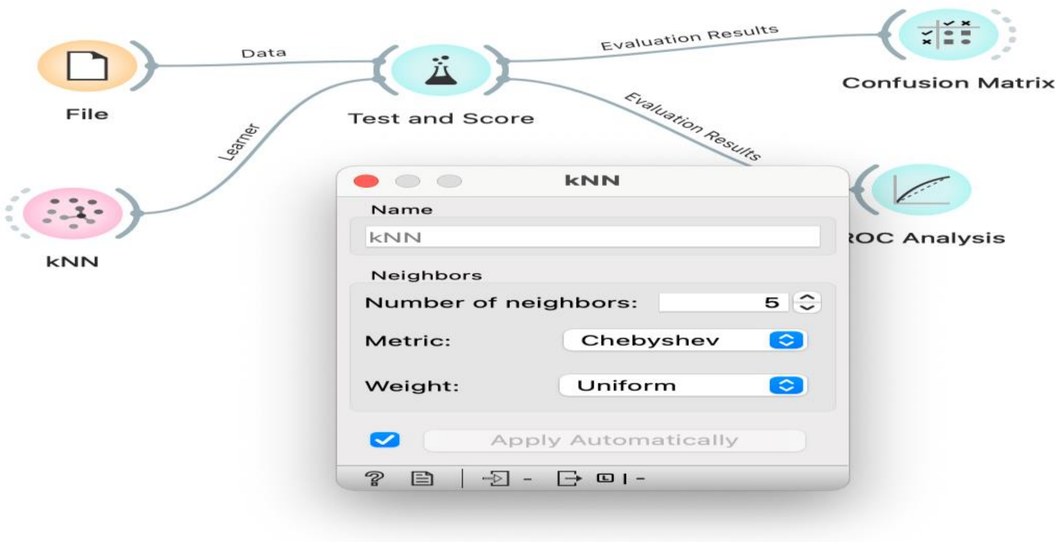
Stratified ✓

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.898	0.718	0.696	0.821	0.718	0.534

Fig 7.9 Evaluation result

Matric: Chebyshev



kNN

Name: kNN

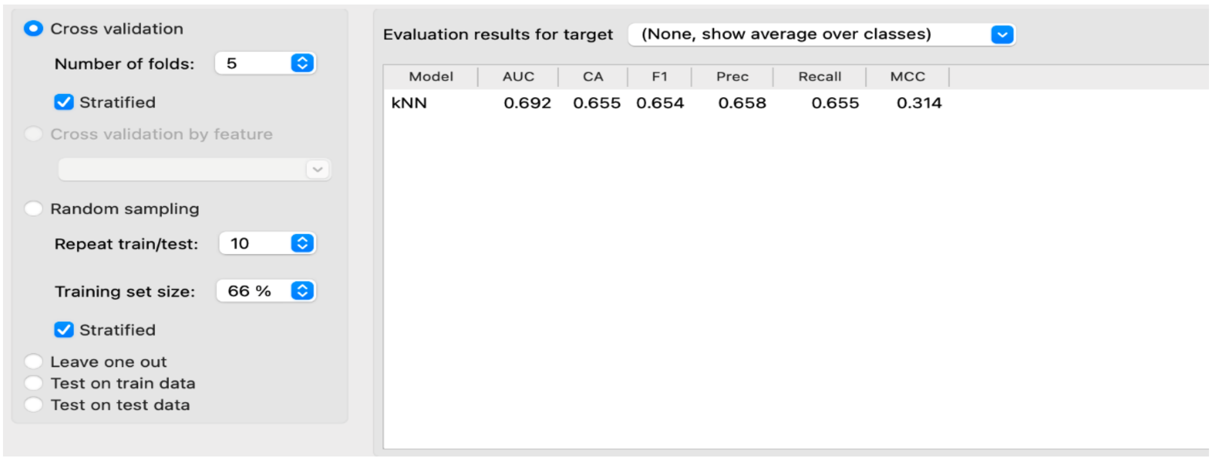
Neighbors

Number of neighbors: 5

Metric: Chebyshev

Weight: Uniform

☑ Apply Automatically



(● Cross validation)

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

○ Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Prec	Recall	MCC
kNN	0.692	0.655	0.654	0.658	0.655	0.314

Fig 7.10 using Chebyshev matric

Practical#8

Aim :- Perform DBSCAN Clustering in Weka and Orange

Theory : DBSCAN, which stands for Density-Based Spatial Clustering of Applications with Noise, is a popular clustering algorithm in data mining and machine learning. It's used to group together points that are closely packed together, marking as outliers those that lie alone in low-density regions.

Here's a brief overview of how DBSCAN works:

1. Density-Based: DBSCAN groups together points based on their density. It doesn't require the user to specify the number of clusters beforehand.
2. Core Points: It defines two parameters: epsilon (ε), which specifies the radius of the neighborhood around each point, and MinPts, the minimum number of points required to form a dense region (cluster).
3. Reachability: A point is considered reachable from another if it is within the ε distance of that point.
4. Core, Border, and Noise Points:
    - Core points: Points with at least MinPts within ε distance.
    - Border points: Points within ε distance of a core point but with fewer than MinPts neighbors.
    - Noise points: Points that are neither core nor border points.
5. Clustering: DBSCAN starts with an arbitrary point and expands the cluster by adding all reachable points to the cluster. It continues this process until the cluster is maximally expanded, and then it starts a new cluster with a new unvisited point.

In Weka :-

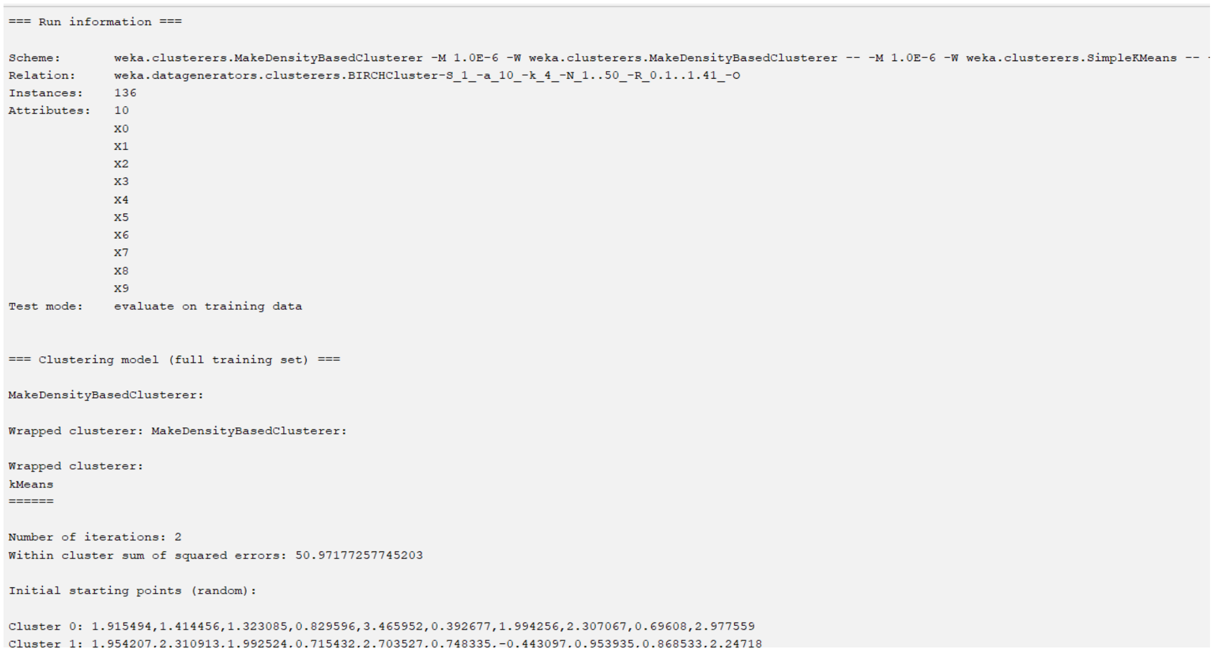
In Weka, we can perform clustering using various algorithms such as k-means, hierarchical clustering, and EM (Expectation-Maximization) clustering. Although DBSCAN is not available out-of-the-box, we can implement a similar approach using the "DBSCAN" package in Weka. This package provides a DBSCAN implementation that we can use within Weka.

We can install the DBSCAN package in Weka by following these steps:

1. Download the DBSCAN package (JAR file) from the Weka package repository or other sources.
2. Place the JAR file in the weka/packages directory.
3. Restart Weka.
4. We should now see DBSCAN listed among the available algorithms.

Dataset chosen is generated by Weka.

Dataset is Breast Cancer Data.



```
=== Run information ===

Scheme:       weka.clusterers.MakeDensityBasedClusterer -M 1.0E-6 -W weka.clusterers.MakeDensityBasedClusterer -- -M 1.0E-6 -W weka.clusterers.SimpleKMeans -- 
Relation:     weka.datagenerators.clusterers.BIRCHCluster-S_1_-a_10_-k_4_-N_1..50_-R_0.1..1.41_-O
Instances:    136
Attributes:   10
              X0
              X1
              X2
              X3
              X4
              X5
              X6
              X7
              X8
              X9
Test mode:    evaluate on training data


=== Clustering model (full training set) ===

MakeDensityBasedClusterer:

Wrapped clusterer: MakeDensityBasedClusterer:

Wrapped clusterer:
kMeans
======

Number of iterations: 2
Within cluster sum of squared errors: 50.97177257745203

Initial starting points (random):

Cluster 0: 1.915494,1.414456,1.323085,0.829596,3.465952,0.392677,1.994256,2.307067,0.69608,2.977559
Cluster 1: 1.954207,2.310913,1.992524,0.715432,2.703527,0.748335,-0.443097,0.953935,0.868533,2.24718
```

Fig 8.1 shows the application of DBSCAN on dataset.

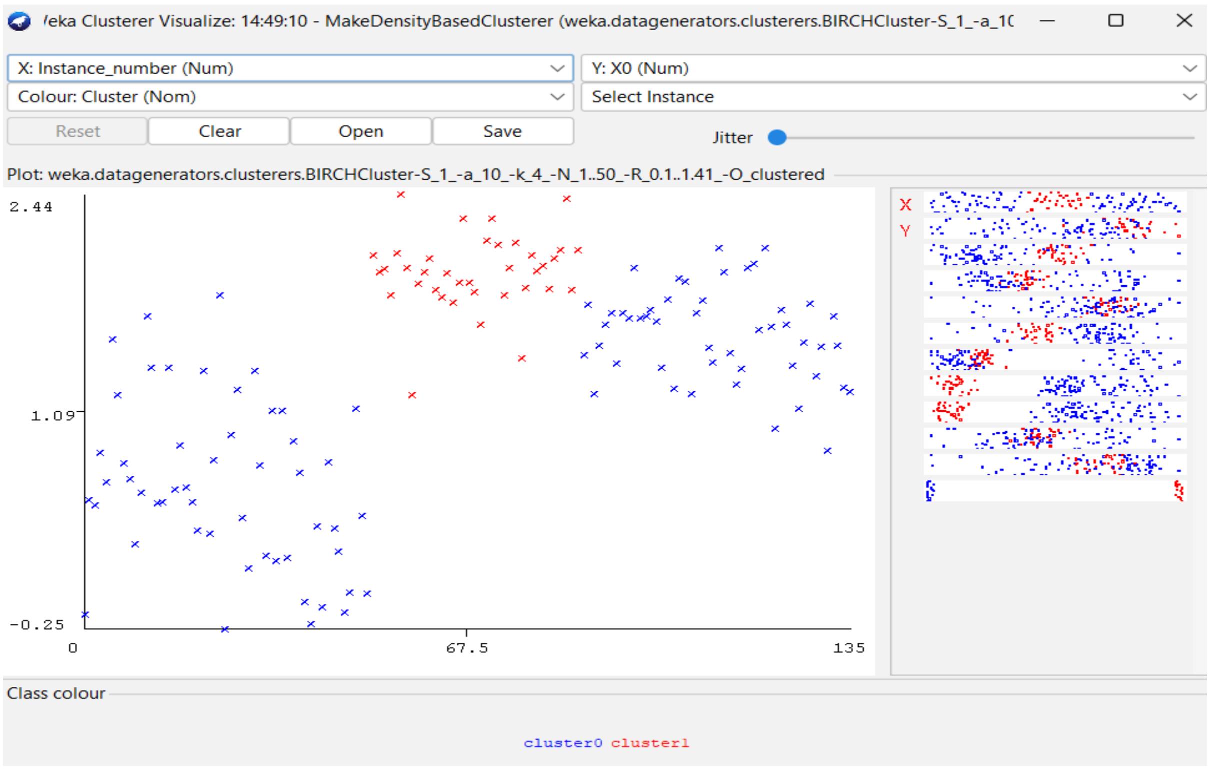
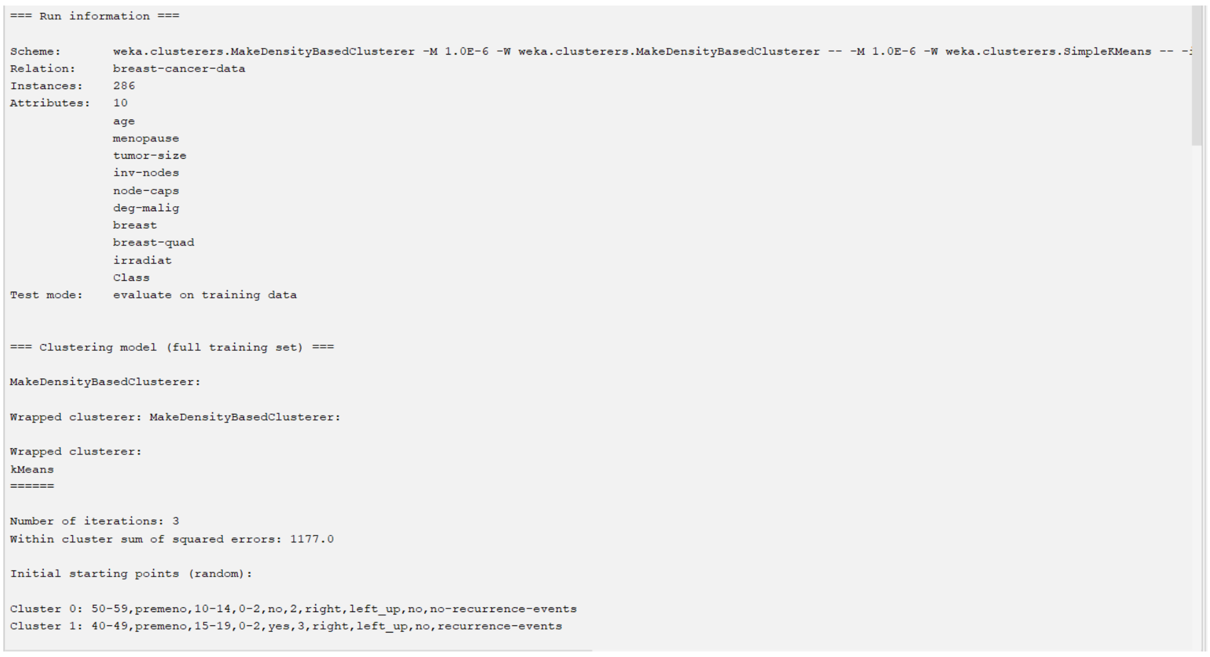


Fig 8.2 shows the visualization of clusters after applying DBSCAN algorithm.

Dataset is Breast Cancer Data.



```
=== Run information ===

Scheme:       weka.clusterers.MakeDensityBasedClusterer -M 1.0E-6 -W weka.clusterers.MakeDensityBasedClusterer -- -M 1.0E-6 -W weka.clusterers.SimpleKMeans -- -
Relation:     breast-cancer-data
Instances:    286
Attributes:   10
              age
              menopause
              tumor-size
              inv-nodes
              node-caps
              deg-malig
              breast
              breast-quad
              irradiat
              Class
Test mode:    evaluate on training data


=== Clustering model (full training set) ===

MakeDensityBasedClusterer:

Wrapped clusterer: MakeDensityBasedClusterer:

Wrapped clusterer:
kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 1177.0

Initial starting points (random):

Cluster 0: 50-59,premeno,10-14,0-2,no,2,right,left_up,no,no-recurrence-events
Cluster 1: 40-49,premeno,15-19,0-2,yes,3,right,left_up,no,recurrence-events
```

Fig 8.3 shows the application of DBSCAN algorithm on the dataset.

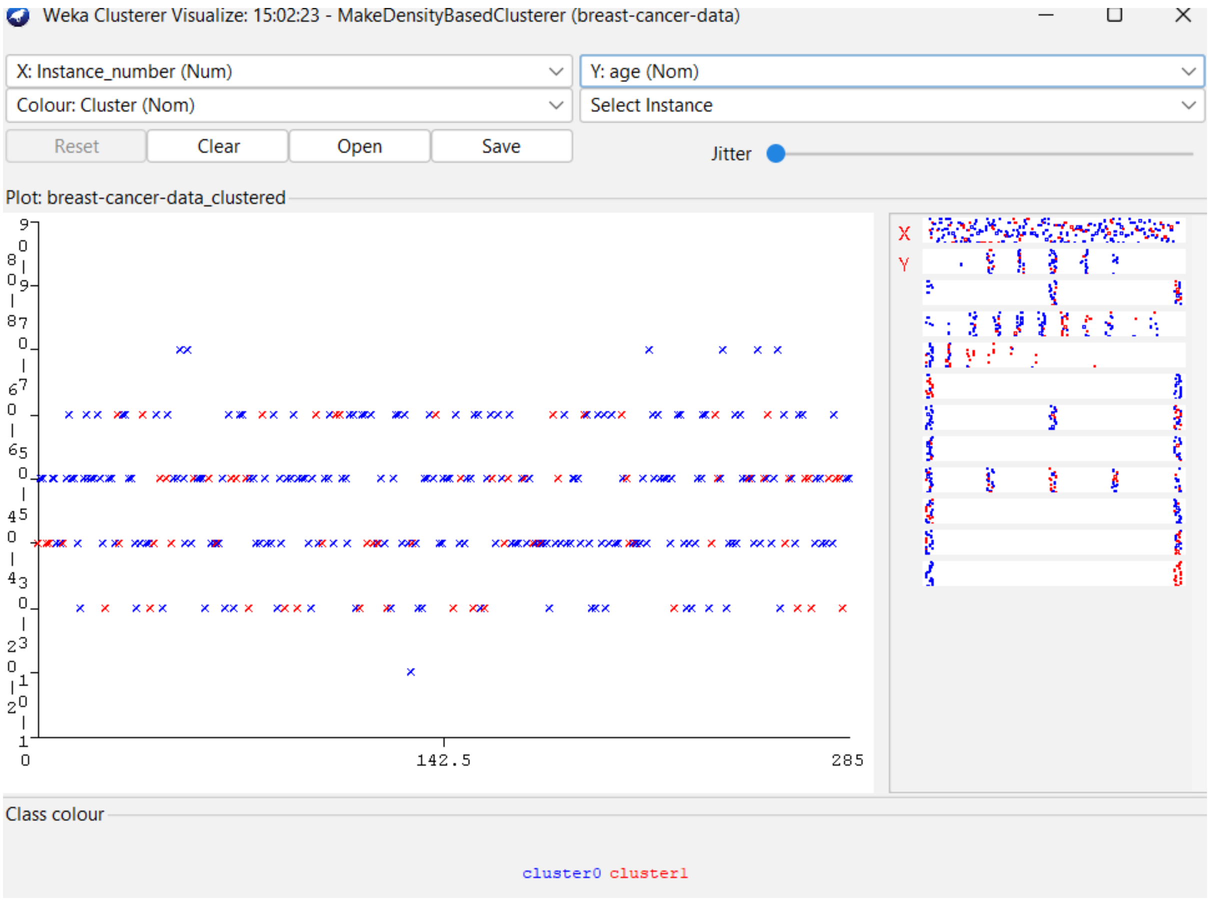
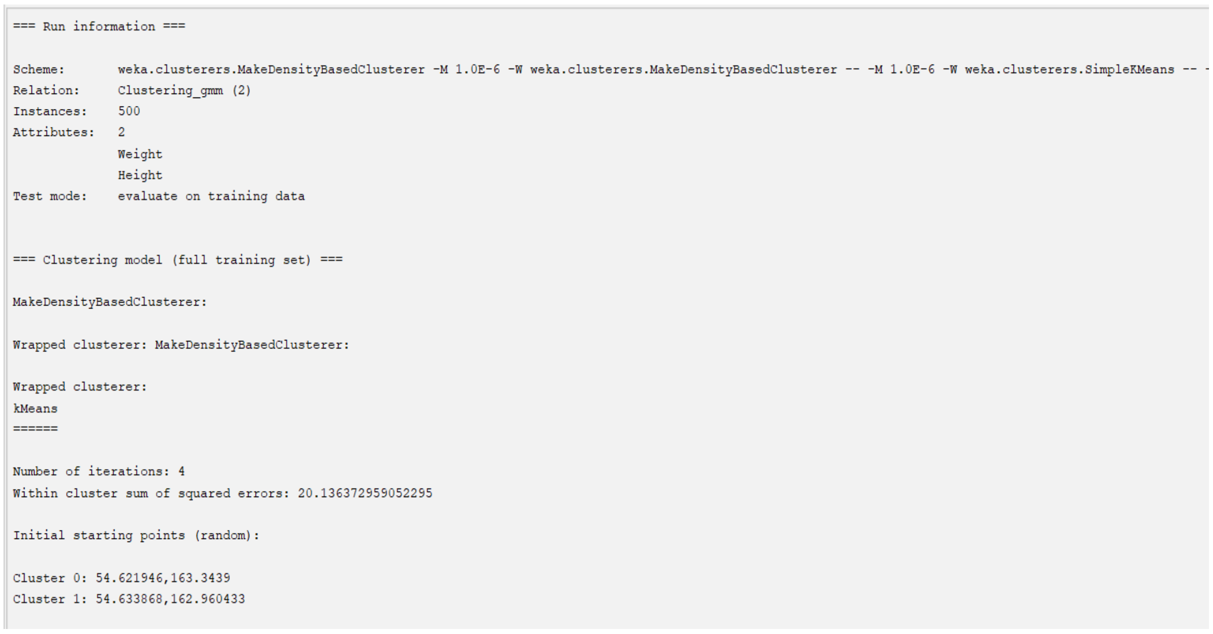


Fig 8.4 shows the visualization of clusters after applying DBSCAN Algorithm.

Dataset is archive.



```
=== Run information ===

Scheme:       weka.clusterers.MakeDensityBasedClusterer -M 1.0E-6 -W weka.clusterers.MakeDensityBasedClusterer -- -M 1.0E-6 -W weka.clusterers.SimpleKMeans -- -
Relation:     Clustering_gmm (2)
Instances:    500
Attributes:   2
              Weight
              Height
Test mode:    evaluate on training data


=== Clustering model (full training set) ===

MakeDensityBasedClusterer:

Wrapped clusterer: MakeDensityBasedClusterer:

Wrapped clusterer:
kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 20.136372959052295

Initial starting points (random):

Cluster 0: 54.621946,163.3439
Cluster 1: 54.633868,162.960433
```

Fig 8.5 shows the applying of DBSCAN on the dataset.

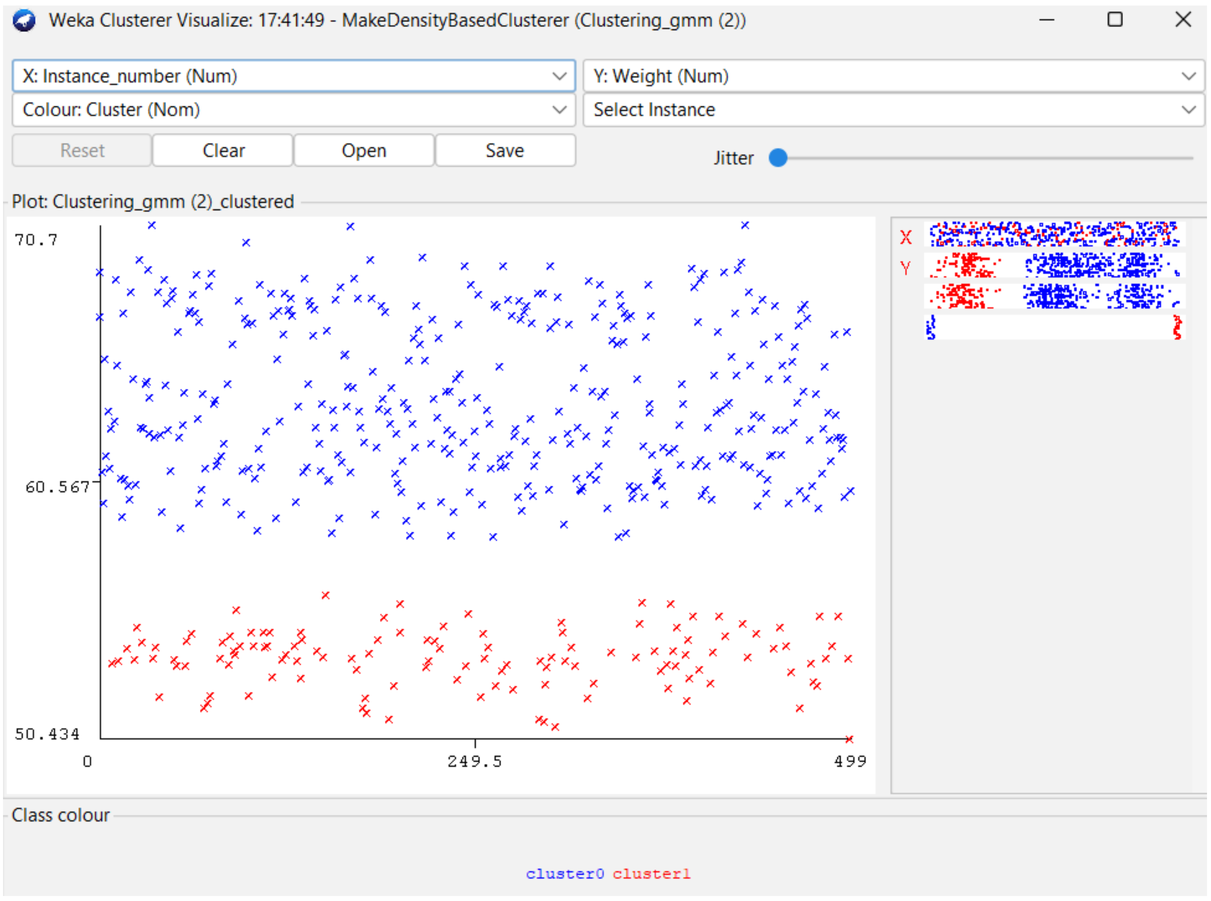


Fig 8.6 shows the visualization of cluster after applying DBSCAN Algorithm.

In Orange :- In Orange, we can perform clustering using algorithms such as k-means, hierarchical clustering, and DBSCAN-like clustering through the DBSCAN and OPTICS algorithms, available in the "orange3-associate" add-on. Here's how we can use DBSCAN-like clustering in Orange:

1. Install the "orange3-associate" add-on if we haven't already. We can do this through the Orange GUI or by using pip:

   ```
   pip install orange3-associate
   ```

2. Launch Orange and load wer dataset.
3. Drag the "DBSCAN" widget from the "Unsupervised" category into the workflow canvas.
4. Connect the dataset to the DBSCAN widget.
5. Configure the parameters of the DBSCAN algorithm, such as epsilon (ε) and MinPts.
6. Run the workflow to perform DBSCAN-like clustering on wer dataset.

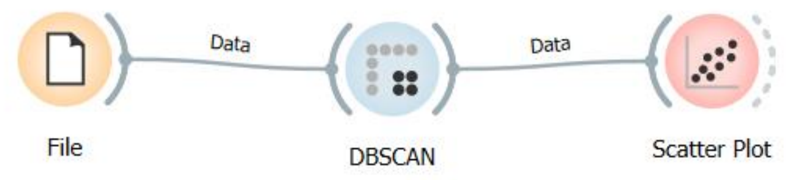


Fig 8.7 shows the building of model on dataset.

Dataset is IRIS.

DBSCAN

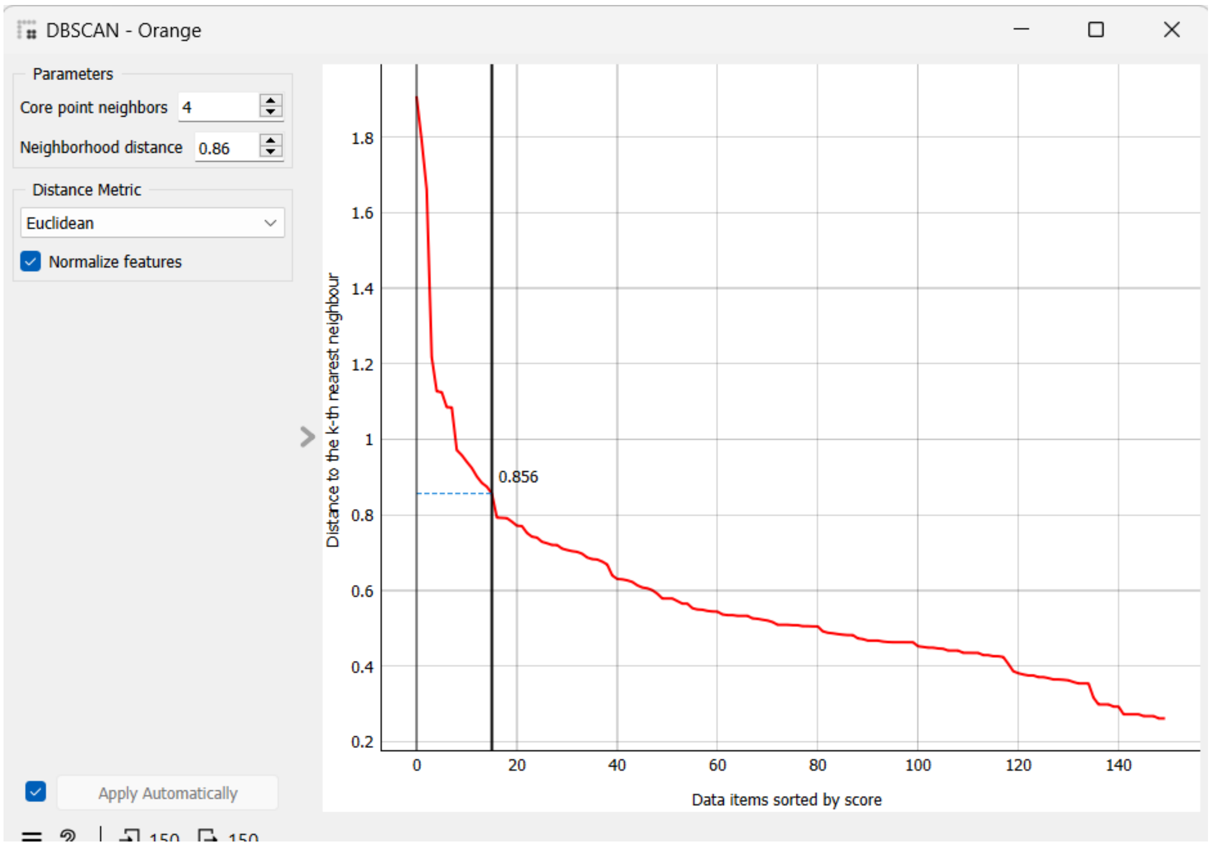


Fig 8.8 shows the DBSCAN applying on dataset.

Scatter Plot

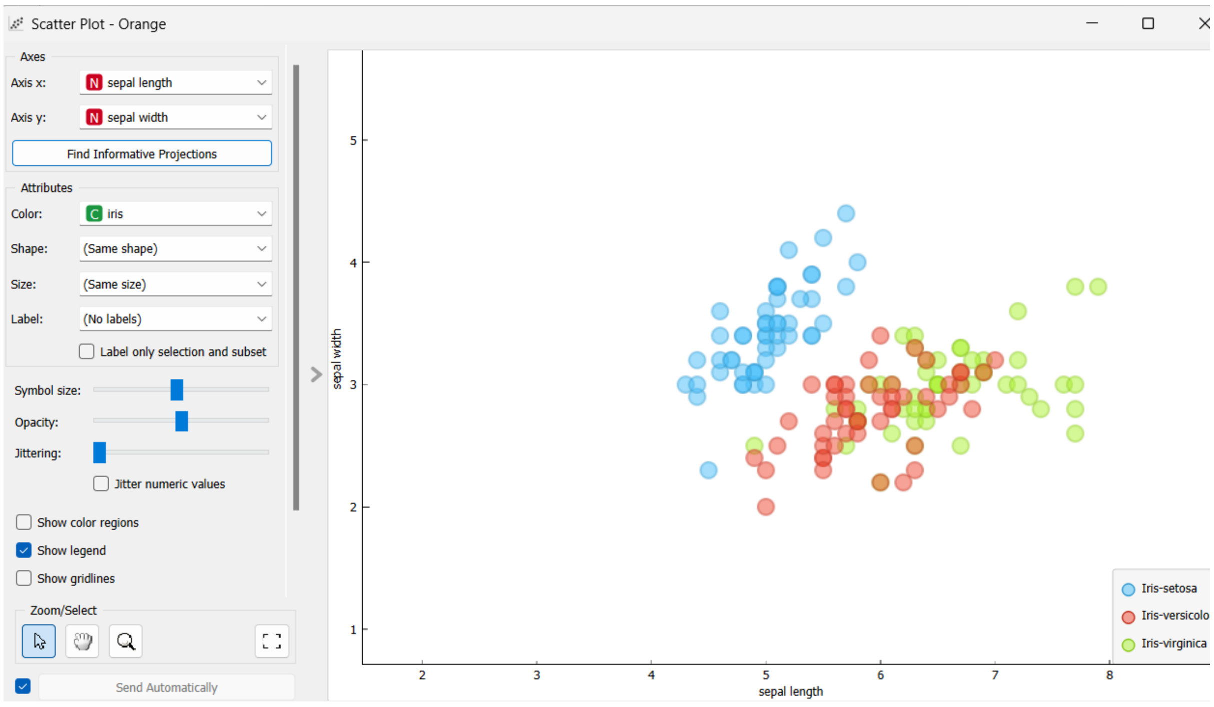
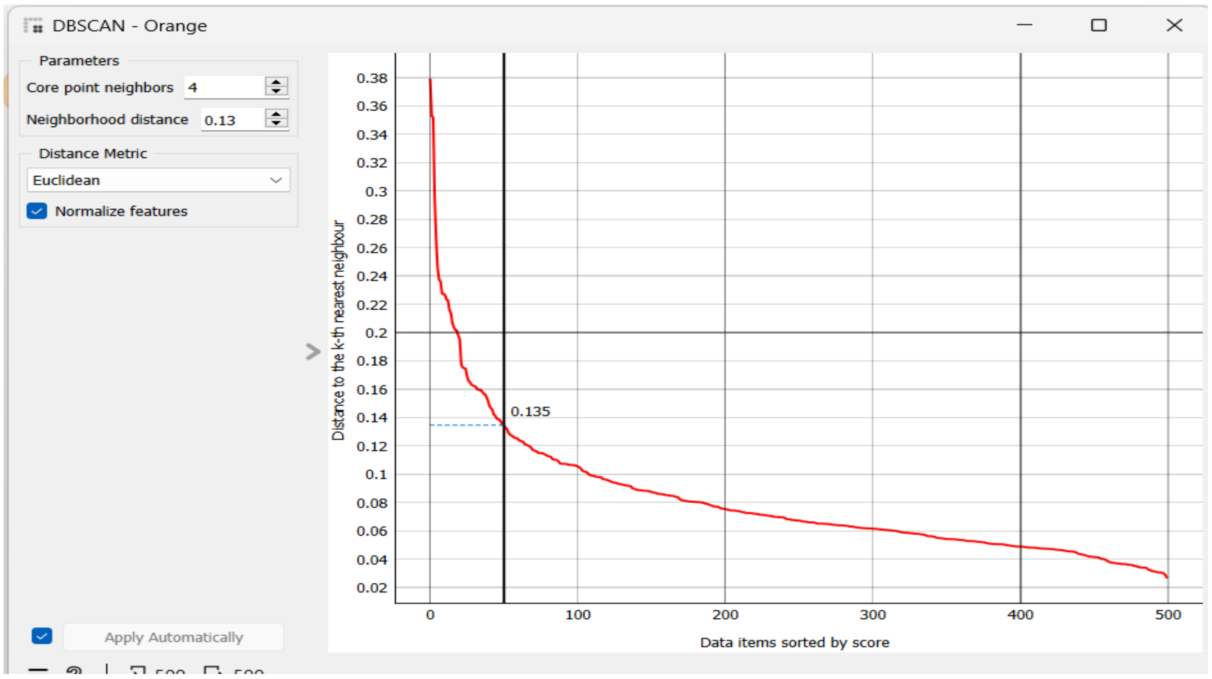


Fig 8.9 shows the scatterplot formed after applying DBSCAN on dataset.

Dataset is archive.

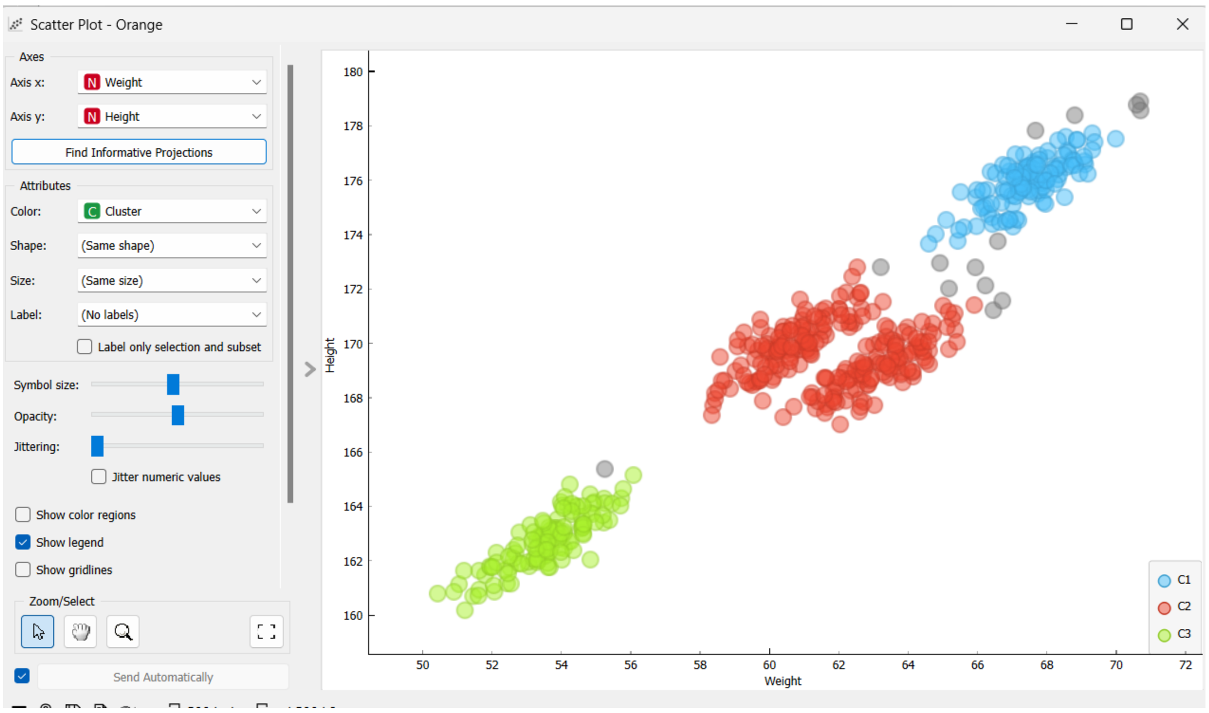
DBSCAN


DBSCAN - Orange

Parameters
Core point neighbors 4
Neighborhood distance 0.13
Distance Metric
Euclidean
✓ Normalize features
✓ Apply Automatically

Fig 8.10 shows the DBSCAN applying on dataset.

Scatter Plot


Scatter Plot - Orange

Axes
Axis x: Weight
Axis y: Height
Find Informative Projections
Attributes
Color: Cluster
Shape: (Same shape)
Size: (Same size)
Label: (No labels)
Label only selection and subset
Symbol size:
Opacity:
Jittering:
Jitter numeric values
Show color regions
✓ Show legend
Show gridlines
Zoom/Select
✓ Send Automatically

Fig 8.11 shows the scatterplot formed after applying DBSCAN on dataset

Practical#9

Objective: Perform Heirarchial Clusrtering

Theory:

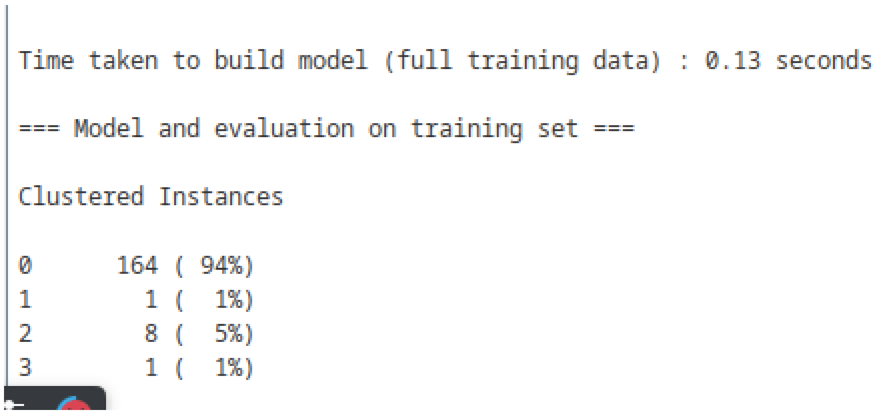
Hierarchical clustering is a popular method used to group similar items into clusters. It builds a hierarchy of clusters either top-down (divisive) or bottom-up (agglomerative). Agglomerative clustering is more commonly used.

PROCEDURE:

USING WEKA TOOL:

Scenario#1:

WITH NUMBER OF CLUSTERS= 4 AND DISTANCE = EUCLIDEAN

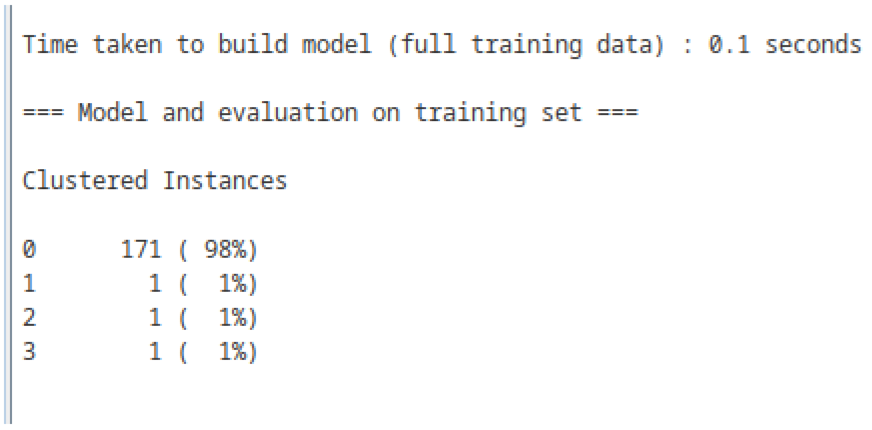


```
Time taken to build model (full training data) : 0.13 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       164 ( 94%)
1         1 (  1%)
2         8 (  5%)
3         1 (  1%)
```

7.9Information with number of cluster =4 and distance =euclidean

Scenario#2:

WITH NUMBER OF CLUSTERS= 4 AND DISTANCE = MANHATTAN



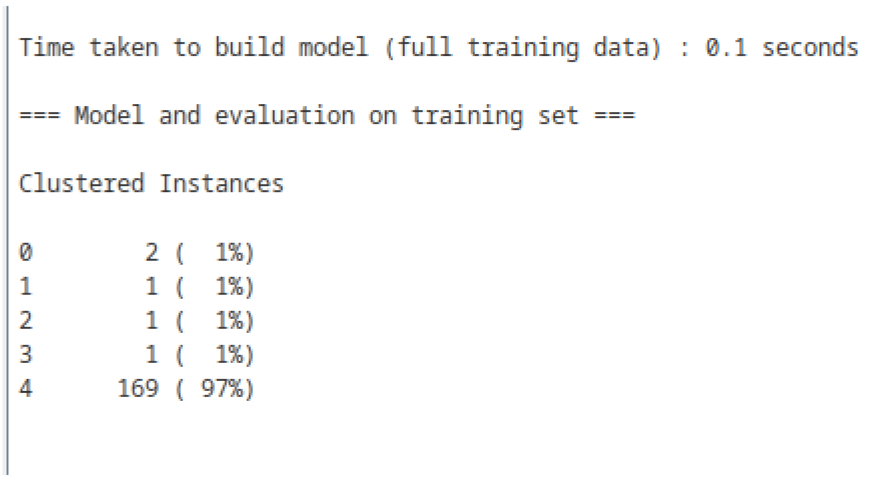
```
Time taken to build model (full training data) : 0.1 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      171 ( 98%)
1        1 (  1%)
2        1 (  1%)
3        1 (  1%)
```

Fig 7.10 Information with number of cluster =4 and distance =manhattan

Scenario#3:

WITH NUMBER OF CLUSTERS= 5 AND DISTANCE = CHEBYSHEV

Fig 7.11 Information with number of cluster =4 and distance =Chebyshev



```
Time taken to build model (full training data) : 0.1 seconds

=== Model and evaluation on training set ===

Clustered Instances

0        2 (  1%)
1        1 (  1%)
2        1 (  1%)
3        1 (  1%)
4      169 ( 97%)
```

USING ORANGE

Import your dataset into Orange using the graphical interface or Python scripting.

Drag and drop the "Hierarchical Clustering" widget onto the canvas.

Set the parameters for hierarchical clustering, such as the distance metric (e.g., Euclidean, Manhattan) and the linkage method

Connect the output of your data source to the input of the Hierarchical Clustering widget.

Execute the workflow to perform hierarchical clustering and visualize the dendrogram, which illustrates the clustering hierarchy and how clusters merge.

Analyze the dendrogram to determine the optimal number of clusters based on the cluster fusion levels.

Optionally, use clustering widgets or Python scripting to extract clusters based on the dendrogram's structure.

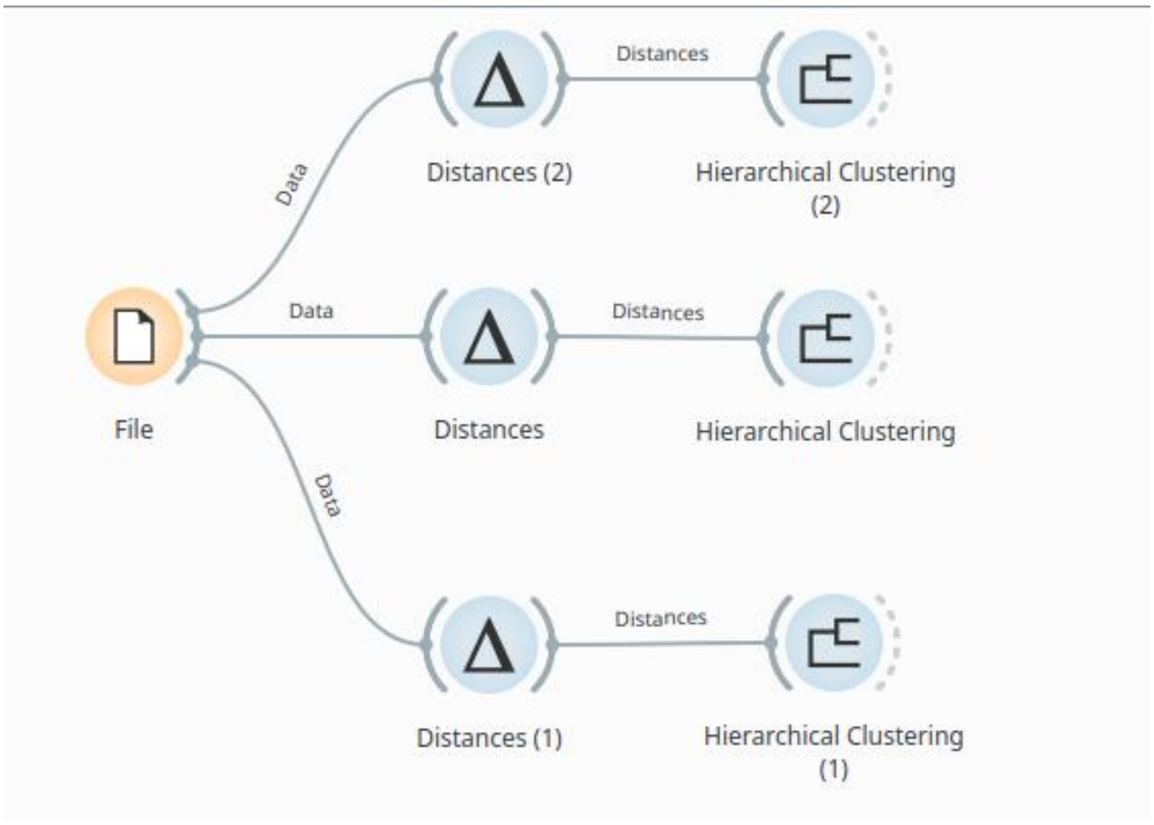


Fig 7.12 Heirarchial clustering using orange

A **dendrogram** is a tree-like diagram that illustrates the arrangement of the clusters produced by hierarchical clusteringIt illustrates the order and distances at which clusters are merged. The y-axis of the dendrogram represents the distance or dissimilarity between clusters, while the x-axis represents individual data points or clusters

Scenario#1: Using Pearson Distance

The Pearson distance, also known as the Pearson correlation coefficient, is a measure of the linear correlation between two variables. It is commonly used as a distance metric in hierarchical clustering, especially when dealing with continuous variables.

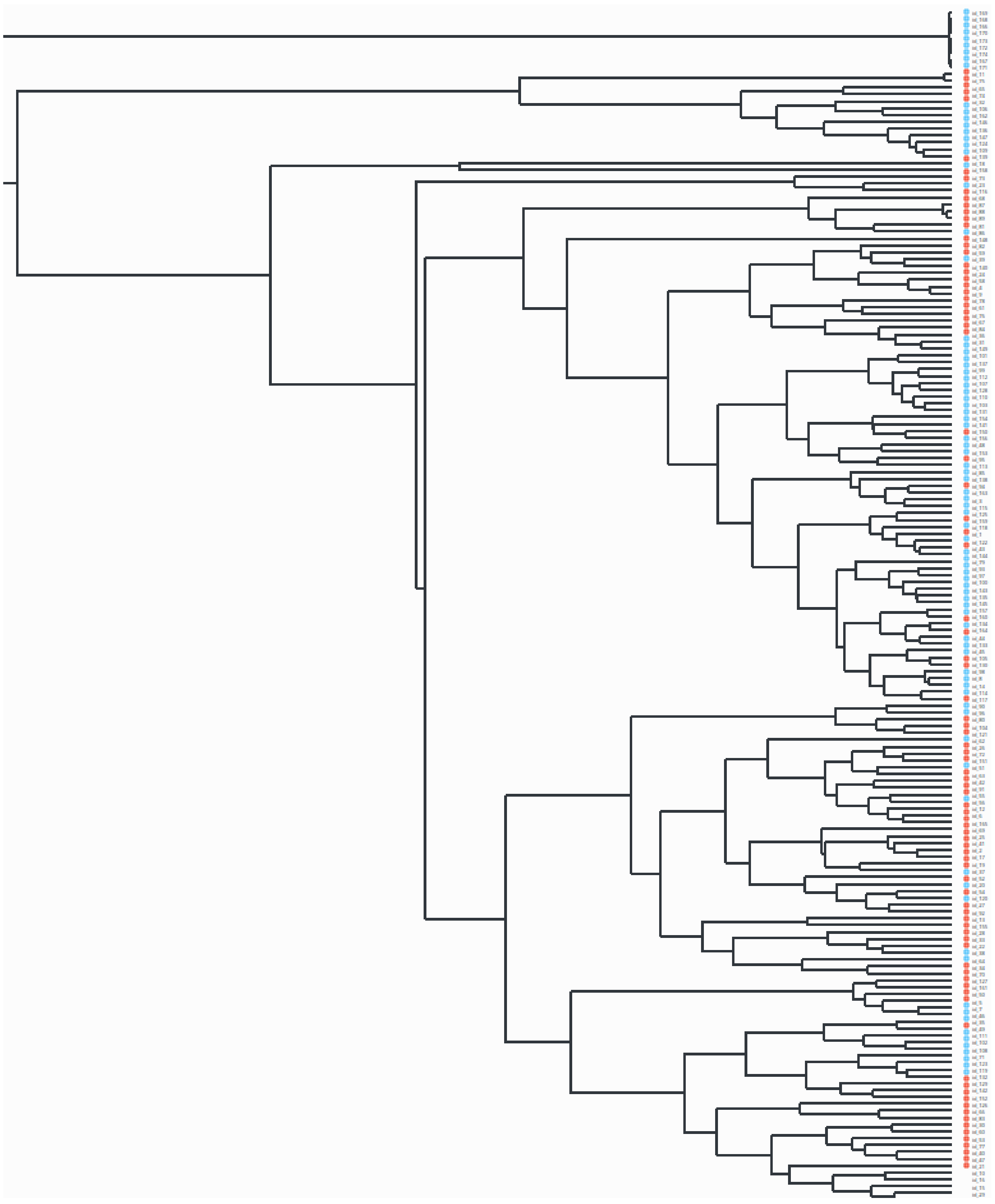


Fig 7.13 Using Pearson Distance

Scenario#2: Using Euclidean Distance

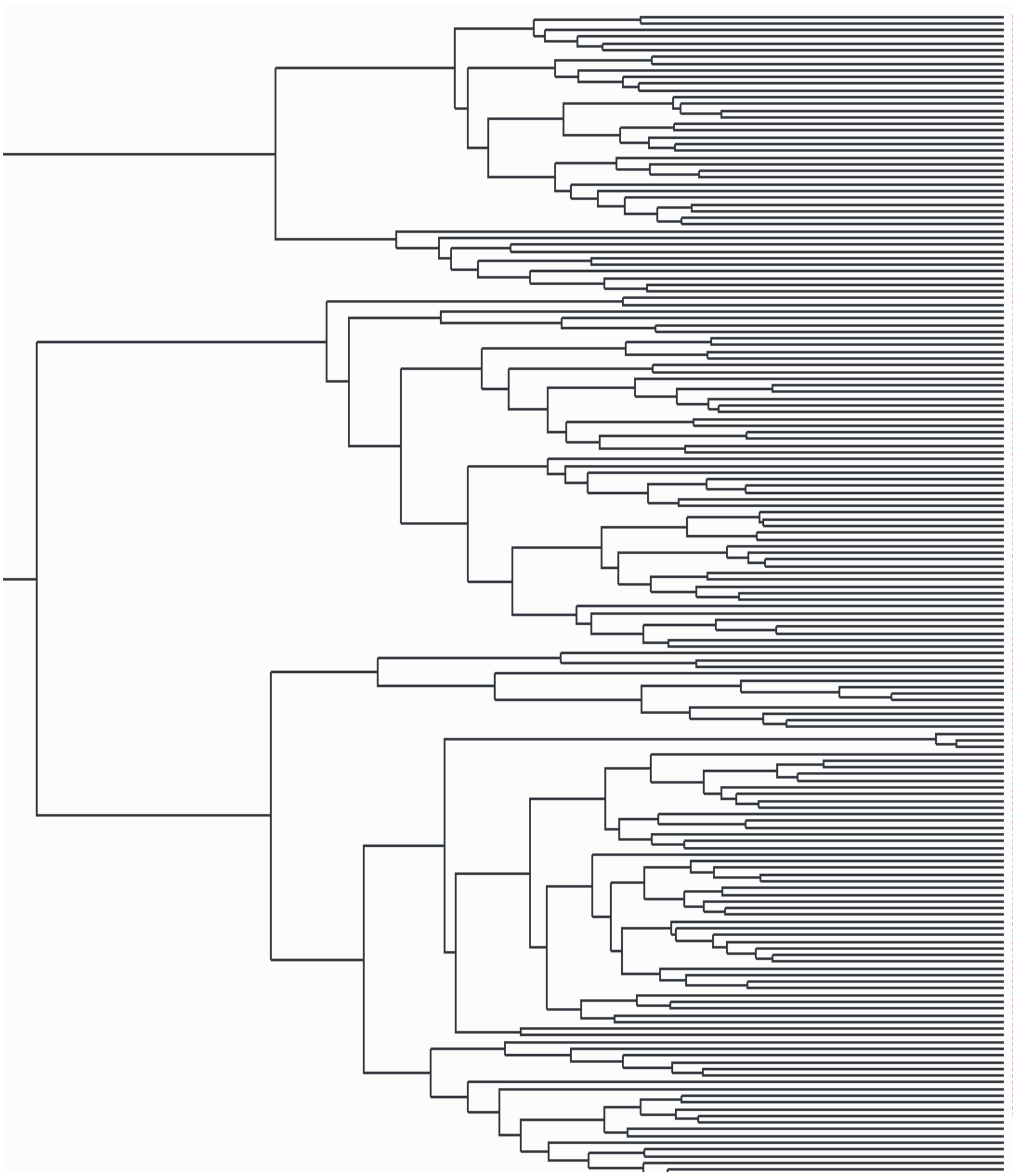


Fig 7.14 Using Euclidean Distance

Scenario#3: Using Cosine Distance

The cosine distance, also known as the cosine similarity, is a measure of similarity between two vectors in a multidimensional space. It calculates the cosine of the angle between the two vectors, indicating the degree of alignment between them.

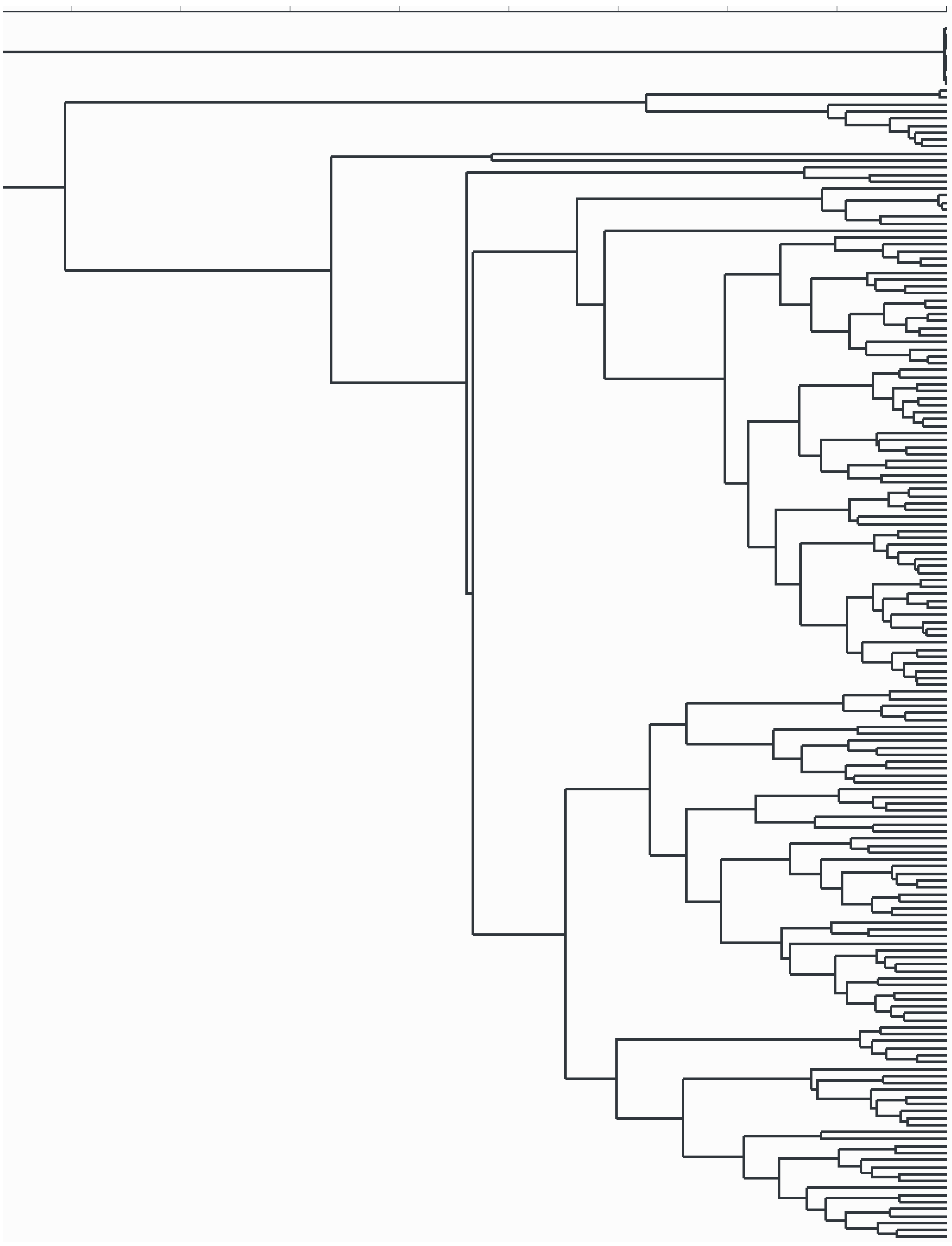


Fig 7.15 Using Cosine Distance