# Opis 1. zadatka

/\*

Importiramo podatke data.csv koje smo dobili preko get\_data.py ručno u MongoDBCompass.

1 Korak:

  - kreiranje baze (naziv baze: aids) i kolekcije (naziv kolekcije: clinical\_trials)

  - Prilikom importiranja data.csv delimiter je "Comma" isključumo Ignore empty strings

2 Korak:

Prije pokretanje ove skripte potrebno je iz terminala pokrenuti MongoDB Shellu s naredbom: "mongosh"

Naredba za učitavnje i pokretanje skripte: "load("data.js")"

\*/

/\*

Opis varijabli:

time - time to failure or censoring;

trt - treatment indicator (0 = ZDV only; 1 = ZDV + ddI, 2 = ZDV + Zal, 3 = ddI only)

age - age (yrs) at baseline

wtkg - weight (kg) at baseline

hemo - hemophilia (0=no, 1=yes)

homo - homosexual activity (0=no, 1=yes)

drugs - history of IV drug use (0=no, 1=yes)

karnof - Karnofsky score (on a scale of 0-100)

oprior - Non-ZDV antiretroviral therapy pre-175 (0=no, 1=yes)

z30 - ZDV in the 30 days prior to 175 (0=no, 1=yes)

zprior - ZDV prior to 175 (0=no, 1=yes)

preanti - # days pre-175 anti-retroviral therapy

race- race (0=White, 1=non-white)

gender - gender (0=F, 1=M)

str2 - antiretroviral history (0=naive, 1=experienced)

strat - antiretroviral history stratification (1='Antiretroviral Naive',2='> 1 but <= 52 weeks of prior antiretroviral therapy',3='> 52 weeks)

symptom - symptomatic indicator (0=asymp, 1=symp)

treat - treatment indicator (0=ZDV only, 1=others)

offtrt - indicator of off-trt before 96+/-5 weeks (0=no,1=yes)

cd40 - CD4 at baseline

cd420 - CD4 at 20+/-5 weeks

cd80 - CD8 at baseline

cd820 - CD8 at 20+/-5 weeks

cid - censoring indicator (1 = failure, 0 = censoring)

\*/

// spajanje na bazu

db = connect("mongodb://localhost/aids");

/\*

Unutar aids\_clinical\_trials\_group\_study\_175\_data\_variable.json je opis svake varijable. Na temelju tih podataka sam odlučio koje je varijabla kategorička, a koja kontinuirana te sam ostavio komentar za svaku varijablu.

\*/

// Definiranje kategorickih i kontinuiranih varijabli

const dataTypes = {

  time: { dataType: "int", variableType: "continuous" },

  trt: { dataType: "int", variableType: "categorical" },

  age: { dataType: "int", variableType: "continuous" },

  wtkg: { dataType: "double", variableType: "continuous" },

  hemo: { dataType: "int", variableType: "categorical" },

  homo: { dataType: "int", variableType: "categorical" },

  drugs: { dataType: "int", variableType: "categorical" },

  karnof: { dataType: "int", variableType: "continuous" },

  oprior: { dataType: "int", variableType: "categorical" },

  z30: { dataType: "int", variableType: "categorical" },

  zprior: { dataType: "int", variableType: "categorical" },

  preanti: { dataType: "int", variableType: "continuous" },

  race: { dataType: "int", variableType: "categorical" },

  gender: { dataType: "int", variableType: "categorical" },

  str2: { dataType: "int", variableType: "categorical" },

  strat: { dataType: "int", variableType: "categorical" },

  symptom: { dataType: "int", variableType: "categorical" },

  treat: { dataType: "int", variableType: "categorical" },

  offtrt: { dataType: "int", variableType: "categorical" },

  cd40: { dataType: "int", variableType: "continuous" },

  cd420: { dataType: "int", variableType: "continuous" },

  cd80: { dataType: "int", variableType: "continuous" },

  cd820: { dataType: "int", variableType: "continuous" },

  cid: { dataType: "int", variableType: "categorical" },

};

/\*

ZADATAK 1

Opis zadatka: Sve nedostajuće vrijednosti kontinuirane varijable zamijeniti sa -1, a kategoričke sa „empty“.

\*/

/\*

DOKUMENTACIJA

<https://www.mongodb.com/docs/manual/aggregation/>

<https://www.mongodb.com/docs/manual/reference/method/db.collection.updateMany/>

<https://www.mongodb.com/docs/manual/reference/operator/query/in/>

\*/

let dataType = "";

for (const field in dataTypes) {

  dataType = dataTypes[field];

/\*

ovaj dio koda ažurira sve dokumente u kolekciji clinical\_trials za kontinuirane varijable gdje je određeno polje null ili prazan string i postavlja vrijednost tog polja na -1. U ovom slučaju ne bi trebalo doci do promjene pošto u Data Setu nemamo missing values.

\*/

  if (dataType.variableType === "continuous") {

    db.clinical\_trials.updateMany(

      // definiranje kriterija pretraživanja, provjeravamo da li su vrijednost polja null ili "" prazan string

      { [`${field}`]: { $in: [null, ""] } },

      // dokumenti koji zadovoljavaju uvjet, njima ćemo dodijeliti vrijednost -1

      [{ $set: { [`${field}`]: -1 } }],

    );

  } else {

    /\*

    ovaj dio koda ažurira sve dokumente u kolekciji clinical\_trials za kategoricke varijable gdje je određeno polje null

    ili prazan string i postavlja vrijednost tog polja na "empty".

    U ovom slučaju ne bi trebalo doci do promjene pošto u Data Setu nemamo missing values.

    \*/

    db.clinical\_trials.updateMany(

      { [`${field}`]: { $in: [null, ""] } },

      [{ $set: { [`${field}`]: "empty" } }],

    );

  }

}

/\*

Napravljeno za Zadatak 1:

Kod je napisan da se za nedostajuće vrijednosti kontinuiranih varijabla postavi na vrijednost -1, a za kategoričke varijable da se vrijednost postavi na "empty". Ovaj Data set nije imao no missing value tako da nije došlo do promijene unutar podataka.

\*/

/\* ZADATAK 2

Opis zadatka: Za svaku kontinuiranu (continuous) vrijednost izračunati srednju vrijednost, standardnu devijaciju i kreirati novi dokument oblika sa vrijednostima, dokument nazvati:  statistika\_ {ime vašeg data seta}. U izračun se uzimaju samo nomissing  vrijednosti.

\*/

/\*

DOKUMENTACIJA

<https://www.mongodb.com/docs/manual/aggregation/>

<https://www.mongodb.com/docs/manual/reference/operator/aggregation-pipeline/#std-label-aggregation-pipeline-operator-reference>

<https://www.mongodb.com/docs/manual/reference/operator/query/gt/>

\*/

for (const field in dataTypes) {

  dataType = dataTypes[field];

  if (dataType.variableType === "categorical") {

    continue;

  }

  db.clinical\_trials.aggregate([

 // Prva faza - filtriranje dokumenata koji zadovoljavaju određeni uvjet, odnosno da je vrijednost gt veća od -1

    { $match: { [`${field}`]: { $gt: -1 } } },

    // Druga faza- grupiranje dokumenata prema polju \_id.

    {

      $group: {

        \_id: `${field}`,

        // Računanje srednje vrijednosti, sd i broj nomissingelemenata

        srednjaVrijednost: { $avg: "$" + [`${field}`] },

        standardnaDevijacija: { $stdDevSamp: "$" + [`${field}`] },

        brojNomissingElemenata: { $count: {} },

      },

    },

  // Treća faza - spajanje rezultate prethodnih faza u novu kolekciju "statistika\_clinical\_trials"

    {

      $merge: {

        into: { db: "aids", coll: "statistika\_clinical\_trials" },

        // polje koje se koristi za usporedvu dokumenta prilikom merganja

        on: "\_id",

        // replace-a kada se dokumenti podudaraju

        whenMatched: "replace",

        // insertira kada se ne podudaraju

        whenNotMatched: "insert",

      },

    },

  ]);

}

/\*

Napravljeno za Zadatak 2:

Za svaku kontinuiranu varijablu izračunate su srednja vrijednost, standardna devijacija i broj nomissing elementa.

Rezultati su spremljeni u novu kolekciju naziva "statistika\_clinical\_trials".

Možemo vijdeti da za svaku varijablu imamo brojNomissingElemenata: 2139 što znači da nismo imali missing elemente u data setu.

\*/

Slika ut MongoDB

A screenshot of a computer

Description automatically generated

// ZADATAK 3

// 3. Za svaku kategoričku  vrijednost izračunati frekvencije pojavnosti po obilježjima varijabli i kreirati novi dokument koristeći nizove,

// dokument nazvati:  frekvencija\_ {ime vašeg data seta} . Frekvencije računati koristeći $inc modifikator.

// prolazimo kroz svaki doc u osnovnoj kolekciji i u support2 kolekciji incrementamo pojavnost te vrijednosti

// svaka kategoricka varijabla ima svoj dokument - \_id je naziv varijable, ostali property-ji su unique vrijednosti i broj pojavljivanja u cijeloj kolekciji

// slozeno ovako jer u zadatku pise da koristimo $inc - moze se napraviti daleko brze koristeci agregaciju i $sum

// <https://www.mongodb.com/docs/manual/reference/method/db.collection.find/>

// <https://www.mongodb.com/docs/manual/reference/method/cursor.forEach/>

// Obrišemo sve postojeće dokumente u kolekciji frekvencija\_clinical\_trials kako se s ponovnim pokretanjem skripte ne bi duplicirali podaci

db.frekvencija\_clinical\_trials.deleteMany({});

db.clinical\_trials.find().forEach(function (trial) {

  for (const field in dataTypes) {

    dataType = dataTypes[field];

    // Provjerava je li tip trenutnog polja "categorical" (kategorički)

    if (dataType.variableType === "categorical") {

      // Ažurira ili stvara dokument u kolekciji frekvencija\_clinical\_trials gdje je \_id jednak imenu polja

      db.frekvencija\_clinical\_trials.updateOne(

        {

          \_id: `${field}`,

        },

        // Inkrementira brojčanu vrijednost za 1

        { $inc: { [trial[field]]: 1 } },

        // Upsert kreira novi dokument ako trenuti field ne postoji

        { upsert: true }

      );

    }

  }

});

/\*

Napravljeno za Zadatak 3:

Za kategoričke varijable analizirali smo pojavnost svih vrijednosti kategoričkih varijabla u kolekciji clinical\_trials.

Ukupan zbroj frekvencija za svaku kategoricku varijable iznosi 2139.

\*/

Slika iz MongoDB-a

A screenshot of a computer

Description automatically generated

//ZADATAK 4

// 4. Iz osnovnog  dokumenta kreirati dva nova dokumenta sa kontinuiranim vrijednostima u kojoj će u prvom dokumentu

// biti sadržani svi elementi <= srednje vrijednosti , a u drugom dokumentu biti sadržani svi elementi >srednje vrijednosti,

// dokument nazvati:  statistika1\_ {ime vašeg data seta} i  statistika2\_ {ime vašeg data seta}

/\*

Iz kolekcije statistika\_clinical\_trials dohvaćamo srednju vrijednost za variablu "time",

\*/

var averageTime = db.statistika\_clinical\_trials.findOne({

  \_id: "time",

}).srednjaVrijednost;

/\*

filtrirali smo dokumente gdje je vrijednost srednje vrijednosti varijable 'time' manja ili jednaka srednjoj vrijednosti

\*/

db.clinical\_trials.aggregate([

  { $match: { time: { $lte: averageTime } } }, //lte označava manje ili jednako

  { $out: "statistika1\_clinical\_trials" }, // rezultati se spremaju u novu kolekciju

]);

/\*

filtrirali smo dokumente gdje je vrijednost srednje vrijednosti varijable 'time' veća od srednje vrijednosti

\*/

db.clinical\_trials.aggregate([

  { $match: { time: { $gt: averageTime } } }, //gt označava veće od

  { $out: "statistika2\_clinical\_trials" }, // rezultati se spremaju u novu kolekciju

]);

/\*

Podaci su podijeljeni u dvije nove kolekcije na temelju srednje vrijednosti odabrane kontinuirane varijable "time". statistika1\_clinical\_trials kolekcija sadrži elemente manje  ili jednake (<=879.09\*)srednjoj vrijednosti, a druga statistika2\_clinical\_trials one veće od srednje vrijednosti (>879.09\*).

\*/

Slika iz MongoDB-a pokazuje kolekciju statistika1 gdje su spremljeni rezultati manji ili jednaki od srednje vrijednosti 879.09\*

A screenshot of a computer

Description automatically generated

Slika iz MongoDB-a pokazuje kolekciju statistika2 gdje su spremljeni rezultati veći od srednje vrijednosti 879.09\*

A screenshot of a computer

Description automatically generated

// ZADATAK 5

// 5. Osnovni  dokument  kopirati u novi te embedati vrijednosti iz tablice 3 za svaku kategoričku vrijednost, :  emb\_ {ime vašeg data seta}

// Kopiranje osnovne kolekcije u novu kolekciju i pretvaramo ih u array

db.clinical\_trials.aggregate([{ $out: "emb\_clinical\_trials" }]);

// Dohvaćanje frekvencijskih podataka:

const frequencies = db.frekvencija\_clinical\_trials.find({}).toArray();

const freqMap = {};

// Za svaki element u dohvaćenom nizu, mapiramo '\_id' na cijeli dokument, kreirajući mapu frekvencija

frequencies.forEach((element) => (freqMap[element.\_id] = element));

for (const field in dataTypes) {

  dataType = dataTypes[field];

  if (dataType.variableType === "categorical") {

    db.emb\_clinical\_trials.updateMany(

      // selekcija se primjenju na sve dokuemente unutar kolekcije bez ograničenja

      {},

      [{ $set: { [`${field}`]: freqMap[field] } }] //zamjenjujemo originalne vrijednosti kategoričkih varijabli s njihovim frekvencijskim podacima.

    );

  }

}

/\*

Napravljeno u zadatku 5:

Kopirali smo osnovnu kolekciju u novu kolekciju emb\_clinical\_trials. Zatim smo dohvatili frekvencijske podatke koje smo dobili

u zadataku 3. Na kraju embedamo vrijednosti iz prethodnog zadatka 3 za svaku kategoričku varijablu

\*/

A screenshot of a computer

Description automatically generated

ZADATAK 6

6.Osnovni  dokument  kopirati u novi te embedati vrijednosti iz tablice 2 za svaku kontinuiranu  vrijednost kao niz :  emb2\_ {ime vašeg data seta}

\*/

// Kopiranje osnovne kolekcije u novu kolekciju

db.clinical\_trials.aggregate([{ $out: "emb2\_clinical\_trials" }]);

// Dohvaćanje statističkih podataka iz kolekcije statistika\_clinical\_tirals iz zadatka 2

const stats = db.statistika\_clinical\_trials.find({}).toArray();

const statMap = {};

// Za svaki element u dohvaćenom nizu, mapiramo '\_id' na cijeli dokument kreirajući mapu statistickih podataka

stats.forEach((element) => (statMap[element.\_id] = element));

for (const field in dataTypes) {

  dataType = dataTypes[field];

  if (dataType.variableType === "continuous") {

    // gledamo za kontunirane varijable

    db.emb2\_clinical\_trials.updateMany(

      // selekcija se primjenju na sve dokuemente unutar kolekcije bez ograničenja

      {},

      [{ $set: { [`${field}`]: statMap[field] } }] //zamjenjujemo originalne vrijednosti kontuniranih varijabli s njihovim statističkim podacima.

    );

  }

}

/\*

Napravljeno u zadatku 6:

Kopirali smo osnovnu kolekciju u novu kolekciju emb2\_clinical\_trials. Zatim smo dohvatili statističke podatke koje smo dobili u zadataku 2. Na kraju embedamo "\_id", "srednjaVrijednost", "standardnaDevijacija" i "brojNomissingElemenata" iz zadatka 2 za svaku kontinuiranu varijablu. Slično kao i prethodni zadatak samo ovdje gledako kontinuirane varijable i statističke podatke,a tamo smo gledali kategoričke varijable i frekvencijske podatke.

\*/

Slike iz MongoDB-a.

A screenshot of a computer

Description automatically generated

// ZADATAK 7

//7.  Iz tablice emb2 izvući sve one srednje vrijednosti iz nizova

// čija je standardna devijacija 10% > srednje vrijednosti koristeći $set modifikator

// Pretpostavljam da se mislilo na uvjet (standardna devijacija) > (10% \* srednja vrijednost) i tako ću računati

db.emb2\_clinical\_trials.aggregate([

  {

    $set: {

      // Dodajte ili promijenite polja unutar svakog dokumenta

      document: {

        $filter: {

          input: { $objectToArray: "$$ROOT" },

          as: "item",

          cond: {

            $and: [

              { $eq: [{ $type: "$$item.v" }, "object"] }, // Provjerite je li vrijednost tipa objekt

              {

                $gt: [

                  "$$item.v.standardnaDevijacija",

                  { $multiply: ["$$item.v.srednjaVrijednost", 1.1] },

                ],

              },

            ],

          },

        },

      },

    },

  },

  {

    $unwind: "$document", // Razdvajanje dokumentata na osnovu polja dokument

  },

  {

    $group: {

      \_id: "$\_id",

      document: { $push: "$document" }, // Grupirajte razdvojene dokumente natrag u niz

    },

  },

  {

    $replaceRoot: { newRoot: { $arrayToObject: "$document" } }, // Zamijenite korijen dokumenta s novim objektom

  },

  {

    $merge: {

      into: "zad7", // Spremanje rezultata u kolekciju

      on: "\_id", // Polje po kojem se spajaju dokumenti

      whenMatched: "replace", // Replace kada se pronađu podudarenih dokumenti

      whenNotMatched: "insert", // Insert kada nema podudarenih dokumenata

    },

  },

]);

// Pošto se u zadataku traži da se vrati samo srednja vrijednost, uklanjamo iz objekata polja \_id,

// standardnaDevijacija i brojNomissingElemenata

// Povezivanje na kolekciju 'zad7' i iteracija kroz kolekciju

db.zad7.find().forEach(function (doc) {

  // Inicijalizacija objekta

  let updateObject = {};

  for (let key in doc) {

    // Uklanjamo polja "\_id", "standardnaDevijacija" i "brojNomissingElemenata"

    updateObject[key + ".\_id"] = "";

    updateObject[key + ".standardnaDevijacija"] = "";

    updateObject[key + ".brojNomissingElemenata"] = "";

  }

  // update

  if (Object.keys(updateObject).length > 0) {

    db.zad7.updateOne({ \_id: doc.\_id }, { $unset: updateObject });

  }

});

/\*

Napravljeno u zadatku 7:

Iz kolekcije "emb2\_clinical\_trials" izvučene su srednje vrijednosti iz nizova čija je standardna devijacija

veća od 10% srednje vrijednosti, koristeći $set modifikator te su rezultati pohranjeni u kolekciju zad7.

\*/

A screenshot of a computer

Description automatically generated

/\*

ZADATAK 8

8. Kreirati složeni indeks na originalnoj tablici i osmisliti upit koji je kompatibilan sa indeksom

kreiramo jednostavan slozeni index i napravimo explain() upita koji ga koristi

(explain mora vratiti naziv indexa koji se koristi)

<https://www.mongodb.com/docs/manual/core/indexes/index-types/index-compound/>

\*/

db.clinical\_trials.createIndex({

  // vrijednost 1 označava uzlazni poredak za "time" i "age"

  time: 1,

  age: 1,

});

//PRVI QUERY

// db.clinical\_trials.explain().find({ time: { $gt: 948 }, age: 45 });

/\*

ova linije se upisuju u terminal mongosh

explain() pruža detaljan izvještaj o tome kako je upit izvršen s informacijama da li je

index korišten. Pošto obje varijable koriste index pisat će pod stage: 'IXSCAN', i naziv tog

indeksa kojeg smo kreiraali indexName: 'time\_1\_age\_1'

\*/

A computer screen shot of a program code

Description automatically generated

// DRUGI QUERY

// db.clinical\_trials.explain().find({ trt: 2, race: 0 });

/\*

Ovaj upit neće efikasno koristiti kreirani složeni indeks jer traži dokumente na

osnovu varijabli "trt" i "race" koji nisu indeksirani. Pod stage će pisati stage: 'COLLSCAN'

\*/

A black screen with green text

Description automatically generated

/\*

Napravljeno u zadatku 8:

Kreirali smo složeni indeks na varijablama "time" i "age" s uzlaznim poredkom.

Složeni indeksi omogućavaju bazi da efikasno izvršava upite koji koriste ove varijable za filtriranje, sortiranje ili i jedno i drugo, a samim time se poboljšavaju performanse query-a.

S explain() metodom možemo analizirati kako baza izvršava query-e i potvrditi koristi li se naš indeks.

Za prvi query koji koristi varijable definirana u indeksu "time" i "age", explain() će prikazati 'IXSCAN' i ime indeksa 'time\_1\_age\_1', što nam pokazuje na to da je indeks korišten.

Za drugi query koji ne koristi varijable uključene u indeks, rezultat explain() će  pokazati na 'COLLSCAN', što znači da baza mora pregledati cijelu kolekciju da bi našao odgovarajuće dokumente.

Ovakav pristup je manje efektivan.

\*/