Analiza utjecaja promatranih varijabli na bolest srca

Vukušić, Katarina

SVEUČILIŠTE U SPLITU
PRIRODOSLOVNO MATEMATIČKI FAKULTET

ZAVRŠNI RAD

ANALIZA UTJECAJA PROMATRANIH VARIJABLI NA BOLEST SRCA

Katarina Vukušić

Split, rujan 2022.

Temeljna dokumentacijska kartica

Završni rad

Sveučilište u Splitu

Prirodoslovno-matematički fakultet

Odjel za informatiku

Ruđera Boškovića 33, 21000 Split, Hrvatska

ANALIZA UTJECAJA PROMATRANIH VARIJABLI NA BOLEST SRCA

Katarina Vukušić

SAŽETAK

U ovome završnom radu su objašnjeni osnovni pojmovi kao što je umjetna inteligencija i strojno učenje. Sadrži i opis klasifikacije i logističke regresije, jezika Python, virtualnog okruženja te Streamlit okvira koji se koriste za izradu aplikacije. Nakon teorijskog dijela, opisan je postupak izrade aplikacije koja čini praktični dio završnog rada. Može se vidjeti način pripremanja podataka, proces modeliranja, evaluacija modela te na kraju i sama izrada aplikacije u Streamlit okviru.

Ključne riječi: model, logistička regresija, skup podataka, atributi, strojno učenje, klasifikacija, vizualizacija, modeliranje, evaluacija, CHD

Rad je pohranjen u knjižnici Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Rad sadrži: 36 stranica, 22 grafičkih prikaza, 1 tablica i 24 literaturnih navoda. Izvornik je na hrvatskom jeziku.

Mentor: **Dr. sc. Divna Krpan,** *docent Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Ocjenjivači: **Dr. sc. Divna Krpan,** *docent Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Dr. sc. Saša Mladenović, *izvanredni profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Dr. sc. Goran Zaharija, *docent Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Rad prihvaćen: **Rujan 2022**

Basic documentation card

Thesis

University of Split

Faculty of Science

Department of informatics

Ruđera Boškovića 33, 21000 Split, Croatia

ANALYSIS OF THE EFFECT OF THE OBSERVED VARIABLES ON HEART DISEASE

Katarina Vukušić

ABSTRACT

In this thesis, terms such as artificial intelligence and machine learning are explained. It also contains a description of classification and logistic regression, the Python language, the virtual environment and the Streamlit framework which are used to create application. After the theoretical part, the process of creating application, which forms the practical part of this thesis is described. You can see the method of data preparation, modeling process, model evaluation and finally the creation of the application in the Streamlit framework.

Key words: model, logistic regression, data set, attributes, machine learning, classification, visualisation, modelling, evaluation, CHD

Thesis deposited in library of Faculty of science, University of Split

Thesis consists of: 36 pages, 22 figures, 1 table and 24 references

Original language: Croatian

Mentor: **Divna Krpan, Ph.D.** *Assistant Professor of Faculty of Science, University of Split*

Reviewers: **Divna Krpan, Ph.D.** *Assistant Professor of Faculty of Science, University of Split*

Saša Mladenović, Ph.D. *Associate Professor of Faculty of Science, University of Split*

Goran Zaharija, Ph.D. *Assistant Professor of Faculty of Science, University of Split*

Thesis accepted: **September 2022**

ZAHVALA

Zahvaljujem se svojoj mentorici Dr. sc. Divni Krpan na susretljivosti i savjetima koji su bili od velike pomoći pri izradi završnog rada.

Zahvaljujem se kolegama i prijateljima koji su mi ovo razdoblje života učinili posebnim.

Najveća hvala mojim roditeljima i bližnjima na iznimnoj podršci i razumijevanju s čime su pridonijeli mom uspjehu.

Sadržaj:

Uvod

U posljednja tri desetljeća, povećana smrtnost od kardiovaskularnih bolesti (engl. Cardiovascular disease) primijećena je u industrijaliziranim i tranzicijskim zemljama. Hrvatska se smatra da je u trećoj fazi epidemiološke tranzicije, definirane kao one s 35%-65% ukupnog mortaliteta od kardiovaskularnih bolesti. Najvažnija dijagnoza kardiovaskularnih bolesti je koronarna bolest srca (engl. Coronary Heart Disease, skraćeno CHD). U svijetu, otprilike 80%-90% bolesnika sa simptomatskom koronarnom bolesti srca i više od 95% pacijenata koji su umrli od te bolesti imalo je barem jedan od četiri tradicionalnih čimbenika rizika (pušenje, hipertenzija, hiperlipidemija i dijabetes). [1] Kako bi se podigla svijest o ovoj bolesti, korisno je izraditi model logističke regresije koji će predvidjeti imamo li 10-godišnji rizik od koronarne srčane bolesti.

U ovome radu promotrit će se skup podataka s čimbenicima rizika od koronarne srčane bolesti te priprema i modeliranje navedenih podataka za model logističke regresije. Logistička regresija je vrsta modela strojnog učenja koja procjenjuje vjerojatnost događaja, u ovome slučaju procjenjuje ima li osoba 10-godišnji rizik od razvoja koronarne srčane bolesti ili nema. Također će se promotriti vizualizacija podataka te odnosi pojedinih čimbenika s rizikom.

1 Umjetna inteligencija

Umjetna inteligencija (engl. Artificial intelligence, skraćeno AI) je sposobnost određenog uređaja da oponaša ljudske aktivnosti. Ona omogućuje tehničkim sustavima percipiranje kruženja te rješava probleme kako bi postigla neki cilj.

Računalo prima podatke te ih obrađuje i daje odgovore na njih. Neke tehnologije umjetne inteligencije su prisutne već 50 godina, ali napredak u računalnoj snazi, dostupnost velike količine podataka te novi algoritmi posljednjih su godina doveli do velikih otkrića u tom području.

Predviđeno je da će umjetna inteligencija donijeti iznimne promjene u budućnosti, no već je prisutna i u našim svakodnevnim životima. [2]

Umjetna inteligencija se dijeli na dvije vrste:

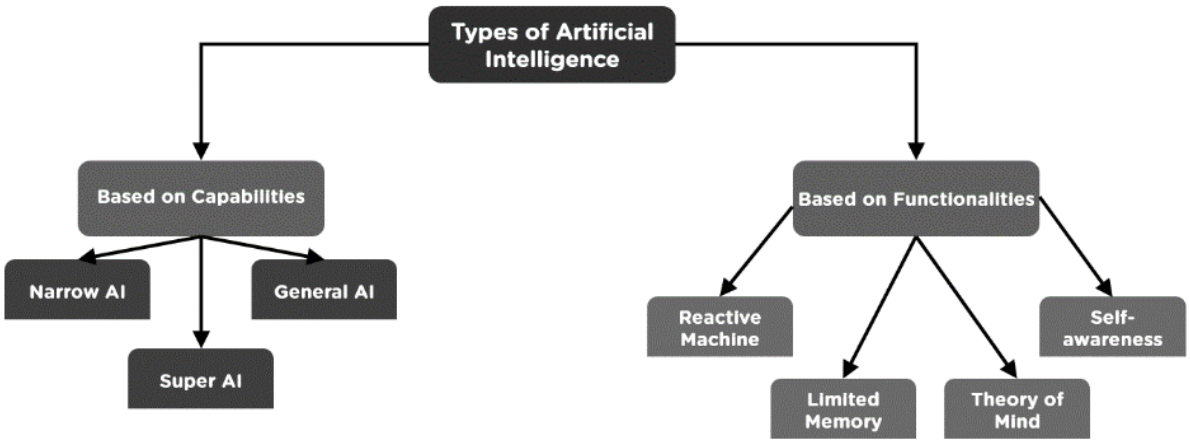
- **Tip 1 (engl. Type 1) -** Na temelju sposobnosti
- **Tip 2 (engl. Type 2)**- Na temelju funkcionalnosti

Postoje tri vrste umjetne inteligencije Tipa 1 [3]:

- **Umjetna Uska Inteligencija (engl. Artificial Narrow Intelligence, skraćeno ANI) :** sve postojeće aplikacije s AI spadaju u ovu kategoriju. Ona uključuje sustav koji može obavljati usko definirane specifične zadatke poput ljudi. Jedina mana je što strojevi ne mogu obavljati zadatke za koje nisu unaprijed isprogramirani.
- **Umjetna Opća Inteligencija (engl. Artificial General Intelligence, skraćeno AGI) :** sadrži sposobnost treniranja, učenja, razumijevanja i obavljanja funkcija poput čovjeka. Sustavi će imati višenamjenske mogućnosti koje se protežu kroz različite domene. Reagirat će i improvizirati baš poput ljudi dok se suočavaju s različitim scenarijima.
- **Umjetna Super Inteligencija (engl. Artificial Super Intelligence, skraćeno ASI):** najmoćniji oblik inteligencije. Smatra se da će biti najviša točka razvoja umjetne inteligencije. Imat će mogućnost obavljati sve zadatke bolje od ljudi zbog svoje neumjereno superiorne obrade podataka, sposobnosti donošenja odluka i pamćenja.

Postoje četiri vrste umjetne inteligencije Tipa 2 [4]:

- **Reaktivna umjetna inteligencija (engl. Reactive AI) :** najosnovniji tip umjetne inteligencije koja je isprogramirana na pružanje predvidljivog izlaza na temelju podataka koje prima. Ona je činila ogroman napredak u povijesti umjetne inteligencije, ali ta vrsta umjetne inteligencije ne može funkcionirati na zadacima koji nisu dizajnirani za njih. To ih čini ograničenima i zrelima za poboljšanje.
- **Umjetna inteligencija s ograničenom memorijom (engl. Limited Memory AI):** najrašireniji tip umjetne inteligencije. Ona uči iz povijesti i izgrađuje eksperimentalno znanje tako što promatra radnje ili podatke. Ona koristi povijesne promatrane podatke u kombinaciji s unaprijed programiranim informacijama za izradu predviđanja te izvođenja složenih zadataka.
- **Teorija uma umjetne inteligencije (engl. Theory of mind AI):** ovom tipu umjetne inteligencije je cilj stvoriti emocionalno inteligentne robote s kojima ćemo imati mogućnost voditi smislene razgovore. Strojevi će steći istinske sposobnosti donošenja odluka koje su slične ljudima. Oni će moći razumjeti i zapamtiti emocije te prilagoditi svoje ponašanje na temelju tih emocija dok komuniciraju s ljudima.
- **Samosvjesna UI (engl. Self-aware AI) :** najnaprednija vrsta umjetne inteligencije. Strojevi će biti svjesni svojih unutarnjih emocija i mentalnih stanja, te će donositi zaključke na temelju njih. To će im omogućiti da imaju razinu svijesti i inteligencije sličnu ljudskim bićima.



Sl. 1.1 Podjela umjetne inteligencije, preuzeto s [5]

2 Strojno učenje

Strojno učenje (engl. Machine Learning) je način postizanja umjetne inteligencije. Odnosi se na primjenu umjetne inteligencije koja sustavima omogućuje učenje i napredovanje na temelju iskustva. Usmjereno je na razvoj računalnih programa koji mogu pristupiti podacima te ih koristiti za vlastito učenje.

Postoje 4 vrste strojnog učenja:

- Nadzirano učenje (engl. Supervised learning)
- Nenadzirano učenje (engl. Unsupervised learning)
- Podržano učenje (engl. Reinforcement learning)
- Djelomično nadzirano učenje (engl. Semi-supervised learning)

Metode strojnog učenja se optimiraju i prilagođavaju svakom problemu tj. skupu podataka kako bi se postigli najbolji mogući rezultati. Optimizacija i prilagodba metoda većinom se provode:

- Pronalaskom najboljih hiperparametara modela
- Izbor najboljeg modela za odabrani problem

Strojno učenje je usko povezano s nekoliko područja:

- Umjetna inteligencija
- Dubinska analiza podataka
- Statistika
- Teorija informacija

Također je povezana i s mnogim drugim primjenskim područjima poput robotike, računalnog vida, medicinske informatike i biomedicinskog inženjerstva... [6]

2.1 Nadzirano strojno učenje

Nadzirano strojno učenje uzima ono što je naučilo iz prošlosti i primjenjuje to na nove podatke pomoću označenih primjera za predviđanje budućih obrazaca. Njegovo učenje se temelji na eksplicitnom primjeru. Nadzirano učenje zahtijeva da mogući izlazi algoritma budu poznati i da podaci koji su korišteni za treniranje algoritma budu označeni točnim odgovorima. Pristup nadziranom učenju je takav da se prezentiraju sve informacije potrebne za donošenje unaprijed određenih zaključaka.

Algoritam uči uspoređujući svoj stvarni izlaz s već unaprijed zapisanim točnim izlazom kako bi pronašao pogreške. Nakon toga se modificira model koji će biti u skladu s tim. [7]

Nadzirano učenje se dalje dijeli na:

- Klasifikacija
- Regresija

2.1.1 Klasifikacija

Klasifikacija ili razvrstavanje je vrsta nadziranog učenja koja služi za predviđanje diskretnih odziva pri čemu se ulazni podaci razvrstavaju u kategorije. Intuicija klasifikacije je izrazito očita, ona za dane primjere predviđa kojoj klasi oni pripadaju. Jednostavan primjer klasifikacije je klasifikacija elektroničkih pošta kao „poželjna pošta“ ili „neželjena pošta“. Poželjna i neželjena pošta čine oznake klasa i moraju biti preslikane prema numeričkim vrijednostima prije nego što se daju algoritmu za modeliranje. Taj proces se naziva label encoding gdje se svakoj oznaci klase dodjeljuje jedinstveni cijeli broj. Tako bi mogli neželjenu poštu označiti kao 0, a željenu poštu kao 1.

Klasifikacijski algoritmi ocjenjuju se na temelju njihovih rezultata. Najkorištenija metrika za procjenu izvedbe modela je točnost klasifikacije (engl. Classification accuracy).

Ona procjenjuje izvedbu modela na temelju predviđenih oznaka klasa. Možemo reći da postoje 4 glavne vrste klasifikacijskih zadataka s kojima se možemo susresti:

- Binarna klasifikacija (engl. Binary classification)
- Multi klasna klasifikacija (engl. Multi-class classification)
- Klasifikacija s više oznaka (engl. Multi-label classification)
- Neuravnotežena klasifikacija (engl. Imbalanced classification)

Pošto se koristi model logističke regresije koja spada pod binarnu klasifikaciju, objasnit ćemo samo binarnu klasifikaciju.

Binarna klasifikacija se odnosi na one zadatke koji imaju dvije oznake klasa. Zadaci binarne klasifikacije sadrže jednu klasu koja je normalno stanje i drugu klasu koja je nenormalno stanje.

Nastavljajući se na primjer elektroničke pošte, „poželjna pošta“ pripada normalnom stanju dok „nepoželjna pošta“ pripada nenormalnom stanju. Klasi normalnog stanja dodjeljuje se oznaka klase 0, a klasi nenormalnog stanja dodjeljuje se oznaka 1. [8]

Atribut *TenYearCHD* koji se koristi u ovome radu sadržava jednu klasu u normalnom stanju, dok je druga klasa u nenormalnom stanju. Njegova klasa normalnog stanja je ta da ispitanik nema 10-godišnji rizik od koronarne srčane bolesti i označena je s 0. Stoga, klasa nenormalnog stanja je da ispitanik ima 10-godišnji rizik od koronarne srčane bolesti i označena je s 1.

Neki od algoritama koji se mogu iskoristiti za binarnu klasifikaciju su:

- Logistička regresija
- K-najbliži susjedi
- Stablo odlučivanja

Kako bi odlučili koji od navedenih algoritama iskoristiti, najbolje je odrediti točnost pojedinog algoritma i uzeti onaj s najboljim rezultatom. Stoga treba provjeriti koji algoritam ima najbolji rezultat na skupu naših podataka.

Prvo se skup podataka mora podijeliti na dva niza:

- Niz x: nezavisne varijable koje će služiti za predviđanje izlazne varijable
- Niz y: zavisna varijabla koja se predviđa s x nizom.

Zatim trebamo podijeliti podatke na skup za treniranje i skup za testiranje. To radi na principu slučajne podijele podataka u skupove treniranja i testiranja.

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.25, random_state=5)
```

Sl. 2.1 Podjela podataka na skup za treniranje i testiranje

Kada je podjela napravljena, treba izraditi svaki algoritam i usporediti rezultate.

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
```

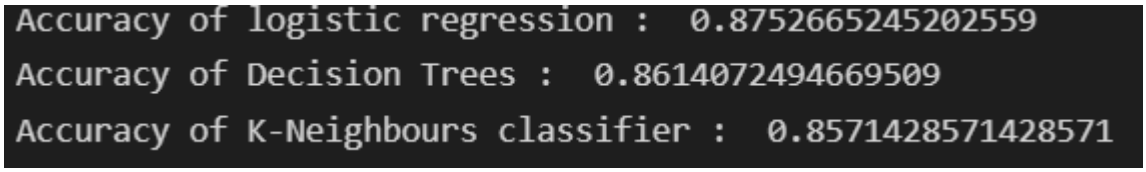
```
from sklearn.tree import DecisionTreeClassifier

decTree = DecisionTreeClassifier(max_depth=6, random_state=0)
decTree.fit(x_train,y_train)
y_pred_DT = decTree.predict(x_test)
```

```
from sklearn.neighbors import KNeighborsClassifier

knc =  KNeighborsClassifier()
knc.fit(x_train,y_train)
y_pred_KNC = knc.predict(x_test)
```

Sl. 2.2 Implementacija logističke regresije, stabla odlučivanja i k-najbliži susjedi



Sl. 2.3 Rezultat točnosti pojedinog algoritma

Možemo zaključiti kako logistička regresija ima najbolju točnost koja iznosi 87.5%. Algoritam k-najbližih susjeda ima točnost 85,7% dok stablo odlučivanja ima točnost 86,1%. Zbog toga ćemo uzeti logističku regresiju za predviđanje 10-godišnjeg rizika od CHD.

2.1.1.1 Logistička regresija

Logistička regresija (engl. Logistic regression) je algoritam binarne klasifikacije iz nadziranog strojnog učenja koji se koristi za predviđanje zavisne varijable. Priroda zavisne varijable je dihotomna što znači da postoje samo dvije moguće klase. Zavisna varijabla je binarne prirode s podacima kodiranim kao 1 (označava „Da“) ili 0 (označava „Ne“). Ona spada u jedan od najjednostavnijih algoritama strojnog učenja koji se može iskoristiti za razne probleme klasifikacije. Uz binarne ciljne varijable mogu postojati još dvije kategorije zavisnih varijabli. [9]

Logistička regresija može se podijeliti na 3 kategorije:

- Binarna – to je ujedno i osnovna logistička regresija
- Multinomijalna– zavisna varijabla može imati 3 ili više mogućih nesređenih tipova (tipovi bez kvantitativnog značenja)
- Ordinalna – zavisna varijabla može imati 3 ili više mogućih poredanih tipova (tipovi s kvantitativnim značenjem)

3 Okruženja i alati korišteni pri izradi aplikacije

3.1 Python

Python je visokorazinski interpretirani programski jezik. Izumio ga je Guido Van Rossum 1991. godine. Njegova srž dizajna je naglašavanje čitljivosti koda uz korištenje značajnog uvlačenja. Python je dinamičan i podržava više programskih paradigmi (uklj. strukturirano, objektno orijentirano i funkcionalno programiranje). Vrlo često se opisuje kao „batteries included“ jezik zbog svoje opsežne standardne biblioteke. Rangiran je kao jedan od najpopularnijih programskih jezika.

Filozofija mu je sažeta u dokumentu The Zen of Python (PEP 20), koji uključuje aforizme kao:

- Lijepo je bolje nego ružno
- Eksplicitno je bolje nego implicitno
- Jednostavno je bolje od složenog
- Složeno je bolje nego komplicirano

Python je dizajniran da bude visoko proširiv putem modula. Ta kompaktna modularnost ga je učinila posebno popularnim kao sredstvo za dodavanje unaprijed programiranih sučelja postojećim aplikacijama. Teži jednostavnijoj, manje pretrpanoj sintaksi i gramatici, dok programerima nudi izbor u njihovoj metodologiji kodiranja. [10]

Kao što je prije navedeno strojno učenje jednostavno prepoznaje obrasce u podacima kako bi moglo samostalno donositi inteligentne odluke. Najprikladniji programski jezik za to je Python isključivo zbog toga što ga je lako razumjeti. Njegova čitljivost, nekompleksnost i mogućnosti brze izrade prototipa čine ga popularnim među programerima diljem svijeta. Python ujedno ima jednostavnu i snažnu implementaciju. Uz to Python sadrži mnoge ugrađene biblioteke koje su isključivo za strojno učenje i općenito umjetnu inteligenciju, te se mogu vrlo jednostavno primijeniti. Čak ako i imamo osnovno znanje o jeziku, već ga možemo koristiti za strojno učenje zbog ogromne količine biblioteka, resursa i alata koji su dostupni. Ujedno ćemo potrošiti manje vremena na pisanje koda i otklanjanje pogrešaka nego da koristimo Java ili C++. [11]

3.2 Jupyter-ova bilježnica

Jupyter-ova bilježnica je IDE otvorenog koda koji se koristi za stvaranje Jupyter dokumenata. To je ujedno i internetsko interaktivno računalno okruženje. Podržava preko 40 programskih jezika uključujući Python, R, Julia i Scala. Bilježnice se mogu dijeliti s drugima preko elektroničke pošte, Dropbox-a, GitHub-a te Jupyter Notebook Viewer-a. U ovome radu koristili smo Jupyter-ovu bilježnicu za učitavanje i pripremanje podataka, stvaranja modela logističke regresije te spremanje modela. [12]

3.3 Streamlit

Streamlit je okvir aplikacije otvorenog koda koji je cijeli pisan u Python-u. Omogućuje izradu i implementaciju moćnih podatkovnih aplikacija u svega nekoliko minuta. Streamlit je objavljen u listopadu 2019 i nedavno ga je kupila kompanija Snowflake. Streamlit je pružio veliko uzbuđenje u zajednici Podatkovne znanosti. Streamlit se fokusira na jednostavnost. On se može postaviti u postojeću skriptu koda strojnog učenja. U osnovi čini taj kod parametriziranim i pretvara ga u prekrasnu aplikaciju. Uz njegovu arhitekturu proširivosti komponenti mogu se izgraditi i integrirati većina web sučelja. [13]

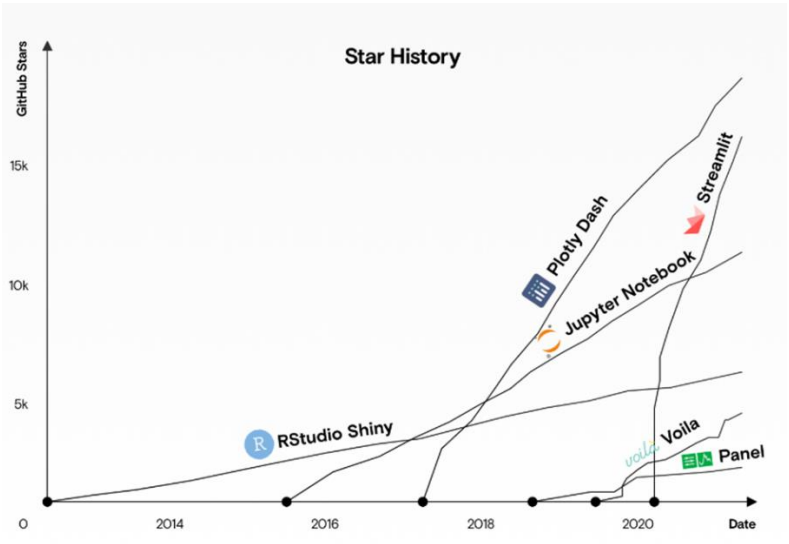
Kompatibilan je s glavnim Python bibliotekama kao što su PyTorch, NumPy, pandas, scikit-learn... Uz Streamlit nije potreban povratni poziv jer se widgeti tretiraju kao varijable. Predmemoriranje podataka pojednostavljuje i ubrzava proračunske cjevovode. Također prati promjene u ažuriranjima povezanog Git repozitorija te će se aplikacija automatski implementirati u dijeljenoj vezi.

Postoje i nedostatci ovog okvira. Može se istaknuti da nema portal za upravljanje mapbox-om i ne sadrži sve funkcije koje on nudi. Također Streamlit-ovo predmemoriranje podataka ne može pratiti promjene podataka koje se događaju izvan tijela funkcije. [14]

3.3.1 Usporedba različitih alata i okvira za nadzornu ploču podataka

Gotovo svaka tvrtka se nalazi na vrijednim podacima kojima interni timovi trebaju pristupiti i analizirati. Netehnički timovi učestalo traže alate kako bi to olakšali.

Timovi žele dinamičke nadzorne ploče pomoću kojih mogu jednostavno pokrenuti upite i vidjeti prilagođene, interaktivne vizualizacije. Postoje različiti okviri i alati koji imaju sličnu namjenu kao i Streamlit-a. Primjeri takvih okvira i alata su: Dash, Panel, Voila, Jupyter i Shiny.



Sl. 3.1 Grafikon popularnosti različitih okvira i alata za Python, preuzeto s [15]

Korisnici na web stranici Github mogu „označiti zvjezdicom“ repozitorije drugih ljudi tako što ih spremaju na svoj popis repozitorija sa zvjezdicom. Neki ih koriste kako bi pokazali da im se sviđa projekt. Github zvjezdice su mjerni podaci koje je lako popratiti i koriste se kao mjerilo koliko je popularan projekt otvorenog koda. S grafikona se jasno vidi kako su u posljednjih tri godine Dash i Streamlit porasli u popularnosti kao sveobuhvatna rješenja za nadzorne ploče.

Dash i Streamlit su najsličnije biblioteke. Iako su oba otvorenog koda, Dash je više fokusiran na poslovno tržište i ne uključuje sve značajke otvorenog koda. Nasuprot tome, Streamlit je potpuno otvorenog koda.

Podatkovna nadzorna ploča sastoji se od različitih komponenti. Potrebna je:

- Analiza (engl. Analyze)– manipuliranje i sažimanje podataka pomoću pozadinske biblioteke (npr. pandas)
- Vizualizacija (engl. Visualize)– stvoriti dijagrame i grafikone podataka pomoću biblioteke grafičkih prikaza (npr. Bokeh)
- Interakcija (engl. Interact) – prihvatiti korisnički unos koristeći biblioteku sučelja (npr. React)
- Posluživanje (engl. Serve) – poslušati zahtjeve korisnika i vratiti web stranicu pomoću web poslužitelja (npr. Flask)

U prošlosti ste morali gubiti znatnu količinu vremena na pisanje „glue“ koda za spajanje komponenti. Uz Streamlit i Dash, te komponente dolaze u jednom paketu. Nisu sve biblioteke izravno usporedive. Na primjer, Dash je izgrađen na Flask-u, a Flask je općenitiji okvir za razvoj web aplikacija.

Svaka se biblioteka usredotočuje na drugačije područje. Tako imamo Streamlit, Dash i Panel koji čine potpuna rješenja za nadzornu ploču, usmjerena na analitiku podataka temeljenu na Python i rade na web okvirima Tornado i Flask. Shiny je više usmjeren na analizu podataka s RStudio-m. U prijašnjem tekstu naveli smo da se Jupyter-ova bilježnica koristi za analizu i mapiranje podataka dok Viola čini biblioteku koja pojedinačne Jupyter-ove bilježnice pretvara u interaktivne web stranice. Panel se može koristiti isto na Jupyter-ove bilježnice i ovisno o potrebama može biti fleksibilniji od Viole.

	Maturity	Popularity	Simplicity	Adaptability	Focus	Language support
Streamlit	C	A	A	C	Dashboard	Python
Dash	B	A	B	B	Dashboard	Python, R, Julia
Panel	C	B	B	B	Dashboard	Python
Shiny	A	B	B	B	Dashboard	R
Voila	C	C	A	C	Dashboard	Python, R, Julia
Jupyter	A	A	B	B	Notebook	Python, R, Julia
Flask	A	A	B	A	Web framework	Python

Sl. 3.2 Usporedba različitih okvira i alata, preuzeto s [15]

Mjerila korištena za usporedbu su:

- **Zrelost (engl. Maturity)** – temelji se na starosti projekta i njegovoj stabilnosti
- **Popularnost (engl. Popularity)** – temelji se na usvajanju i Github zvjezdicama
- **Jednostavnost (engl. Simplicity)** – temelji se na jednostavnosti korištenja
- **Prilagodljivost (engl. Adaptability)** – temelji se na fleksibilnosti i samostalnosti
- **Fokus (engl. Focus) -** temelji se na problemu koji rješava
- **Jezična podrška (engl. Language support)** – s kojim jezicima su podržani

Ovim mjerilima je namjena dati brzi pregled načina na koji se alati preklapaju i kako se međusobno razlikuju. [15]

3.4 Virtualno okruženje

Virtualna okruženje (engl. Virtual environment) je alat za upravljanje ovisnostima i izolaciju projekta. Omogućuje lokalnu instalaciju Python web paketa u izoliranom direktoriju za određeni projekt. [16]

Virtualno okruženje je samo jedan direktorij koji sadrži tri važne komponente :

- Mapa Lib/site-packages – u njoj se nalaze instalirani web paketi
- Simbolne veze (engl. Symlinks) – u njoj se nalaze simbolne veze na Python izvršne datoteke u sustavu
- Skripte – one osiguravaju izvršenje Python koda, koriste web pakete instaliranje unutar danog virtualnog okruženja

3.5 Anaconda Navigator

Anaconda Navigator je desktop grafičko korisničko sučelje (GUI) koje omogućuje pokretanje aplikacija i jednostavno upravljanje conda paketima, okruženjima i kanalima bez korištenja naredbi naredbenog retka.

Navigator pretražuje pakete na Anaconda.org ili u lokalnom Anaconda repozitoriju. Program naredbenog retka conda je ujedno upravitelj paketa i upravitelj okoline. To pomaže znanstvenicima osigurati da sve verzije pojedinog paketa imaju sve ovisnosti koje zahtijeva. [17]

4 Izrada aplikacije

Za izradu aplikacije korišten je programski jezik Python. Okvir pomoću kojeg je napravljena aplikacija je Streamlit. Također je potrebno imati instaliran Anaconda Navigator (kod Windows-a) za instalaciju biblioteka i Streamlit-a te izradu i aktivaciju virtualnog okruženja. Model logističke regresije je izrađen na primjeru modela logističke regresije na internetu. [18]

Aplikacija se sastoji od:

- **Jupyter-ova Bilježnica (engl. Jupyter Notebook) :** izrada modela logističke regresije
- **virtualnog okruženja**
- **dvije datoteke :** framingham.csv (datoteka za model), visualisation.csv (datoteka za vizualizaciju)
- **saved_steps.pkl:** služi za pozivanje modela unutar aplikacije
- **tri stranice za prikaz:** app.py (glavna stranica), prediciton_page.py, visualisation_page.py

4.1 Skup podataka

Skup podataka u slučaju tabličnih podataka odgovara jednoj ili više tablica gdje svaki stupac odgovara određenoj varijabli te svaki redak odgovara odgovarajućem zapisu u skupu podataka. Svaki član skupa podataka sadrži vrijednost za svaku od varijabla. [19] Skup podataka koji se koristi u ovome radu je preuzet sa stranice Kaggle. Skup podataka je prvotno u obliku .csv (datoteka s vrijednostima odvojenim zarezima) i potrebno je preoblikovati u DataFrame.

DataFrame je dvodimenzionalni tablični podaci čija struktura podataka sadrži označene osi (redovi i stupci). Ona se može smatrati kao spremnikom nalik na dict za objekte serije. Kako bi mogli uopće iskoristiti DataFrame moramo napraviti import pandas alata za analizu i manipulaciju podacima.

4.1.1 Framingham-ova studija bolesti srca

Kao skup podataka za izračun rizika od CHD koristila sam podatke iz Framingham-ove studije bolesti srca. To je kontinuirana epidemiološka studija bolesti srca ustanovljena u Framinghamu, Massachusetts, tijekom 1948.-1950. Studija se bavi mjerenjima opsega i razvoja kardiovaskularnih bolesti u presjeku populacije u dobi od 30-69 godina 01.01.1950., te proučavanjem onih okolišnih i osobnih čimbenika koji su povezani s pojavom kardiovaskularnih bolesti i njenim napredovanjem.

Podaci su primarno dobiveni dvogodišnjim pregledima provedenim u klinici koja je isključivo održavana za ovu studiju. Planirano je nastaviti promatranje skupine ispitanika u narednih 20 godina. [20]

Prikupljeno je preko 4000 zapisa o ispitanicima i 16 atributa od kojih je jedan i predviđeni 10-godišnji rizik za razvoj koronarne srčane bolesti.

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	77.0	0
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	76.0	0
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	70.0	0
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	103.0	1
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	85.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	72.0	22.00	84.0	86.0	0
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	87.0	19.16	86.0	NaN	0
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.47	80.0	107.0	0
4238	1	40	3.0	0	0.0	0.0	0	1	0	185.0	141.0	98.0	25.60	67.0	72.0	0
4239	0	39	3.0	1	30.0	0.0	0	0	0	196.0	133.0	86.0	20.91	85.0	80.0	0

4240 rows × 16 columns

Sl. 4.1 Skup podataka Framingham-ove studije

Atributi ujedno čine i ulazne varijable za model logističke regresije. Opširnije pojašnjenje pojedinog atributa:

Atributi	Objašnjenje
Sex	je li ispitanik muškarac ili žena
Age	dob ispitanika
currentSmoker	je li ispitanik pušač ili ne
cigsPerDay	prosječan broj cigareta koje je ispitanik iskoristio u jednome danu
BPMeds	je li ispitanik koristio tablete za tlak
prevalentStroke	je li ispitanik ikada imao moždani udar
prevalentHyp	je li ispitanik ikada bolovao od hipertenzije
diabetes	je li ispitanik imao dijabetes
totChol	ukupna razina kolesterola mjerena u mg/dL
sysBP	sistolički krvni tlak
dialBP	dijastolički krvni tlak
BMI	indeks tjelesne mase
heartRate	otkucaji srca
glucose	razina glukoze mjerena u mg/dL
TenYearCHD	10-godišnji rizik od koronarne srčane bolesti

Tablica 4.1. Atributi skupa podataka

4.2 Priprema podataka

Neki podaci u skupu podataka su nam prikazani u obliku decimalnih brojeva dok su neki prikazani kao cijeli brojevi. Atributi *sex, currentSmoker, prevalentStroke, prevalentHyp, diabetes i TenYearCHD* su zapisani binarno gdje 0 označava Ne dok 1 označava Da.

Kako bi se pripremio skup podataka za modeliranje trebamo ukloniti atribut *education* te preimenovati atribut *sex* u *sex_mal*e. Glavni razlog preimenovanja atributa je što podaci pod atributom *sex* ukazuju je li ispitanik žena (0) ili muškarac (1).

```
df.drop(['education'],axis=1,inplace=True)
df.rename(columns={'male':'sex_male'},inplace=True)
```

Kôd 4. – Uklanjanje atributa *education* i promjena imena atributu *sex*

Također je važno u pripremi podataka provjeriti postoje li neki atributi kojima nedostaje vrijednost.

```
count = 0
for i in df.isnull().sum(axis=1):
        if i > 0:
              count = count+1
print('Total number of rows with missing values is', count)
```

Kôd 2. – Ispis ukupnog broja redova u kojima nedostaje vrijednost

Nakon provođenja ove ćelije izračunato je da nedostaje ukupno 489 vrijednosti. Pošto taj broj čini vrlo mali dio skupa podataka, možemo izbaciti te podatke iz skupa.

```
df.dropna(axis=0, inplace=True)
```

Kôd 3. – Izbacivanje redaka u kojima nedostaju vrijednosti

4.3 Proces modeliranja

Nakon prikupljanja i pripremanja podataka, treća faza uključuje stvaranje inteligentnih modela strojnog učenja za podršku naprednoj analitici. Takvi modeli koriste različite algoritme. U ovom slučaju koristi se logistička regresija.

Prvo se dodaje konstantna vrijednost očišćenom skupu podataka. Točnije, dodaje se novi stupac s jedinicama u skup podataka.

	const	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
0	1.0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	77.0	0
1	1.0	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	76.0	0
2	1.0	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	70.0	0
3	1.0	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	103.0	1
4	1.0	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	85.0	0

Sl.4.2 Dodavanje konstante u tablicu podataka

Nakon toga želimo vidjeti rezultat od logističke regresije i parametre.

	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.6532	0.687	-12.589	0.000	-10.000	-7.306
sex_male	0.5742	0.107	5.345	0.000	0.364	0.785
age	0.0641	0.007	9.799	0.000	0.051	0.077
currentSmoker	0.0739	0.155	0.478	0.633	-0.229	0.377
cigsPerDay	0.0184	0.006	3.000	0.003	0.006	0.030
BPMeds	0.1448	0.232	0.623	0.533	-0.310	0.600
prevalentStroke	0.7193	0.489	1.471	0.141	-0.239	1.678
prevalentHyp	0.2142	0.136	1.571	0.116	-0.053	0.481
diabetes	0.0022	0.312	0.007	0.994	-0.610	0.614
totChol	0.0023	0.001	2.081	0.037	0.000	0.004
sysBP	0.0154	0.004	4.082	0.000	0.008	0.023
diaBP	-0.0040	0.006	-0.623	0.533	-0.016	0.009
BMI	0.0103	0.013	0.827	0.408	-0.014	0.035
heartRate	-0.0023	0.004	-0.549	0.583	-0.010	0.006
glucose	0.0076	0.002	3.409	0.001	0.003	0.012

Sl. 4.3 Rezultat logističke regresije

Možemo zaključiti po rezultatu da neki atributi imaju nizak statistički značajan odnos s vjerojatnošću CHD. To su atributi čija je P-vrijednost veća od preferirane alfe (koja iznosi 5%). Kako bi se to riješilo koristi se metoda eliminacije unatrag pomoću koje se uklanjaju oni atributi s najvišim P-vrijednostima, te nakon toga slijedi uzastopno izvođenje regresije dok svi atributi ne budu imali P-vrijednost manju od 0,05.

Zavisna varijabla i popis naziva stupaca pokreću značajku ponavljajuće eliminacije regresije s P-vrijednošću iznad alfe (5%) jednu po jednu, te vraća sažetak logističke regresije sa svim p-vrijednostima ispod alfe.

```
def back_feature_elem (data_frame, dep_var, col_list):
     while len(col_list)>0 :
            model = sm.Logit(dep_var,data_frame[col_list])
            result = model.fit(disp=0)
            largest_pvalue = round(result.pvalues,3).nlargest(1)
            if largest_pvalue[0]<(0.05):
                 return result
                 break
            else:
            col_list = col_list.drop(largest_pvalue.index)
result = back_feature_elem(df_constant, df.TenYearCHD, cols)
result.summary()
```

Kôd 4. – Funkcija metode eliminacije unatrag i sažetak rezultata

	coef	std err	z	P>|z|	[0.025	0.975]
const	-9.1264	0.468	-19.504	0.000	-10.043	-8.209
sex_male	0.5815	0.105	5.524	0.000	0.375	0.788
age	0.0655	0.006	10.343	0.000	0.053	0.078
cigsPerDay	0.0197	0.004	4.805	0.000	0.012	0.028
totChol	0.0023	0.001	2.106	0.035	0.000	0.004
sysBP	0.0174	0.002	8.162	0.000	0.013	0.022
glucose	0.0076	0.002	4.574	0.000	0.004	0.011

Sl. 4.4 Rezultat logističke regresije nakon metode eliminacije unatrag

Iz rezultata možemo vidjeti atribute kojima je p-vrijednost manja od alfe, a to su:

- **male**
- **age**
- **cigsPerDay**
- **totChol**
- **sysBP**
- **glucose**

Iz toga možemo izvući jednadžbu logističke regresije :

$$P = \frac{e^{\beta_0+\beta_1 x_1}}{1 + e^{\beta_0+\beta_1 x_1}}$$

logit(p) = $\log(\frac{p}{(1-p)})$

= $\beta_0 + \beta_1 * sex + \beta_2 * age + \beta_3 * cigsPerDay + \beta_4 * totChol + \beta_5 * sysBP + \beta_6 * glucose$

Nakon toga dobro je protumačiti omjer izgleda, intervala pouzdanosti i P-vrijednost.

```
params = np.exp(result.params)
conf = np.exp(result.conf_int())
conf['OR'] = params
pvalue = round(result.pvalues,3)
conf['pvalue'] = pvalue
conf. stupci = ['CI 95%(2,5%)','CI 95%(97,5%)', 'Odds Ratio', 'pvalue']

print((conf))
```

Kôd 5. Omjer izgleda, intervala pouzdanosti i P-vrijednosti

	0	1	OR	pvalue
const	0.000043	0.000272	0.000109	0.000
sex_male	1.455242	2.198536	1.788687	0.000
age	1.054483	1.080969	1.067644	0.000
cigsPerDay	1.011733	1.028128	1.019897	0.000
totChol	1.000158	1.004394	1.002273	0.035
sysBP	1.013292	1.021784	1.017529	0.000
glucose	1.004346	1.010898	1.007617	0.000

Sl. 4.5 Rezultat omjera izgleda,intervala pouzdanosti i P-vrijednosti

Sada možemo protumačiti izglede dijagnosticiranja CHD. Šansa da vam se dijagnosticira CHD za muškarce u odnosu na žene je 1,788687. Možemo reći da su izgledi za dijagnosticiranje muškaraca 78,9% veći od izgleda za žene. Koeficijent dobi gledajući parametar jednogodišnjeg povećanja starosti iznosi 1,067644 ili oko 7%.

Svaka dodatna popušena cigareta povećava izglede za dijagnosticiranje za 2% (1,019897). Za svaku jedinicu povećanja ukupne razine kolesterola, izgledi dijagnosticiranja se povećava za 0,2%. Za svaku jedinicu povećanja sistoličkog krvnog tlaka izgledi dijagnosticiranja se povećavaju za 1,7%. Za svaku jedinicu povećanja razine glukoze izgledi dijagnosticiranja CHD se povećava za 0,7%.

4.4 Evaluacija modela

Nakon stvaranja i treniranja modela strojnog učenje odnosno logističke regresije potrebno je provesti evaluaciju modela. Evaluacija modela ima za cilj procijeniti točnost generalizacije modela na budućim podacima.

Prilikom izvođenja klasifikacijskih predviđanja mogu se pojaviti četiri vrste ishoda:

- **Istinski pozitivni** – kada predvidite da opažanje pripada klasi i stvarno njoj pripada
- **Istinski negativni** – kada predvidite da opažanje ne pripada klasi i stvarno ne pripada
- **Lažno pozitivni**- kada predvidite da opažanje pripada klasi, a u stvari ne pripada
- **Lažno pozitivni** – kada predvidite da opažanje ne pripada klasi, a u stvari pripada

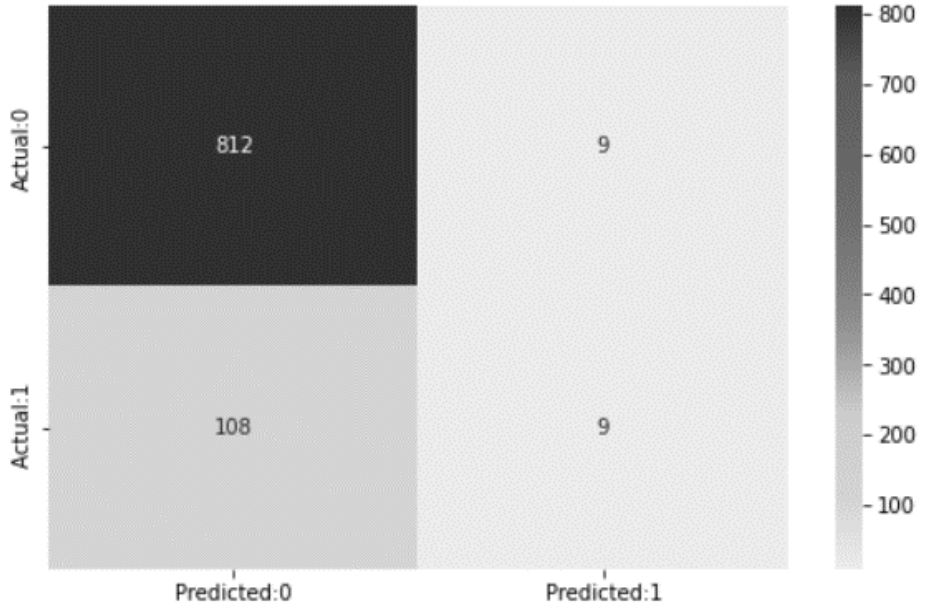
Navedeni ishodi se često iscrtavaju na matrici konfuzije. Ona detektira zbroj istinskih pozitivnih (IP) i negativnih (IN), te zbroj lažno pozitivnih (LP) i negativnih (LN) u predviđanjima klasifikatora. Iz nje možemo izvesti točnost koja je dana zbrojem ispravljenih predviđanja podijeljenim s ukupnim brojem predviđanja.

$$T = \frac{IP + IN}{IP + LP + IN + LN}$$

Također, postoji još nekoliko standarda mjera performansi:

- **Osjetljivost ili opoziv (engl. Sensitivity or Recall):** $\frac{IP}{IP+LN}$ ILI **True-Positive Rate, skraćeno TPR**
- **Specifičnost (engl. Specificity)**: $\frac{IN}{IN+LP}$ ILI **True-Negative-Rate, skraćeno TNR**
- **Preciznost**: $\frac{IP}{IP+LP}$ ILI **Positive Predicted Value**
- **False-Positive-Rate, skraćeno FPR:** $\frac{LP}{LP+IN}$
- **False-Negative-Rate, skraćeno FNR:** $\frac{LN}{LN+IP}$
- **F1 Score:** $2 * \frac{Specifičnost * Preciznost}{Specifičnost + Preciznost}$

Za dobar klasifikator potrebno je da su TPR i TNR što bliže 100%. Slično je i za parametre preciznosti i točnosti. Naprotiv FPR I FNR trebaju biti što bliže 0%.



Sl. 4.6 Matrica kofuzije logističke regresije

Možemo vidjeti da je istinskih pozitivnih 9, istinskih negativnih 812, lažno pozitivnih 9 i lažno negativnih 108. Također imamo 812+9=821 točnih predviđanja i 108+9=117 netočnih predviđanja. Nakon stvaranja matrice konfuzije, izračunavamo točnost, osjetljivost, specifičnost, preciznost, FPR, FNR.

```
The acuuracy of the model = TP+TN/(TP+TN+FP+FN) =  0.8752665245202559
 The Missclassification = 1-Accuracy =  0.12473347547974412
 Sensitivity or True Positive Rate = TP/(TP+FN) =  0.07692307692307693
 Specificity or True Negative Rate = TN/(TN+FP) =  0.9890377588306942
 Positive Predictive value = TP/(TP+FP) =  0.5
 Negative Predictive Value = TN/(TN+FN) =  0.8826086956521739
 Positive Likelihood Ratio = Sensitivity/(1-Specificity) =  7.017094017093985
 Negative Likelihood Ratio = (1-Sensitivity)/Specificity =  0.9333080712391059
```

Sl. 4.7 Rezultat standardnih mjera performansi

Za model logističke regresije, mjere performansi iznose:

- Točnost iznosi 87,52%
- Pogreška klasifikacije iznosi 12,47%
- Osjetljivost ili TPR iznosi 7,6%
- Specifičnost ili TNR iznosi 98,9%
- Preciznost iznosi 50%
- Negativno predviđena vrijednost iznosi 88,2%
- Omjer pozitivne vjerojatnosti iznosi 7,01
- Omjer negativne vjerojatnosti iznosi 0,93

Iz gornje statistike vidimo da je model više specifičan nego osjetljiv. Negativne vrijednosti predviđaju se točnije od pozitivnih. Iz matrice konfuzije možemo zaključiti ako je broj lažno negativnih rezultata vrlo velik, onda se može klasificirati kao umjereno opasan jer to znači ignoriranje vjerojatnosti bolesti kada zapravo postoji. Dakle kako bi se povećala osjetljivost, prag se može sniziti.

Logistička regresija nema ugrađenu metodu za podešavanje praga. Budući da znamo da je prema zadanim postavkama prag postavljen na 0,5 možemo upotrijebiti 0,25. To nam govori da će sve iznad 0,25 biti klasificirano kao 1.

```
import numpy as np
cm1=0
THRESHOLD1 = 0.25
y_pred1 = np.where(model.predict_proba(x_test)[:,1] >= THRESHOLD1, 1, 0
)
y_pred_proba_new_threshold1 = (model.predict_proba(x_test)[:,1] >= THRE
SHOLD1).astype(int)
cm1=confusion_matrix(y_test,y_pred_proba_new_threshold1)
print('With',THRESHOLD1,'threshold the Confusion Matrix is ','\n',cm1,'
\n',
            'with',cm1[0,0]+cm1[1,1],'correct predictions and',cm1[1,0]
,'Type II errors( False Negatives)','\n\n',
          'Sensitivity: ',cm1[1,1]/(float(cm1[1,1]+cm1[1,0])),'Specific
ity: ',cm1[0,0]/(float(cm1[0,0]+cm1[0,1])),'\n\n\n')
```

Kôd 6. – Postavljanje donjeg praga na 0.25

```
With 0.25 threshold the Confusion Matrix is
 [[698 123]
 [ 64  53]]
 with 751 correct predictions and 64 Type II errors( False Negatives)

 Sensitivity:  0.452991452991453 Specificity:  0.8501827040194885
```

Sl. 4.8 Rezultat mijenjanja donjeg praga na 0.25

Po rezultatu se vidi da se povećala osjetljivost na 45,29% dok je sa zadanim pragom logističke regresije osjetljivost 7,6%. Specifičnost se malo smanjila. Sada iznosi 85,01% dok je prije iznosila 98,9%. Još uvijek je model više specifičan nego osjetljiv.

4.5 Spremanje i pozivanje modela

Nakon provođenja modeliranja i evaluacije, još je potrebno spremiti model kako bi se mogao pozivati iz aplikacije. Za to smo iskoristili Python modul pickle. Pickle se koristi za implementaciju binarnih protokola za serijalizaciju i deserijalizaciju strukture Python objekta. „Salamurenje“ (engl. „Pickling“) čini proces u kojem se hijerarhija Python objekta pretvara u tok byte-ova dok „odvajanje“ (engl. „unpickling“) inverzna operacija odnosno tok byte-ova se pretvara natrag u hijerarhiju objekta. Ukratko, Pickle se koristi za pohranu datoteka/baza podataka, održavanje stanja programa kroz sesije ili za prijenos podataka preko mreže. U ovome slučaju, datoteka za pohranu se sastoji od modela logističke regresije.

```
new_data={"model":model}

with open('saved_steps.pkl','wb') as file:

pickle.dump(new_data,file)
```

Kôd 7. Spremanje modela

Model je sada spremljen u datoteci saved_steps.pkl. Kako bismo dohvatili model u aplikaciji, potrebno je napraviti poziv na funkciju koja će pročitati datoteku i vratiti model.

```
def load_model():
    with open('saved_steps.pkl','rb') as file:
        new_data = pickle.load(file)
    return new_data
new_data = load_model()
model = new_data["model"]
```

Kôd 8. Učitavanje modela unutar aplikacije

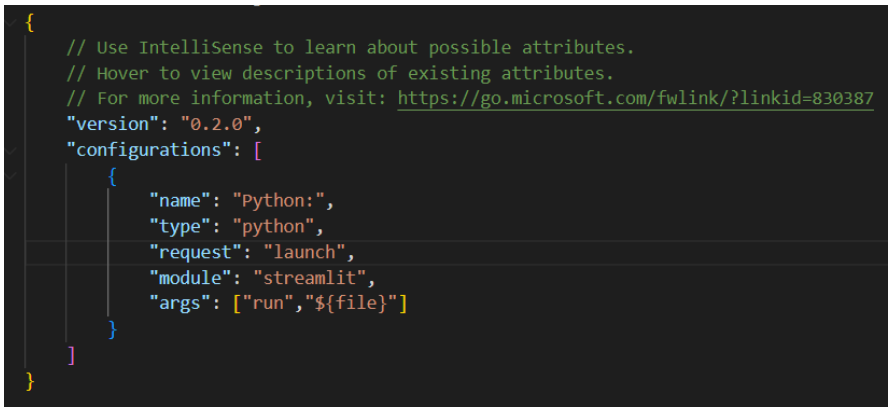
4.6 Konfiguracija Visual Studio Code-a

Naredbeni redak „conda“ se koristi za instalaciju Streamlit-a i ostalih biblioteka koje su potrebne za izradu aplikacije te za aktivaciju virtualnog okruženja. Kako Visual Studio Code (VSCode) u terminalu ne razumije naredbu „conda“, potrebno je konfigurirati postavke samo za trenutni radni direktorij.

Sl. 4.9 Konfiguracija "conda" u VSCode-u

Dovoljno je obrazložiti nekoliko postavki. Prva postavka „pythonPath“ određuje Python koji se izvršava, a to je onaj koji se nalazi u virtualnom okruženju „ml“. Druga postavka „activateEnvironment“ omogućuje aktivaciju „conda“ okruženja i mora biti postavljena na true. Postavka „terminal.integrated.profiles.windows“ nam prikazuje zadanu terminalsku konzolu, a to je cmd.exe. S ovim postavkama se omogućila aktivacija conda okruženja za VSCode terminal. Sada je dovoljno upisati naredbu „conda activate ml“ kako bismo aktivirali virtualno okruženje. [21]

VSCode ne razumije ni naredbu „streamlit“. Naredba u kojoj se to koristi je „streamlit run app.py“. Ona služi za pokretanje aplikacije na local host-u. Potrebno je zato naučiti VSCode kako pokrenuti aplikaciju tako što ćemo izraditi launch.json datoteku. Kada uđemo u Run and Debug, imamo opciju kreiranje launch.json i potrebno je samo odabrati modul streamlit i sve se automatski kreira. Jedino što je potrebno dodati je „args“ odnosno argumente koji se nalaze uz naredbu „streamlit“. [22]



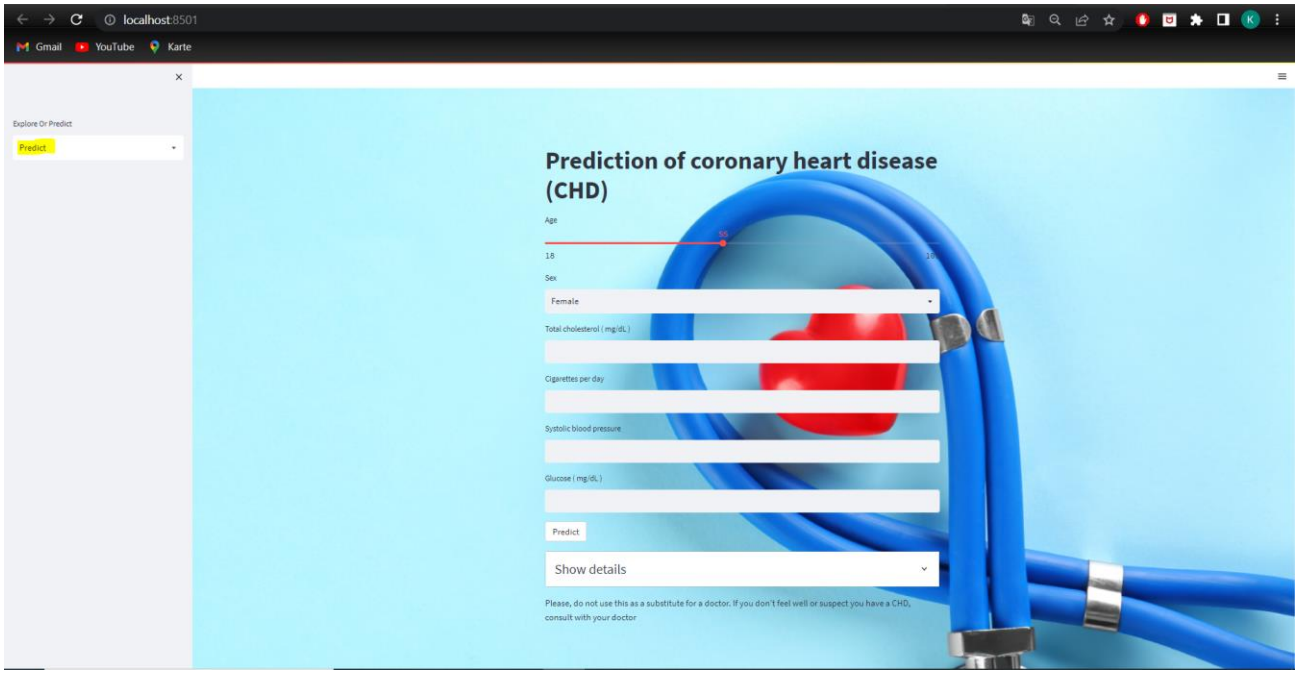
Sl. 4.10 Konfiguracija pokretanja aplikacije s naredbom "streamlit run app.py"

Kada smo konfigurirali, imamo mogućnost pokretanja aplikacije kroz VSCode. Prije toga aplikacija se trebala pokretati iz Anaconda Navigator terminala.

4.7 Izrada web stranica

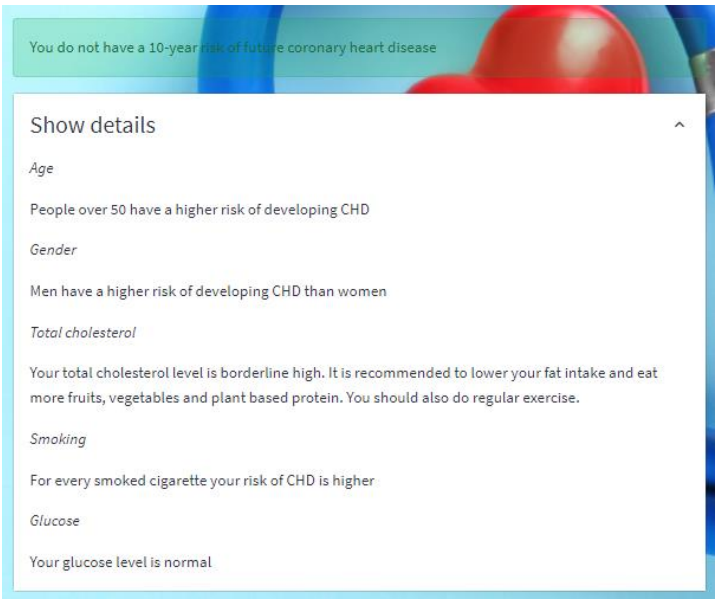
Za korištenje Streamlit-a nije potrebno front-end iskustvo. Različiti widgeti se deklariraju kao varijable. Nema potrebe za pisanjem backend-a, definiranjem ruta, pisanjem HTML-a, CSS-a i JavaScripta. To čini Streamlit jednostavnim i popularnim među konkurentima. Sve naredbe su pojednostavljenje, potrebno je samo napraviti uvoz biblioteke „streamlit“ i pogledati dokumentaciju. Sve naredbe koje koristimo iz te biblioteke započinju s nazivom uvoza, u ovome slučaju je to „st“.

Aplikacija je podijeljena u 3 web stranice. Glavna stranica „app.py“ služi za odabir koju stranicu želimo prikazati. Potrebno je deklarirati varijablu koja predstavlja bočnu traku na aplikaciji. U tu varijablu je ugrađen i selectbox pomoću kojeg se određuje koja će se stranica prikazati.



Sl. 4.11 Prikaz stranice za predviđanje

Zatim, imamo stranicu „predict-page.py“ koja daje korisniku mogućnost unosa parametara pomoću kojih se predviđa njegov 10-godišnji rizik od CHD. Cijeli izgled te stranice je napisan unutar funkcije „show_predict_page()“ koja se poziva kada select box ima vrijednost „Predict“. Također se u njoj nalazi funkcija za učitavanje modela (Kôd 8), te funkcija za dodavanje pozadine aplikaciji (koja se poziva na glavnoj stranici). Nakon što model predvidi rizik, korisnik može ući u „Show details“ gdje će vidjeti pojašnjenje unosa (ovisno o parametrima).



Sl. 4.12 Primjer pojašnjenja unosa

Treća stranica je „visualisation_page.py“ koja nam služi za bolje razumijevanje skupa podataka. Na njoj su vizualizirani grafikoni pomoću kojih se mogu vidjeti odnosi u skupu podataka. Za vizualizaciju je korištena datoteka „visualisation.csv“ u kojoj su pojedini atributi podijeljeni u više grupa. Atribut *totChol* podijeljen je u tri grupe [23]:

- ***Normal*** – kada je razina ukupnog kolesterola manja od 200 mg/dL
- ***Borderline high*** – kada je razina ukupnog kolesterola između 200 i 239 mg/dL
- ***High*** – kada je razina ukupnog kolesterola veća od 240 mg/dL

Najviše ispitanika spada u grupu „High“ zatim slijedi „Borderline high“ te najmanje ispitanika pripada grupi „Normal“.

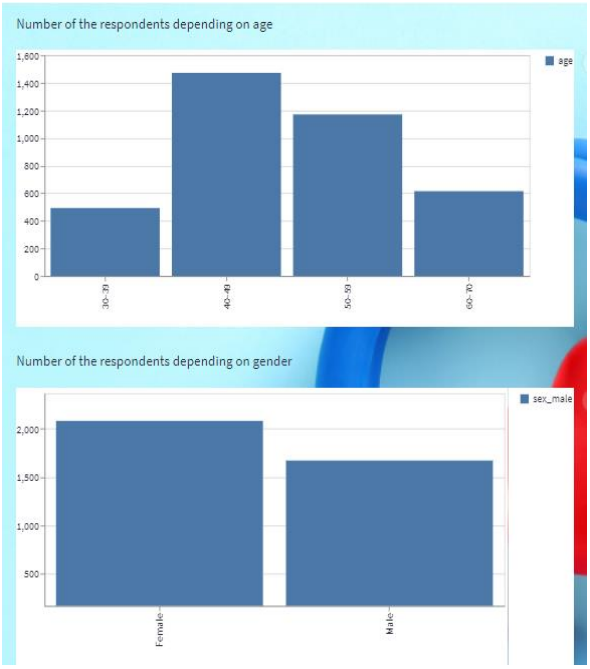
Pošto za razinu glukoze u Framingham-ovoj studiji nije navedeno kako su vrijednosti testirane, prema podacima se može zaključiti da se radi o testu šećera u krvi natašte. Zato je atribut *glucose* podijeljen u tri grupe [24]:

- ***Normal***- kada je razina glukoze manja od 99 mg/dL
- ***Prediabetes***- kada je razina glukoze između 100 i 125 mg/dL
- ***Diabetes***- kada je razina veća od 126 mg/dL

Velika većina ispitanika ima normalnu razinu glukoze, dok ispitanici s dijabetesom i predijabetesom čine samo 9% od ispitanih.

Atribut *age* je podijeljen u četiri grupe: 30-39, 40-49, 50-59, 60-70. Najviše ispitanika ima između 40-49 godina, nakon toga slijedi 50-59, 60-70. Najmanje ispitanika spada u grupu 30-39.

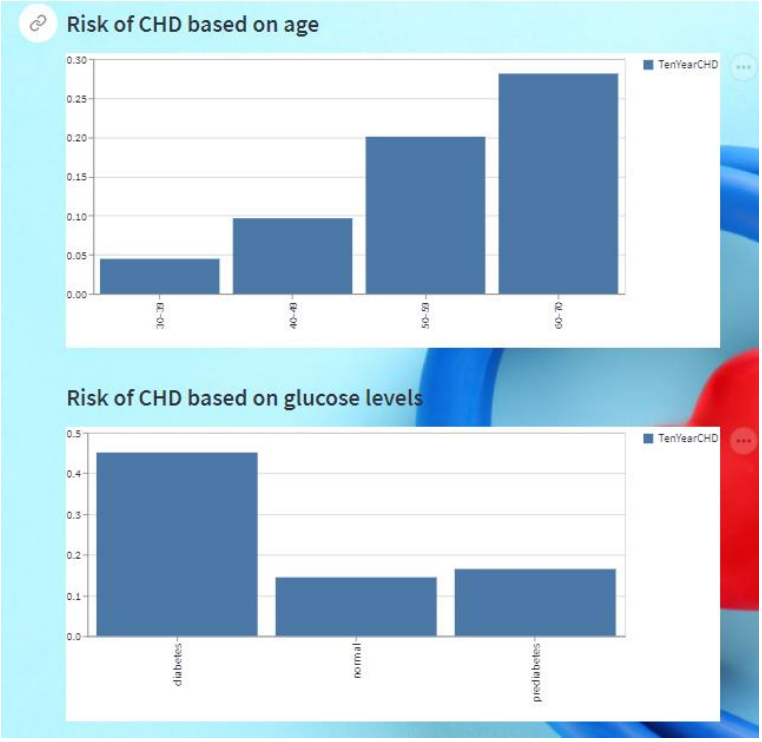
Atribut *sex_male* je podijeljen u dvije grupe: „Female“ i „Male“. Među ispitanicima je malo više žena nego muškaraca.



Sl. 4.13 Grafikon omjera ispitanika po godinama i spolu

Sl. 4.14 Grafikon omjera ispitanika po razini glukoze i kolesterola

Uz razumijevanje skupa podataka, vizualizirana je i ovisnost 10-godišnjeg rizika o atributima.



Sl. 4.15 Rizik ovisno o godinama(gornji graf) i razini glukoze (donji graf)

S grafova se vidi da što je ispitanik stariji, ima veći rizik od dijagnosticiranja CHD. Također se može vidjeti da najveći rizik imaju osobe s dijabetesom (iako ih je u ukupnom broju ispitanika samo 82 s dijabetesom). Zatim slijede osobe s predijabetesom, te na kraju osobe s normalnom razinom glukoze.

Sl. 4.16 Rizik ovisno o kolesterolu (gornji graf) i spolu (donji graf)

S ovih grafova se može vidjeti kako najveći rizik imaju osobe s visokom razinom kolesterola. Zatim slijede osobe sa graničnim kolesterolom i na kraju su osobe s normalnom razinom kolesterola. Po spolu se vidi da muškarci imaju dosta veći rizik od rizika nego žene.

Zaključak

Umjetna inteligencija se već danas koristi u medicini. Daljnjim razvojem metoda umjetne inteligencije doći će do napretka u istraživanjima i pomaganju cjelokupnom liječenju pacijenata. Skup podataka se temelji na različitim atributima i bilo ih je potrebno pripremiti za modeliranje. Korišten je model logističke regresije kako bi predvidjeli 10-godišnji rizik od CHD. On je samo jedan od načina na koji smo mogli pristupiti skupu podataka. Točnost modela iznosi 87,5% što je zadovoljavajuće. Iako je kroz usporedbu s ostalim algoritmima logistička regresija dala najbolji rezultat, njeno predviđanje nije savršeno. Svi atributi koji su odabrani za izradu modela imaju P-vrijednost nižu od 5% što znači da imaju značajnu ulogu u predviđanju CHD. Nakon procesa modeliranja može se zaključiti da muškarci imaju veći rizik dijagnosticiranja CHD nego žene. Što su veće razine kolesterola i glukoze u krvi, to su i veći izgledi za razvoj CHD. Starija životna dob ima veliki utjecaj na oboljenje od CHD. Povećanjem broja popušenih cigareta dnevno i sistoličkog krvnog tlaka također se povećavaju izgledi za CHD. U matrici konfuzije se moglo vidjeti kako je broj lažno negativnih rezultata vrlo velik. Iz toga se može zaključiti da je model više specifičan nego osjetljiv. Model se može smatrati umjereno opasnim jer ignoriranjem tih vjerojatnosti, ignoriramo postojanje te bolesti. Povećanjem skupa podataka cjelokupni model bi se mogao poboljšati.

Literatura

[1] V. Carević, M. Rumboldt i Z. Rumboldt, »Koronarni čimbenici rizika u hrvatskoj i u svijetu: rezultati Interheart studije,« *Acta Medica Croatica,* svez. 61, br. 1330-0164, pp. 299-306, Lipanj 2007.

[2] Društvo, »Što je umjetna inteligencija i kako se upotrebljava?,« Europski parlament, 4 Rujan 2020. [Mrežno]. Available: https://www.europarl.europa.eu/news/hr/headlines/society/20200827STO85804/sto-je-umjetna-inteligencija-i-kako-se-upotrebljava.

[3] D. Ankers, »Types of Artificial Intelligence: A Detailed Guide,« 2018. [Mrežno]. Available: https://certes.co.uk/types-of-artificial-intelligence-a-detailed-guide/.

[4] B. Marr, »Understanding the 4 Types of Artificial intelligence,« Bernard Marr & Co., 2021. [Mrežno]. Available: https://bernardmarr.com/understanding-the-4-types-of-artificial-intelligence/.

[5] A. Biswal, »7 Types of Artificial Intelligence That You Should Know in 2022,« 21 Lipanj 2022. [Mrežno]. Available: https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/types-of-artificial-intelligence. [Pokušaj pristupa 03 Kolovoz 2022].

[6] D. d. s. A. Jović, »Uvod u strojno učenje,« 23 Listopad 2019. [Mrežno]. Available: https://www.bib.irb.hr. [Pokušaj pristupa 16 Kolovoz 2022].

[7] N. Bolf, »Strojno učenje,« u *Osvježimo znanje*, Zagreb, Hrčak, 2021, pp. 591-593.

[8] J. Brownlee, »4 Types of Classification Tasks in Machine Learning,« 8 Travanj 2020. [Mrežno]. Available: https://machinelearningmastery.com/types-of-classification-in-machine-learning/. [Pokušaj pristupa 16 Kolovoz 2022].

[9] »Machine Learning - Logistic Regression,« Tutorials Point, [Mrežno]. Available: https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_logistic_regression.htm#. [Pokušaj pristupa 09 Kolovoz 2022].

[10] »Python (programming language),« [Mrežno]. Available: https://en.wikipedia.org/wiki/Python_(programming_language). [Pokušaj pristupa 15 Kolovoz 2022].

[11] »Why Python for Machine Learning?,« Python basics, [Mrežno]. Available: https://pythonbasics.org/. [Pokušaj pristupa 09 Kolovoz 2022].

[12] A. R. Rout, »Difference Between Jupyter and Pycharm,« 11 Listopad 2021. [Mrežno]. Available: https://www.geeksforgeeks.org/difference-between-jupyter-and-pycharm/. [Pokušaj pristupa 09 Kolovoz 2022].

[13] A. Sehmni, "Introduction to Streamlit and Streamlit Components," 20 Travanj 2022. [Online]. Available: https://auth0.com/blog/introduction-to-streamlit-and-streamlit-components/. [Accessed 06 Kolovoz 2022].

[14] »Introduction to Streamlit,« LatentView Analytics, [Mrežno]. Available: https://www.latentview.com/data-engineering-lp/introduction-to-streamlit/. [Pokušaj pristupa 21 Srpanj 2022].

[15] M. Schmitt, »Streamlit vs. Dash vs. Shiny vs. Voila vs. Flask vs. Jupyter,« Data Revenue, [Mrežno]. Available: https://www.datarevenue.com/en-blog/data-dashboarding-streamlit-vs-dash-vs-shiny-vs-voila. [Pokušaj pristupa 06 Kolovoz 2022].

[16] M. Sarmiento, »A Guide to Python's Virtual Environments,« 24 May 2019. [Mrežno]. Available: https://towardsdatascience.com/virtual-environments-104c62d48c54. [Pokušaj pristupa 03 Kolovoz 2022].

[17] »Anaconda Navigator,« Anaconda Inc., [Mrežno]. Available: https://docs.anaconda.com/anaconda/navigator/. [Pokušaj pristupa 09 Kolovoz 2022].

[18] N. K. Nissa, »How to Predict Coronary Heart Disease Risk using Logistic Regression?,« 17 Travanj 2021. [Mrežno]. Available: https://medium.com/analytics-vidhya/how-to-predict-coronary-heart-disease-risk-using-logistic-regression-c069ab95cbec. [Pokušaj pristupa 20 Srpanj 2022].

[19] C. Snijders, U. Matzat and U. D. Reips, "'"Big Data": Big Gaps of Knowledge," *International Journal of Internet Science,* vol. 7, no. 1, pp. 1-5, 21 Prosinac 2012.

[20] T. R. Dawber, F. E. Moore and G. V. Mann, "II. Coronary Heart Disease in the Framingham Study," *International Journal of Epidemiolog,* vol. 44, no. 6, p. 1767–1780, 2015.

[21] Kathryn, »Efficient Way to Activate Conda in VSCode,« 27 Svibanj 2020. [Mrežno]. Available: https://medium.com/analytics-vidhya/efficient-way-to-activate-conda-in-vscode-ef21c4c231f2. [Pokušaj pristupa 26 Srpanj 2022].

[22] A. Jones, »How to Run Your Streamlit Apps in VSCode,« 23 Studeni 2021. [Mrežno]. Available: https://medium.com/geekculture/how-to-run-your-streamlit-apps-in-vscode-3417da669fc. [Pokušaj pristupa 26 Srpanj 2022].

[23] »Lipid Panel,« The Johns Hopkins University, The Johns Hopkins Hospital, and Johns Hopkins Health System, [Mrežno]. Available: https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/lipid-panel. [Pokušaj pristupa 23 Kolovoz 2022].

[24] »Diabetes Tests,« Centers for Disease Control and Prevention, [Mrežno]. Available: https://www.cdc.gov/diabetes/basics/getting-tested.html. [Pokušaj pristupa 23 Kolovoz 2022].

Pregled slika