



**Univerzitet u Nišu  
Elektronski fakultet  
Katedra za računarstvo**



## **Seminarski rad**

# **Sistem za predikciju različitih bolesti na osnovu zadatih simptoma**

**Mentor:**  
**Doc. dr Miloš Bogdanović**

**Student:**  
**Andrea Popović 1475**

**Niš, 2023**

## Sadržaj

1. Uvod .....	3
2. Predviđanje u Web miningu.....	4
3. Podaci.....	6
3.1. Opis skupa podataka.....	6
3.1.1. Prvi skup podataka- <i>Stroke detection</i> .....	7
3.1.2. Drugi skup podataka- <i>Heart attack detection</i> .....	8
3.1.3. Treci skup podataka .....	10
4. Implementacija sistema za predikciju.....	11
4.1. Iskorišćene biblioteke .....	11
4.2. Preprocesiranje podataka .....	12
4.3. Iskorišćeni algoritmi .....	14
4.3.1. KNN algoritam .....	14
4.3.2. Decision Tree algoritam .....	17
4.3.3. Random Forest algoritam .....	19
5. Provera rezultata i testiranje tačnosti .....	21
6. Izgled aplikacije .....	26
7. Zaključak.....	28
8. Literatura.....	29

## 1. Uvod

Web mining predstavlja proces otkrivanja šablona u velikim skupovima podataka dobijenih ekstrakcijom iz web dokumenata, web sadržaja, stranica, hiperveza i sl. , korišćenjem metoda mašinskog učenja, statistike ili sistema baza podataka. Opšti cilj ove oblasti predstavlja izvlačenje informacija (korišćenjem inteligentnih metoda) iz skupova podataka i transformacija tih informacija u razumljivu strukturu radi dalje upotrebe, tako da ona danas igra značajnu ulogu za rukovanje ogromnom količinom podataka.

Poslednjih godina, web mining je privukao ogromnu pažnju u oblasti zdravstvene zaštite kao sredstvo za predviđanje i prevenciju bolesti. Analizom velikih količina onlajn podataka, može se pomoći u identifikaciji ranih znakova upozorenja i faktora rizika za različita zdravstvena stanja. U ovom radu ćemo istražiti primenu web mining-a u predviđanju više različitih bolesti na osnovu zadatih simptoma i informacija, a sve to zasnovano na obučanim modelima i korišćenjem podataka sa weba.

Jedna od ključnih prednosti web mining-a u predviđanju bolesti je njegova sposobnost prikupljanja podataka iz različitih izvora, uključujući platforme društvenih medija, medicinske časopise i novinske članke, kao i javno dostupne podatke objavljene od strane velikih medicinskih ustanova. Analizom ovog raznolikog skupa podataka, algoritmi za web mining mogu identifikovati korelacije i obrasce koji ljudima možda nisu odmah očigledni. Na primer, praćenjem učestalosti određenih ključnih reči u objava na društvenim mrežama i upitima za pretragu na mreži, algoritmi za web mining mogu otkriti pojavu nove epidemije i bolesti pre nego što ona postane široko rasprostranjena. Ovaj sistem ranog upozoravanja može pomoći službenicima javnog zdravlja da preduzmu proaktivne mere za obuzdavanje epidemije i sprečavanje njenog širenja.

Web mining se takođe može koristiti za predviđanje pojave hroničnih bolesti, kao što su dijabetes, bolesti srca i rak. Analizom podataka iz elektronskih zdravstvenih kartona, medicinskih baza podataka i kliničkih ispitivanja, algoritmi za web mining mogu identifikovati faktore rizika i predvideti verovatnoću da pacijent razvije određenu bolest. Na primer, može se analizirati pacijentova medicinska istorija, izbor načina života i genetska predispozicija kako bi se utvrdio rizik od razvoja određene vrste bolesti. Ovaj princip je jako bitan jer ako se reaguje na vreme, može se sprečiti nastanak moguće štete i zdravstvenih problema. Uz ove informacije, zdravstveni radnici mogu dati personalizovane preporuke za prevenciju ili upravljanje bolešću. Upravo ovaj princip i algoritmi su iskorišćeni u ovom radu za predviđanje mogućnosti da pacijent razvije dijabetes i dobije moždani ili srčani udar.

## 2. Predviđanje u Web miningu

Predviđanje je važan aspect web mining-a, koji uključuje korišćenje tehnika rudarenja podataka i mašinskog učenja za analizu web podataka i predviđanja budućih događaja ili ishoda. U web mining-u, ova tehnika se koristi za predviđanje ponašanja kupaca, predviđanje trendova prodaje, predviđanje cena nekretnina, otkrivanje prevare i još mnogo toga. U ovom radu ova tehnika se koristi za predviđanje bolesti pacijenata. Postoji nekoliko tehnika predviđanja koje se mogu koristiti u web mining-u, svaka sa svojim prednostima i slabostima. U nastavku su izdvojene neke od njih koje se koriste često.

Prva tehnika predviđanja je regresiona analiza. Ova tehnika uključuje korišćenje statističkih metoda za analizu odnosa između dve ili više varijabli. Zavisna varijabla u rudarenju na webu može biti ponašanje korisnika, kao što su klikovi ili kupovine, ili uspeh marketinške kampanje, kao što je broj prikaza ili stope učestalosti klikova. Regresiona analiza se obično koristi za predviđanje prodajnih trendova ili predviđanje ponašanja kupaca. Na primer, prodavac može da koristi regresionu analizu da bi predvideo prodaju na osnovu faktora kao što su doba godine, vreme i promocije.

Druga tehnika predviđanja je analiza vremenskih serija, koja uključuje analizu podataka koji se prikupljaju tokom vremena. U ovoj vrsti analize beleže se tačke podataka u konzistentnim intervalima tokom određenog vremenskog perioda, umesto da se samo beleže tačke podataka povremeno ili nasumično. Ono što izdvaja podatke vremenske serije od ostalih podataka jeste to što analiza može pokazati kako se oni menjaju tokom vremena. Drugim rečima, vreme je ključna varijabla. Pruža dodatni izvor informacija i postavljeni redosled zavisnosti između podataka. Analiza vremenskih serija se obično koristi za predviđanje budućih trendova. Na primer, finansijska institucija može koristiti analizu vremenskih serija za predviđanje cena akcija na osnovu istorijskih podataka, predviđanje cena nekretnina ili neke robe. Takodje i u trgovini ima veliku primenu za procenu potrebne robe i analizu prodaje kao i sezonskih trendova.

Treća tehnika predviđanja su stabla odlučivanja, koja uključuju kreiranje vizuelnog prikaza procesa donošenja odluka. Stabla odlučivanja se pored predviđanja obično koriste u web mining-u za predviđanje ponašanja kupaca i davanje preporuka. *Recommendent* sistemi, odnosno sistemi za preporuku su jako bitna grana i oblast web mining-a. Primene ovog algoritma su raznovrsne. Na primer, web lokacija za e-trgovinu može da koristi stablo odlučivanja da preporuči proizvode kupcima na osnovu njihovih prošlih kupovina i ponašanja prilikom pregledanja. Takodje preporuke filmova ili knjiga na

osnovu prethodno odgledanih filmova ili procitanih knjiga. Ova tehnika je takodje iskorišćena i biće detaljno opisana u nastavku.

Četvrta tehnika predviđanja su veštačke neuronske mreže, koje uključuju kreiranje modela koji oponašaju ponašanje ljudskog mozga. One odražavaju ponašanje ljudskog mozga, omogućavajući kompjuterskim programima da prepoznaju obrasce i rešavaju uobičajene probleme u oblastima veštačke inteligencije (AI), mašinskog učenja i dubokog učenja, kompijuterskog vida, data i web mininga. Njihova primena je raznovrsna. Veštačke neuronske mreže se pored predikcije obično koriste u web minigu za predviđanje ponašanja kupaca, kao što je verovatnoća da će kupac obaviti kupovinu. Na primer, onlajn prodavac može da koristi veštačku neuronsku mrežu da predvidi koje će proizvode kupac verovatno kupiti na osnovu njihovog ponašanja u prošlosti. U finansijama, postoje aplikacije neuronske mreže za otkrivanje prevara, upravljanje i predviđanje i pomažu bankama da pronađu rešenja prilikom analize rizika i verovatne dobiti, kao što je da li da odobre kredit. Takodje u medicini se koristi klasifikacija pomoću veštačkih neuronskih mreža za odredjivanje da li pacijent ima rak ili ne na osnovu rendgenskih snimaka. Veštačke neuronske mreže su postale pouzdan i koristan alat za pronalaženje rešenja unutar nestrukturiranih podataka, zahvaljujući svojoj sposobnosti da daju smisao nelinearnim procesima. Različiti problemi se sada rešavaju, što olakšava donosiocima odluka da saznaju pravi put napred i preduzimaju sigurnije i održivije korake ka budućnosti svog poslovanja.

Peta tehnika predviđanja su mašine za podršku vektorima, koje uključuju kreiranje modela koji razdvajaju podatke u različite klase. Koriste se za obuku modela za učenje klasifikacije i pravila regresije. Mašine sa vektorima podrške se obično koriste u web mining-u za zadatke klasifikacije i predviđanja, kao što je otkrivanje prevare ili predviđanje odliva kupaca. Na primer, finansijska institucija može da koristi mašinu za vektor podrške da otkrije prevaru sa kreditnim karticama analizom prošlih transakcija i identifikacijom obrazaca povezanih sa prevarama. Takodje i se koriste često za otkrivanje lažnih vesti. Može se iskoristiti i za predikciju ili detekciju bolesti.

Šesta izdvojena tehnika predviđanja je Random Forest. To je popularan algoritam mašinskog učenja koji pripada tehnici učenja pod nadzorom. Može se koristiti za probleme klasifikacije i regresije u ML. Zasnovan je na konceptu učenja ansambla, što je proces kombinovanja više klasifikatora za rešavanje složenog problema i poboljšanje performansi modela. Kao što ime sugerise, Random Forest je klasifikator koji sadrži niz stabala odlučivanja o različitim podskupovima datog skupa podataka i uzima prosek da bi poboljšao tačnost predviđanja tog skupa podataka. Ova tehnika je iskorišćena za predikciju u implementaciji tako da će o njoj biti više reči u nastavku.

### 3. Podaci

Praktični deo rada sastoji se od predikcije bolesti na osnovu simptoma, odnosno mogućnosti da pacijent dobije dijabetes, srčani ili moždani udar. Korisnik unosi podatke koji se od njega traže zatim nakon obrade tih podataka, algoritam obaveštava da li postoji mogućnost za nastanak neke od tih bolesti ili ne. Aplikacija je razvijena u programskom jeziku *python* i korišćene su dodatne biblioteke o kojima će kasnije biti reč. Cilj aplikacije jeste da pokaže kako kombinovanjem algoritama mašinskog učenja sa medicinskim podacima, web rudarenje može pomoći zdravstvenim radnicima da donesu bolje informisane odluke o nezi pacijenata i razviju personalizovane planove lečenja koji su prilagođeni individualnim potrebama pacijenata. U nastavku će biti opisani podaci i algoritmi koji su korišćeni, kao i detaljan izgled same aplikacije.

#### 3.1. Opis skupa podataka

S obzirom na to da je čitav koncept ovog rada zasnovan na mašinskom učenju potrebno je pre svega pomenuti sastavni deo mašinskog učenja bez koga ono ne bi imalo smisla jer se ne bi proizvodili valjani rezultati, a to podaci. Skup podataka je kritičan faktor u mašinskom učenju, jer pruža osnovu na kojoj se gradi model. Visokokvalitetni skup podataka koji je reprezentativan, raznovrstan i dovoljne veličine može pomoći u stvaranju tačnog i robusnog modela mašinskog učenja, što može dovesti do efikasnijih predviđanja i boljih rezultata. Jedan od glavnih razloga zašto je visokokvalitetni skup podataka neophodan u mašinskom učenju je taj što modelu pruža informacije koje su mu potrebne da nauči i pravi tačna predviđanja. Veliki i raznovrstan skup podataka može pomoći modelu da identifikuje obrasce i odnose između tačaka podataka, čineći ga bolje opremljenim za rukovanje scenarijima iz stvarnog sveta.

Kao što je već napomenuto implementirana aplikacija se koristi za predikciju 3 različitih bolesti, tako da je korišćeno 3 različita skupa podataka. Svi ovi podaci koriste se isključivo za treniranje modela kako bi oni kasnije sa velikim procentom tačnosti izvršili predikciju na osnovu unetih parametara od strane korisnika. Podaci su preuzeti sa sledećih linkova:

- <https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset>,
- <https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset>
- <https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey?resource=download>

### 3.1.1. Prvi skup podataka- *Stroke detection*

Prvi skup podataka se koristi za predviđanje da li će pacijent verovatno dobiti moždani udar na osnovu ulaznih parametara kao što su pol, starost, razne bolesti i pušački status i sl. Svaki red u podacima pruža relevantne informacije o pacijentu. Podaci o 5111 različitih pacijenata nalaze se u dokumentu *healthcare-dataset-stroke-data.csv*. Tabela podataka prikazana je ispod.

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0000	0	1	Yes	Private	Urban	228.6900	36.6000	formerly smoked	1
1	51676	Female	61.0000	0	0	Yes	Self-employed	Rural	202.2100	<NA>	never smoked	1
2	31112	Male	80.0000	0	1	Yes	Private	Rural	105.9200	32.5000	never smoked	1
3	60182	Female	49.0000	0	0	Yes	Private	Urban	171.2300	34.4000	smokes	1
4	1665	Female	79.0000	1	0	Yes	Self-employed	Rural	174.1200	24.0000	never smoked	1

Slika 1. Korišćeni skup podataka

Atributi podataka su:

- 1) **id**: jedinstveni identifikator,
- 2) **gender**: pol osobe  
"Male"-muskarac, "Female"-žena ili "Other"-ostalo,
- 3) **age**: godine osobe,
- 4) **hypertension**: 0 ako pacijent nema hipertenziju, odnosno 1 ako ima,
- 5) **heart\_disease**: 0 ako pacijent nema srčanih problema, odnosno 1 ako ima,
- 6) **ever\_married**: da li je pacijent ikada bio u braku "No"-ne ili "Yes"-da,
- 7) **work\_type**: Tip posla-"children"-deca, "Govt\_job"-državni posao, "Never\_worked"-osoba nikad nije radila, "Private"- privatni posao ili "Self-employed"-samozaposleni,
- 8) **Residence\_type**: tip sredine u kojoj pacijent živi: "Rural"-ruralna ili "Urban"-urbana,
- 9) **avg\_glucose\_level**: prosečni nivo glukoze u krvi,
- 10) **bmi**: indeks telesne težine,
- 11) **smoking\_status**: da li je osoba pušač: "formerly smoked"-bivši pušač, "never smoked"- osoba nikada nije pušila, "smokes"-pušač or "Unknown"-osoba ne želi da se izjasni,
- 12) **stroke**: 1 pacijent je imao moždani udar, odnosno 0 ako nije.

Radi lakse obuke modela i obrade podataka kolona koja predstavlja *id* osobe je izbačena jer nema nikakvu ulogu u predikciji, a vrednosti ostalih atributa su predstavljene numerički. Takodje izbačene su i *Null*-te vrednosti svih podataka. Izgled skupa podataka nakon izmene je prikazan na sledećoj slici.

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	0.0000	67.0000	0	1	1.0000	0.0000	0.0000	228.6900	36.6000	2.0000	1
1	1.0000	61.0000	0	0	1.0000	1.0000	1.0000	202.2100	34.5500	3.0000	1
2	0.0000	80.0000	0	1	1.0000	0.0000	1.0000	105.9200	32.5000	3.0000	1
3	1.0000	49.0000	0	0	1.0000	0.0000	0.0000	171.2300	34.4000	1.0000	1
4	1.0000	79.0000	1	0	1.0000	1.0000	1.0000	174.1200	24.0000	3.0000	1

Slika 2. Izmenjeni skup podataka

### 3.1.2. Drugi skup podataka- *Heart attack detection*

Drugi skup podataka se koristi za predviđanje da li pacijent ima velike šanse da dobije srčani udar na osnovu ulaznih parametara koje se od njega zahtevaju da unese, a koji se podudaraju sa podacima iz pomenutog skupa. Podaci o 304 različitih pacijenata nalaze se u dokumentu *heart.csv* i prikazani su na slici ispod. Broj pacijena nije znatno veliki jer se radi o privatnim i poverljivim podacima za koje je potrebna njihova lična saglasnost kako bi se objavili i koristili u svrhe istraživanja.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3000	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5000	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4000	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8000	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6000	2	0	2	1

Slika 3. Skup podataka



Atributi podataka koji se koriste su:

1. **Age:** godine pacijenta.
2. **Sex:** pol pacijenta :  
1 = muškarac  
0 = žena
3. **Chest-pain type:** opisuje tip bola u grudima koji se javljaju kod pacijenta  
1 = tipičana angina  
2 = atipičana angina  
3 = pacijent nema anginu  
4 = angina bez simptoma
4. **Resting Blood Pressure:** krvni pritisak pacijenta izražen u *mmHg*
5. **Serum Cholesterol:** serumski holesterol izražen u *mg/dl*
6. **Fasting Blood Sugar:** nivo šećera u krvi  
1 = nivo šećera je veći od 120mg/dl  
0 = nivo šećera je manji od 120mg/dl
7. **Resting ECG :** rezultati EKG zapisa  
0 = normalni  
1 = ima abnormalne vrednost ST-T talasa  
2 = hipertrofija leve komore
8. **Max heart rate achieved:** maksimalni broj otkucaja srca individue.
9. **Exercise induced angina :** da li pacijent ima angina izazvanu vežbanjem  
1 = da  
0 = ne
10. **ST depression induced by exercise relative to rest:** predstavlja vrednost ST depresije u toku vežbanja, odnosno pod određenim opterećenjem. To je najčešće hodanje na traci pod određenim nagibom i brzinom.
11. **Peak exercise ST segment :** vrhunac opterećenja(nagiba) koji je pacijent izdržao  
1 = naviše  
2 = ravno  
3 = naniže
12. **Number of major vessels (0–3) colored by fluoroscopy:** broj velikih krvnih sudova obojenih fluroskopijom.
13. **Thal:** talasemija  
3 = normalna  
6 = fiksni defekt  
7 = reverzibilni defect

Ovi podaci su jako bitni i nijedan ne sme biti isključen, što je i naučno dokazano. Međutim treba istaći da normalan EKG ne isključuje infarkt. Takođe je pokazano da 40-50% bolesnika sa akutnim bolom u grudima koji razvijaju akutni infarkt miokarda. Ima i nalaza koji sugerišu da se to ređe dešava kod muškaraca, nego kod žena, zbog toga je bitno uneti i pol osobe.

### 3.1.3. Treci skup podataka

Poslednji iskorišćeni skup podataka se koristi za predviđanje da li će pacijent dobiti ili već ima dijabetes. U dokumentu `heart.csv` nalaze se podaci o 769 različitih pacijenata i prikazani su na slici ispod. Kao i u prethodnim slučajevima postoji kolona koja sadrži informacije o tome da li pacijent ima veliku mogućnost da razvije bolest ili ne, ali se ti podaci kao što je već rečeno koriste isključivo za obuku i treniranje modela.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6000	0.6270	50	1
1	1	85	66	29	0	26.6000	0.3510	31	0
2	8	183	64	0	0	23.3000	0.6720	32	1
3	1	89	66	23	94	28.1000	0.1670	21	0
4	0	137	40	35	168	43.1000	2.2880	33	1

Slika 4. Skup podataka

Atributi podataka koji se koriste su:

1. **Pregnancies**: broj trudnoća kod pacijenta,
2. **Glucose**: nivo glukoze u krvi,
3. **Blood Pressure**: krvni pritisak pacijenta izražen u *mmHg*,
4. **Skin Thickness**: debljina kože,
5. **Insulin**: nivo insulina,
6. **BMI**: maksimalni broj otukcaja srca individue,
7. **DiabetesPedigreeFunction** : ukazuje na funkciju koja ocenjuje verovatnoću dijabetesa na osnovu porodične istorije,
8. **Age**: godine pacijenta,
9. **Outcome**: da li pacijent ima dijabetes ili ne.

## 4. Implementacija sistema za predikciju

Za implementaciju sistema za preporučivanje korišćen je Python skriptni jezik i dodatne biblioteke koje ovaj programski jezik nudi. Python predstavlja izuzetno popularan programski jezik koji se koristi i za mašinsko učenje i za web rudarenje (mining). U ovim oblastima, Python se široko koristi zbog svoje jednostavnosti, fleksibilnosti i velikog broja biblioteka otvorenog koda i okvira koji su mu dostupni.

### 4.1. Iskorišćene biblioteke

Kao što je već rečeno Python ima široku lepezu biblioteka, koje su kolekcije unapred napisanog koda koji se može koristiti za obavljanje određenih zadataka u programu. Ovo su neke od najpopularnijih biblioteka u Python-u koje su i iskorišćene u implementaciji:

- **NumPy:** Python biblioteka otvorenog koda koja se koristi u skoro svim oblastima nauke i inženjerstva. To je univerzalni standard za rad sa numeričkim podacima u Python-u. Korisnici NumPy-a uključuju svakoga, od početnika kodera do iskusnih istraživača koji rade najsavremenija naučna i industrijska istraživanja i razvoj. Ova biblioteka sadrži višedimenzionalne nizove i matricne strukture podataka. NumPy se može koristiti za izvođenje širokog spektra matematičkih operacija nad nizovima. On dodaje moćne strukture podataka koje garantuju efikasne proračune sa nizovima i matricama i obezbeđuje ogromnu biblioteku matematičkih funkcija visokog nivoa koje rade na ovim nizovima i matricama. Pruža podršku za velike nizove i matrice i uključuje funkcije za matematičke operacije kao što su linearna algebra, Furijeova transformacija i generisanje slučajnih brojeva.
- **Pandas:** paket otvorenog koda koji se najviše koristi za nauku o podacima/analizu podataka i zadatke mašinskog učenja. Izgrađen je na vrhu drugog paketa pod nazivom NumPy, koji pruža podršku za višedimenzionalne nizove. Kao jedan od najpopularnijih paketa za procesiranje podataka, Pandas dobro funkcioniše sa mnogim drugim modulima za nauku o podacima unutar Python ekosistema i obično je uključen u svaku Python distribuciju, od onih koje dolaze sa vašim operativnim sistemom do komercijalnih distribucija.
- **Scikit-learn:** scikit-learn (ranije scikits.learn i takođe poznat kao sklearn) je besplatna softverska biblioteka za mašinsko učenje za programski jezik Python. Posедуje različite algoritme za klasifikaciju, regresiju i grupisanje uključujući mašine za vektor podrške, nasumične šume, povećanje gradijenta, k-srednje vrednosti i dizajniran je da interoperiše sa Python numeričkim i naučnim bibliotekama NumPy i SciPy.

- **Streamlit:** Streamlit je open source framework na programskom jeziku Python. Pomaže nam da kreiramo web aplikacije za nauku o podacima i mašinsko učenje za kratko vreme. Kompatibilan je sa glavnim Python bibliotekama kao što su scikit-learn, Keras, PyTorch, SymPy, NumPy, pandas, Matplotlib itd.

```
import streamlit as stl
from streamlit_option_menu import option_menu
import pandas as pd
import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
```

Slika 5. Uključivanje biblioteka

#### 4.2. Preprocesiranje podataka

Nakon uključivanja potrebnih biblioteka sledeći korak predstavlja preprocesiranje podataka. Preprocesiranje podataka je suštinski korak u mašinskom učenju. To uključuje transformaciju neobrađenih podataka u format koji se lako može razumeti i analizirati algoritmima mašinskog učenja. Evo nekih uobičajenih tehnika predobrade:

- Čišćenje podataka: Ova tehnika uključuje uklanjanje ili ispravljanje nevažjećih podataka ili podataka koji nedostaju, kao što je uklanjanje duplikata zapisa, ispravljanje pravopisnih grešaka i popunjavanje vrednosti koje nedostaju, uklanjanje null-tih vrednosti podataka.
- Standardizacija podataka: To je još jedan integralni korak prethodne obrade podataka. Na primer ako imamo 2 numeričke vrednosti: Starost izražena u godinama i Težinu izraženu u kilogramima, one nisu na istoj skali i pošto je veća verovatnoća da će težina biti veća od starosti, stoga će naš model dati veću težinu težini, što nije idealan scenario jer je starost takođe jako bitan faktor. Da bismo izbegli ovaj problem, vršimo standardizaciju.
- Skaliranje karakteristika: Ovo uključuje skaliranje karakteristika skupa podataka tako da budu na sličnoj skali. Ovo može sprečiti određene karakteristike da dominiraju nad drugima i može olakšati algoritmu da nauči odnose između karakteristika.

- Kodiranje kategoričkih varijabli: Ovo uključuje pretvaranje kategoričkih varijabli (npr. pol, grad) u numeričke varijable tako da se mogu koristiti u algoritmima za mašinsko učenje.
- Smanjenje dimenzionalnosti: Ovo uključuje smanjenje broja karakteristika u skupu podataka, kao što je tehnika poput analize glavnih komponenti . Ovo može pomoći da se smanji složenost podataka i ubrza proces učenja.

Ove tehnike (ali i mnoge druge) mogu pomoći u poboljšanju tačnosti i performansi algoritama mašinskog učenja tako što će podatke učiniti pogodnijim za analizu.

Na sledećoj slici prikazano je preprocesiranje jednog skupa podataka. Nakon učitavanja seta podataka pre svega je izmenjena vrednost podataka kolone bmi koje je u ovom slučaju imala većinu vrednosti jednake *null*. Za taj slučaj iskorišćena je linearna interpolacija. Nakon toga sve tekstualne vrednosti su zamenjene su numeričkim. Na slici je prikazana obrada samo jednog skupa podataka jer su sve obrade slične.

```
dataset = pd.read_csv('dataset/healthcare-dataset-stroke-data.csv')
dataset[dataset['bmi'].isnull()]['smoking_status'].value_counts()
dataset['bmi'] = dataset['bmi'].interpolate(method='linear', limit_direction='forward')
dataset.drop(['id'], axis=1, inplace=True)
dataset['gender'] = dataset['gender'].replace(['Male'], 0.0)
dataset['gender'] = dataset['gender'].replace(['Female'], 1.0)
dataset['gender'] = dataset['gender'].replace(['Other'], 2.0)
dataset['ever_married'] = dataset['ever_married'].replace(['No'], 0.0)
dataset['ever_married'] = dataset['ever_married'].replace(['Yes'], 1.0)
dataset['work_type'] = dataset['work_type'].replace(['Private'], 0.0)
dataset['work_type'] = dataset['work_type'].replace(['Self-employed'], 1.0)
dataset['work_type'] = dataset['work_type'].replace(['Govt_job'], 2.0)
dataset['work_type'] = dataset['work_type'].replace(['children'], 3.0)
dataset['work_type'] = dataset['work_type'].replace(['Never_worked'], 4.0)
dataset['Residence_type'] = dataset['Residence_type'].replace(['Urban'], 0.0)
dataset['Residence_type'] = dataset['Residence_type'].replace(['Rural'], 1.0)
dataset['smoking_status'] = dataset['smoking_status'].replace(['Unknown'], 0.0)
dataset['smoking_status'] = dataset['smoking_status'].replace(['smokes'], 1.0)
dataset['smoking_status'] = dataset['smoking_status'].replace(['formerly smoked'], 2.0)
dataset['smoking_status'] = dataset['smoking_status'].replace(['never smoked'], 3.0)
```

Slika 6. Preprocesiranje skupa podataka

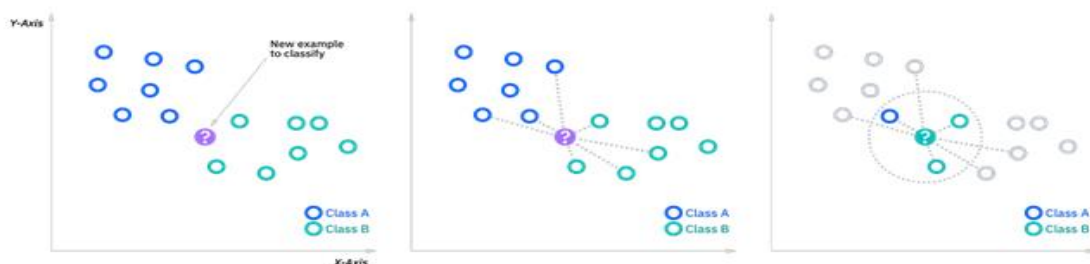
### 4.3. Iskorišćeni algoritmi

U prethodnom delu opisani su algoritmi koji se najčešće koriste u mašinskom učenju i pametnoj obradi podataka. Za neke od njih je rečeno da su iskorišćeni u implementaciji pomenute aplikacije za predikciju i oni će sada biti detaljnije opisani, a takodje će biti prikazan i kod i način kako su ti algoritmi iskorišćeni.

#### 4.3.1. KNN algoritam

Prvi iskorišćeni algoritam koji se koristi za predikciju mogućnosti za dobijanje srčanog udara kod pacijenta jeste KNN algoritam, odnosno *k nearest neighbors*- *k* najbližih suseda. Algoritam *k*-najbližih suseda, je neparametarski, nadgledani klasifikator učenja, koji koristi blizinu da napravi klasifikacije ili predviđanja o grupisanju pojedinačne tačke podataka. Iako se može koristiti za probleme regresije ili klasifikacije, obično se koristi kao klasifikacioni algoritam, radeći na pretpostavci da se slične tačke mogu naći jedna blizu druge.

Za probleme klasifikacije, oznaka klase se dodeljuje na osnovu većine glasova-tj. koristi se oznaka koja je najčešće predstavljena oko date tačke podataka. Vrednost *k* u *k*-NN algoritmu definiše koliko će suseda biti provereno da bi se odredila klasifikacija određene tačke upita. Na primer, ako je *k*=1, instanca će biti dodeljena istoj klasi kao njen najbliži sused. Definisavanje *k* može biti balansiranje jer različite vrednosti mogu dovesti do preopterećenja ili nedovoljnog prilagođavanja. Niže vrednosti *k* mogu imati veliku varijansu, ali nisku pristrasnost, a veće vrednosti *k* mogu dovesti do velike pristrasnosti i niže varijanse. Izbor *k* će u velikoj meri zavisiti od ulaznih podataka pošto će podaci sa više odstupanja ili šuma verovatno imati bolji učinak sa višim vrednostima *k*. Sve u svemu, preporučuje se da imate neparan broj za *k* da biste izbegli veze u klasifikaciji, a taktike unakrsne provere mogu vam pomoći da odaberete optimalni *k* za vaš skup podataka. Izbor parametra *k* je jako bitan i njegov uticaj na klasifikaciju je prikazan na sledećoj slici.



Slika 7. KNN algoritam

Kao što je prikazano na poslednjoj slici ako je  $k=3$ , od tri najbliža suseda dva od njih pripadaju klasi B, dok treći sused pripada klasi A. Zbog toga će tački označena znakom '?' biti dodeljena klasa B.

Što se tiče implementacije aplikacije i koda na sledećim slikama su prikazani delovi koda u kome se KNN klasifikator koristi za predikciju.

Python kod i njegove biblioteke, u ovom konkretnom slučaju biblioteka *Scikit-learn* omogućava korišćenje KNN klasifikatora i algoritma izuzetno jednostavno, brzo i lako. Zbog toga je Python izuzetno popularan u mašinskom učenju i Web mining-u. Karakteristika ovog jezika jeste ta većina biblioteka nudi već obučene modele kojima je samo potrebno proslediti podatke koje treba obraditi i koji će iskoristiti odmah te podatke na pravi način. Medjutim ako je pak potrebno obučiti model, to je omogućeno završiti u svega nekoliko linija koda.

Da bi se iskorstio ovaj klasifikator potrebno je dodatno obraditi podatke, odnosno odvojiti trening i test skup podataka i skalirati te podatke korišćenjem *StandardScaler* klase kao što je prikazano na sledećoj slici.

```
heart_disease_dataset = pd.read_csv('D:/WebMining/dataset/heart.csv')
x= heart_disease_dataset.iloc[:,0:13].values
y= heart_disease_dataset['target'].values
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)
st_x = StandardScaler()
x_train = st_x.fit_transform(x_train)
x_test = st_x.transform(x_test)
```

Slika 8. Procesiranje podataka

Nakon toga proverava se najbolja vrednost za parametar  $k$  koja odgovara datom skupu podataka. Preciznije rečeno proverava se tačnost za različite vrednosti u opsegu od 1 do 29 i kreira se klasifikator sa vrednošću parametra koji je proizveo najmanju stopu greške prilikom testiranja.

```
for i in range(1, 30):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(x_train, y_train)
    pred_i = knn.predict(x_test)
    error.append(np.mean(pred_i != y_test))
K = error.index(min(error))+1
classifier = KNeighborsClassifier(n_neighbors = K)
```

Slika 9. Izbor vrednosti za parametar  $k$  i kreiranje klasifikatora



Nakon toga vrši se obuka klasifikatora trening podacima, korišćenjem funkcije *fit*, a zatim se za predikciju koriste ulazni podaci koje je korisnik uneo i koji su smešteni u niz nazvan *parameters*. Ti podaci dobijeni su preko *streamlit input* polja.

```
Age = st.number_input('Age of the person')
Sex = st.number_input('Gender of the person: 0-female, 1-male')
ChestPain = st.number_input('Type of chest-pain: 1 = typical angina, 2 = atypical angina, 3 = non - angina pain, 4 = asymptotic')
BloodPressure = st.number_input('Blood pressure value in mmHg ')
SerumCholesterol = st.number_input('Serum cholesterol in mg/dl')
FastingBloodSugar = st.number_input('If fasting blood sugar > 120mg/dl then : 1, else : 0 ')
ECG = st.number_input('Resting electrocardiographic results: 0 = normal, 1 = having ST-T wave abnormality, 2 = left ventricular hypertrophy')
HeartRate = st.number_input('Max heart rate ')
Angina = st.number_input('Exercise induced angina : 1 = yes, 0 = no ')
STDepression = st.number_input('ST depression induced by exercise relative to rest: integer or float number')
STSegment = st.number_input('Peak exercise ST segment : 1 = upsloping, 2 = flat ,3 = downsloping')
Vessels = st.number_input('Number of major vessels (0-3) colored by fluoroscopy: displays the value as integer or float')
Thal = st.number_input('Thal : displays the thalassemia : 3 = normal, 6 = fixed defect, 7 = reversible defect')
```

Slika 10. Lični podaci koje korisnik unosi

```
classifier = KNeighborsClassifier(n_neighbors = K)
classifier.fit(x_train, y_train)
parameters = [[Age, Sex, ChestPain, BloodPressure, SerumCholesterol, FastingBloodSugar, ECG, HeartRate, Angina, STDepression, STSegment,
parameters = st_x.transform(parameters)
prediction = classifier.predict(parameters)
```

Slika 11. Predikcija bolesti srca

Kao rezultat dobija se niz *prediction* koji ima samo jedan element koji može biti 0 ako korisnik verovatno neće dobiti srčani udar ili 1 ako korisnik ima velike šanse za dobijanje srčanog udara. Nakon provere te vrednosti korisnik se obaveštava o rezultatu.

```
if (prediction[0] == 1):
    diagnosis = 'The person has a high chance of having a heart attack.'
else:
    diagnosis = 'The person does not have a high chance of having a heart attack.'
st.success(diagnosis)
```

Slika 12. Obaveštavanje korisnika

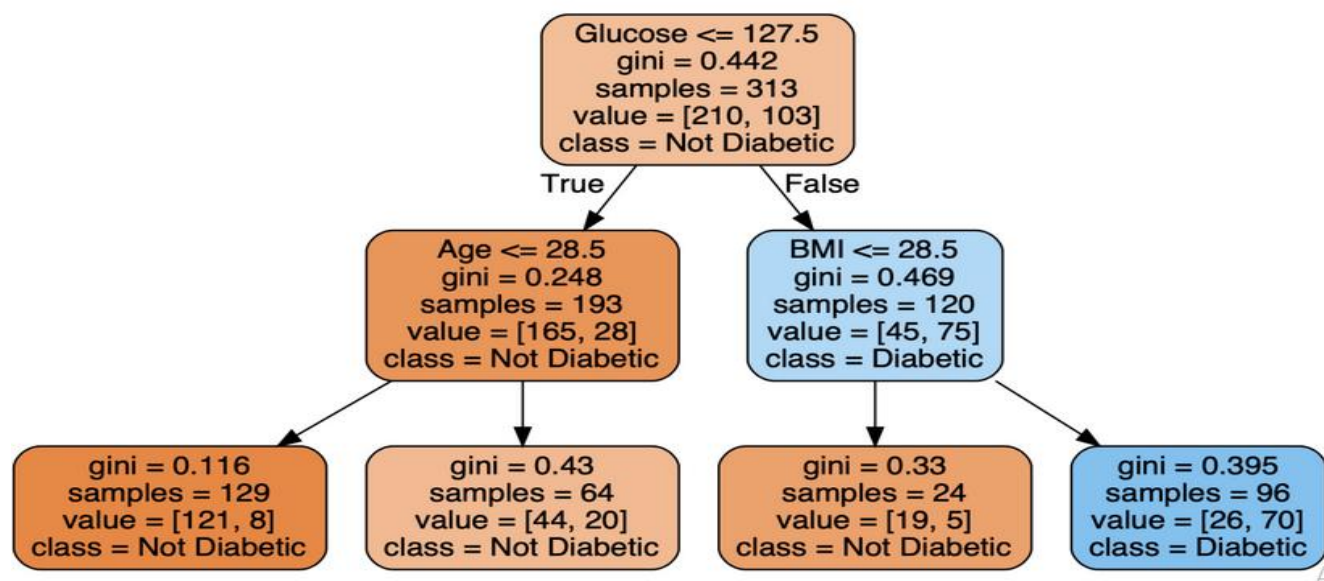


#### 4.3.2. Decision Tree algoritam

Stablo odlučivanja je alat za podršku odluka koji koristi grafikon u obliku stabla ili model odluka i njihovih mogućih posledica, uključujući i šansu ishoda događaja, troškove resursa i korisnosti. To je jedan način da se prikaže algoritam. Obično su se koristila u operacionim istraživanjima, posebno u analizi odluke, da pomognu u identifikovanju strategije kojom će se najverovatnije postići cilj.

Stabla odlučivanja su poznata kao modeli „bele kutije“, što znači da možete lako pronaći i protumačiti njihove odluke. Ovo je u suprotnosti sa neuronskim mrežama 'crne kutije' gde je izuzetno teško shvatiti kako su tačno napravljena konačna predviđanja. Na sreću, modele stabla odlučivanja je lako objasniti jednostavnim rečima, zajedno sa razlogom i kako je model napravio predviđanja.

Stabla odlučivanja mogu se lako vizualizovati u dijagramu nalik stablu što čini još lakšim razumevanje i tumačenje modela. Na sledećoj slici prikazano je pojednostavljeno stablo odlučivanja. Mi zapravo možemo uzeti jednu tačku podataka i pratiti putanju koja bi prošla da bi se došlo do konačnog predviđanja za nju.



Slika 13. Stablo odlučivanja

Glavni cilj je da se podaci podele na različite regione i da se potom naprave predviđanja na osnovu tih regiona. Počevši od vrha stabla, prvo pitanje na koje algoritam mora da odgovori je „Koju osobinu treba izabrati u korenu?“. Da bi odgovorio na ovo pitanje, algoritam treba način da proceni svaku funkciju i da odabere „najbolju“ funkciju za početak. Nakon toga, algoritam treba da nastavi da postavlja slično pitanje na svakom čvoru: „Koju funkciju treba koristiti za razdvajanje ovog čvora?“. To radi tako što izračunava i optimizuje metriku u odnosu na svaku od dostupnih funkcija.

Postoji nekoliko metrika koje se mogu koristiti u zavisnosti od problema. Na primer, ako imamo posla sa problemom regresije, onda možemo tražiti funkciju sa najnižim RSS-om (preostali zbir kvadrata). Međutim, ako imamo problem sa klasifikacijom, onda možemo izabrati osobinu sa najnižom entropijom (ili najvećim dobitkom informacija) ili najnižom gini nečistoćom. Scikit-learn podrazumevano koristi gini nečistoću. Gini nečistoća je metrika koja meri čistoću čvora. To jest, meri koliko su zapažanja slična jedna drugoj. Ako sva zapažanja pripadaju istoj klasi, onda bi gini nečistoća bila 0 i čvor bi se smatrao „čistim“.

Algoritam stabla odlučivanja je u stalnoj potrazi za čistim čvorovima i nastaviće da deli podatke u sve dublja i dublja stabla sve dok konačno ne dostigne čiste lisne čvorove (ili mu ponestane podataka ili karakteristika).

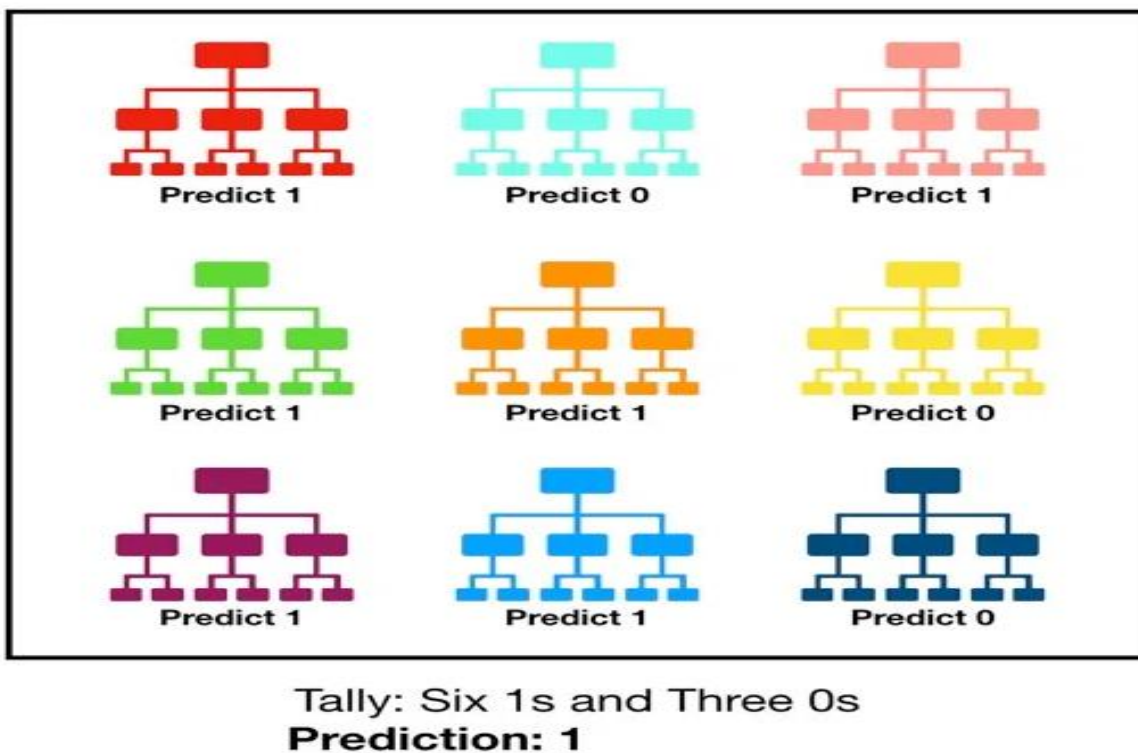
Ovaj algoritam iskorišćen je za predikciju mogućnosti da pacijent dobije dijabetes i zahvaljujući *Python* programskom jeziku izuzetno se jednostavno koristi u kodu. S obzirom na to da je u prethodnom slučaju detaljno opisan način dodatne obrade podataka i razdvajanje u test i trening skupove, kao i kreiranje niza ulaznih podataka koje korisnik unosi, taj deo će biti izostavljen sada i na sledećoj slici prikazan je samo deo koda koji pokazuje kreiranje klasifikatora arame odlučivanja i njegovo korišćenje za predikciju.

```
DecisionTree = tree.DecisionTreeClassifier(random_state = 42)
DecisionTree = DecisionTree.fit(x_train, y_train)
parameters = [[Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age]]
prediction = DecisionTree.predict(parameters)
if (prediction[0] == 1):
    diab_diagnosis = 'The person is diabetic.'
else:
    diab_diagnosis = 'The person is not diabetic.'
str1.success(diab_diagnosis)
```

Slika 14. Kreiranje DecisionTree klasifikatora u kodu i predikcija

#### 4.3.3. Random Forest algoritam

*Random forest* – Slučajne šume ili šume slučajnih odluka su metoda učenja ansambla za klasifikaciju, regresiju i druge zadatke koji funkcioniraju tako što se konstruira mnoštvo stabala odlučivanja u vreme obuke. Za zadatke klasifikacije, izlaz nasumične šume je klasa koju bira većina stabala. Za zadatke regresije, vraća se srednja vrednost ili prosečno predviđanje pojedinačnih stabala. Slučajne šume generalno nadmašuju arame odlučivanja. Karakteristike podataka mogu uticati na njihov učinak. Osnovni aramet iza nasumične šume je jednostavan, ali moćan – mudrost gomile. U nauci o podacima, razlog zašto model slučajne šume tako dobro funkcioniraju je veliki broj relativno nekoreliranih modela (stabala) koji rade kao komitet će nadmašiti bilo koji od pojedinačnih konstitutivnih modela. Na sledećoj slici prikazan je princip funkcionisanja ovog algoritma.



Slika 15. Random forest algoritam

Python programski jezik kao Scikit-learn algoritam i u ovom slučaju omogućuju pisanje izuzetno jednostavnog koda koji izvršava izuzetno složene algoritme i funkcionalnosti. Na sledećoj slici je prikazan deo koda koji predstavlja korišćenje RandomForest klasifikatora za predikciju. Ostali deo koda koji predstavlja dodatnu pripremu podataka, preuzimanje ulaznih podataka od korisnika i kreiranje niza od tih podataka neće biti prikazan kako ne bi došlo do ponavljanja zato što je princip isti kao u prethodnom slučaju.

```

rf=RandomForestClassifier()
rf.fit(x_train,y_train)
prediction=rf.predict(parameters)
if (prediction[0] == 1):
    diab_diagnosis = 'The person has a high chance of having a stroke.'
else:
    diab_diagnosis = 'The person has not a high chance of having a stroke.'
str1.success(diab_diagnosis)

```

Slika 16. Random forest algoritam za predikciju moždanog udara u kodu

Što se tiče klase *RandomForestClassifier* njene *default*-ne vrednosti su prikazane na sledećoj slici i kao što se može videti u kodu, nijedan parametar nije primenjen prilikom kreiranja objekta te klase.

```

RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                        criterion='gini', max_depth=None, max_features='auto',
                        max_leaf_nodes=None, max_samples=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, n_estimators=100,
                        n_jobs=None, oob_score=False, random_state=0, verbose=0,
                        warm_start=False)

```

Slika 17. RandomForestClassifier klasa

## 5. Provera rezultata i testiranje tačnosti

Pre primene pomenutih algoritama nad podacima koje korisnik unosi, potrebno je odrediti preciznost i tačnost njihovih rada, odnosno odrediti u kolikom procentu su rezultati predikcije tačni.

Postoje četiri načina da proverite da li su predviđanja tačna ili pogrešna:

- **TN / True Negative:** slučaj je bio negativan i predviđao negativan,
- **TP / True Positive:** slučaj je bio pozitivan i predviđano pozitivan,
- **FN / False Negative:** slučaj je bio pozitivan, ali je predviđao negativan,
- **FP / False Positive:** slučaj je bio negativan, ali je predviđao pozitivan.

Za proveru u kodu koristi se funkcija *classification\_report* koja kao rezultat vraća četiri različita parametra koji su povezani sa prethodno definisanim terminima:

- **Precision,**
- **Recall,**
- **F1 score,**
- **Support.**

*Precision*, odnosno preciznost je sposobnost klasifikatora da ne označi instancu pozitivnom koja je zapravo negativna. Za svaku klasu se definiše kao odnos istinito pozitivnih i sume istinito pozitivnih i lažno pozitivnih. Ukratko, preciznost predstavlja tačnost pozitivnih predviđanja.

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

*Recall* je sposobnost klasifikatora da pronađe sve pozitivne primere. Za svaku klasu se definiše kao odnos istinitih pozitivnih rezultata prema zbiru istinitih pozitivnih i lažno negativnih. Ukratko, *recall predstavlja deo pozitivnih koji su tačno identifikovani*.

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

*F1 score* je prosečna vrednost *precision*-a i *recall*-a tako da je najbolji rezultat 1.0, a najgori 0.0. F1 rezultati su niži od mera tačnosti jer ugrađuju vrednosti drugih pomenutih parametara u svoje proračune. Kao pravilo, za poređenje modela klasifikatora treba koristiti prosečnu vrednost F1, a ne globalnu tačnost. Ukratko, ovaj parametar odgovara na pitanje "Koji procenat pozitivnih predviđanja je bio tačan?".

$$F1 \text{ score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

*Support* je broj stvarnih pojavljivanja klase u navedenom skupu podataka. Neuravnotežena podrška u podacima o obuci može ukazivati na strukturne slabosti u prijavljenim ocenama klasifikatora i može ukazati na potrebu za stratifikovanim uzorkovanjem ili rebalansom. Podrška se ne menja između modela, već dijagnostikuje proces evaluacije.

Procena tačnosti u kodu pomoću pomenute funkcije proverava se na sledeći način:

```
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
prediction = rf.predict(x_test)
accuracy = accuracy_score(y_test,prediction)
print(prediction)
print("Accuracy score of prediction is ",accuracy*100,'%.')
print(classification_report(y_test, prediction))
```

Slika 18. Testiranje rezultata i tačnosti predikcije u kodu

Nakon testiranja svih klasifikatora, odnosno tačnosti algoritama dobijeni su sledeći rezultati.

Tačnost KNN algoritma je približno 86%.

Accuracy score of prediction is 86.8421052631579 %.

	precision	recall	f1-score	support
0	0.90	0.79	0.84	33
1	0.85	0.93	0.89	43
accuracy			0.87	76
macro avg	0.87	0.86	0.86	76
weighted avg	0.87	0.87	0.87	76

Slika 19. Valjanost rezultata predviđanja KNN klasifikatora

Tačnost RandomForest algoritma je približno 94%.

Accuracy score of prediction is 93.9334637964775 %.

	precision	recall	f1-score	support
0	0.94	1.00	0.97	960
1	0.00	0.00	0.00	62
accuracy			0.94	1022
macro avg	0.47	0.50	0.48	1022
weighted avg	0.88	0.94	0.91	1022

Slika 20. Valjanost rezultata predviđanja RandomForest klasifikatora

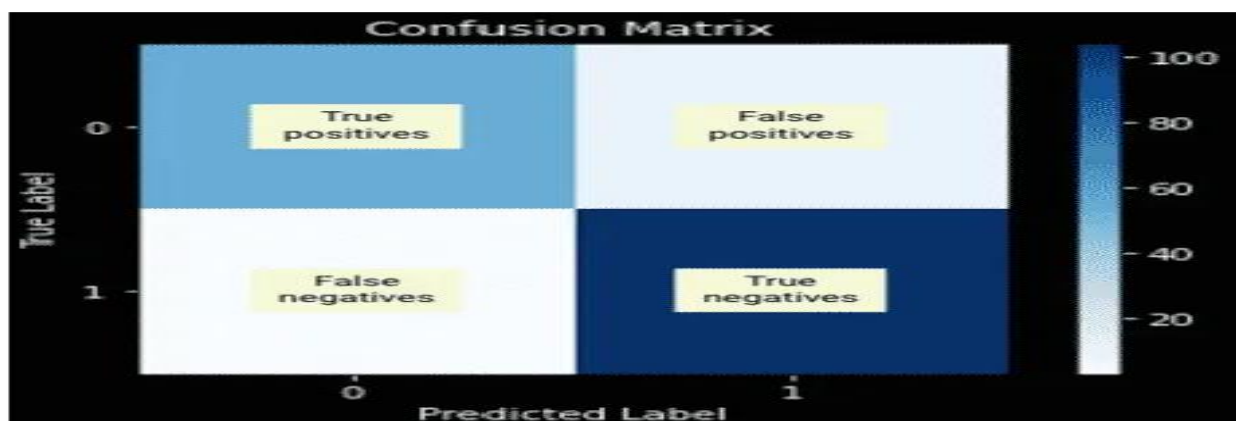
Tačnost DecisionTree algoritma je približno 75%.

Accuracy score of prediction is 74.67532467532467 %.

	precision	recall	f1-score	support
0	0.83	0.76	0.79	99
1	0.62	0.73	0.67	55
accuracy			0.75	154
macro avg	0.73	0.74	0.73	154
weighted avg	0.76	0.75	0.75	154

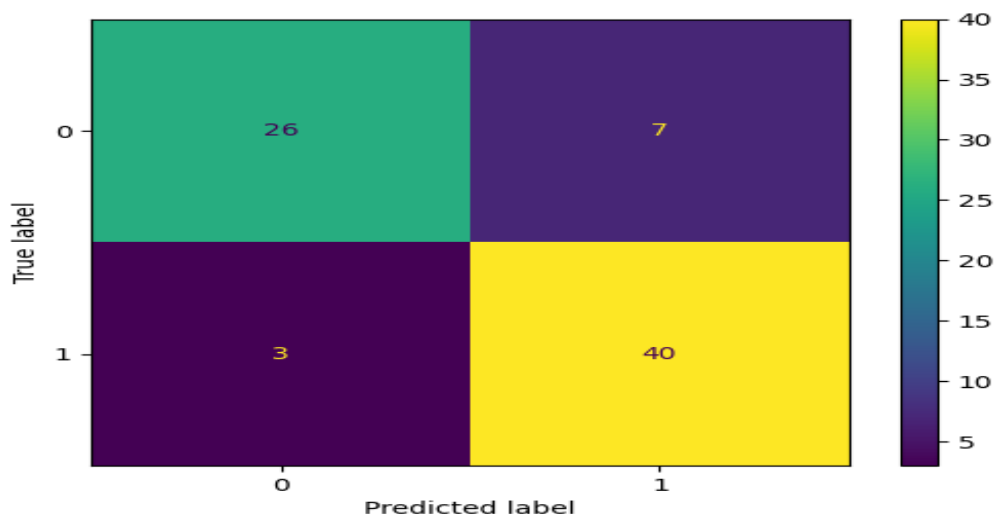
Slika 21. Valjanost rezultata predviđanja Decision Tree klasifikatora

Još jedan od načina da se prikaže valjanost predikcije, odnosno tačnost rezultata jeste *confusion matrix* u *python*-u. Na sledećoj slici je prikazano značenje svakog polja matrice, kako bi rezultati bili razumljivi.



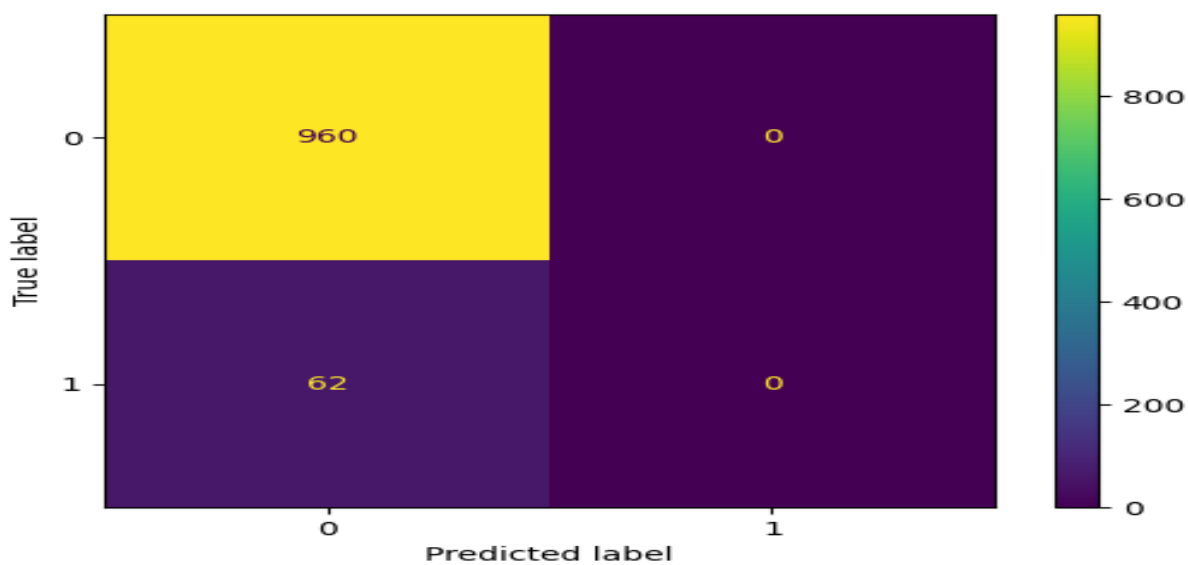
Slika 21. Confusion matrix

Konfuziona matrica za KNN klasifikator izgleda ovako:



Slika 22. Konfuziona matrica za KNN algoritam

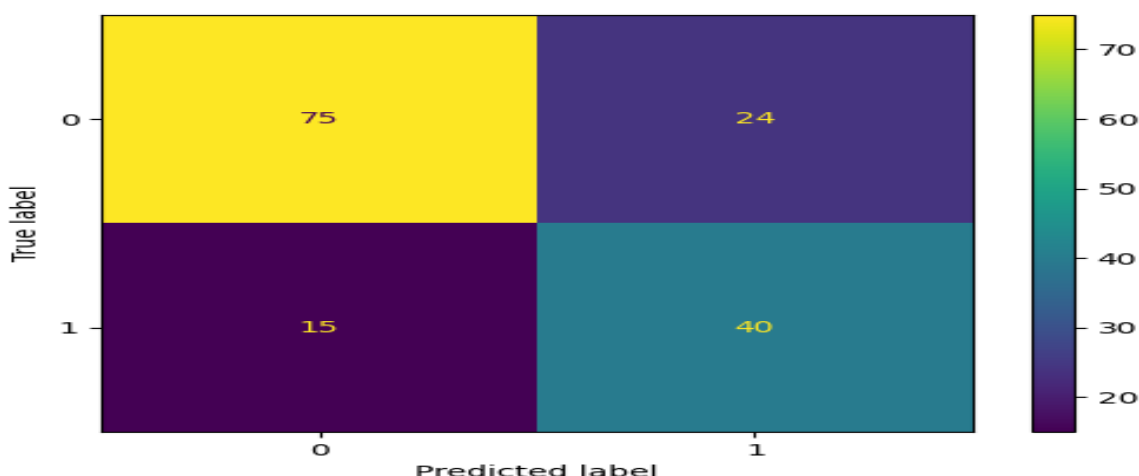
Konfuziona matrica za Random Forest klasifikator prikazan je na sledećoj slici.



Slika 23. Konfuziona matrica za Random Forest klasifikator



Konfuziona matrica za Decission Tree algoritam izgleda ovako:



Slika 24. Konfuziona matrica za Decission Tree algoritam

Kada detaljno proučimo prikazane rezultate lako je uočljivo da je od sva tri klasifikatora najtačniji i najprecizniji Random Forest, kao što je i predviđeno. Njegova tačnost predviđanja je čak 94% što znači da se sa velikom sigurnošću možemo osloniti na predikciju ovog klasifikatora. To prikazuje i konfuziona matrica čija je slika prikazana iznad.

Nakon ovog klasifikatora po tačnosti i preciznosti sledi KNN klasifikator, čije je tačnost 86%, što znači da je i on izuzetno precizan. Kao što se može videti na slici iznad najprecizniji je u predviđanju negativnih rezultata.

Na kraju, najmanje precizan klasifikator je Decission Tree algoritam sa izmerenom tačnošću od 75%. Kao što je prikazano na slici iznad ovaj algoritam pravi greške i kod pozitivnih i kod negativnih rezultata. Medjutim 75% nije malo, ali ipak moramo uzeti rezultate predviđanja ovog klasifikatora sa rezervom i ne smemo biti jako sigurni u njegovu tačnost.

Medjutim, iako naravno postoje greške, neverovatna je preciznost ovih predviđanja i to što se na izuzetno jednostavan način postižu ovakvi rezultati. Kao što vidimo, algoritmi su precizni i verovatno bi bili još precizniji i tačniji da su istrenirani na robusnijem skupu podataka. Problem je naravno što je za prikupljanje takvih podataka potrebna saglasnost pacijenata, pa zbog toga korišćeni skupovi nemaju veliku raznovrsnost podataka kao i veliki broj podataka.

## 6. Izgled aplikacije

Inicijalni izgled aplikacije nakon pokretanja prikazan je na sledećim slikama.

**Heart Disease prediction**

Age of the person

0.00 - +

Gender of the person: 0-female, 1-male

0.00 - +

Type of chest-pain: 1 = typical angina, 2 = atypical angina, 3 = non — angina pain, 4 = asymptotic

0.00 - +

Blood pressure value in mmHg

0.00 - +

Serum cholesterol in mg/dl

0.00 - +

If fasting blood sugar > 120mg/dl then : 1, else : 0

0.00 - +

Activate Windows  
Go to Settings to activate Windows

Slika 25. Izgled aplikacije

0.00 - +

Exercise induced angina : 1 = yes, 0 = no

0.00 - +

ST depression induced by exercise relative to rest: integer or float number

0.00 - +

Peak exercise ST segment : 1 = upsloping, 2 = flat, 3 = downsloping

0.00 - +

Number of major vessels (0-3) colored by fluoroscopy: displays the value as integer or float

0.00 - +

Thal : displays the thalassemia : 3 = normal, 6 = fixed defect, 7 = reversible defect

0.00 - +

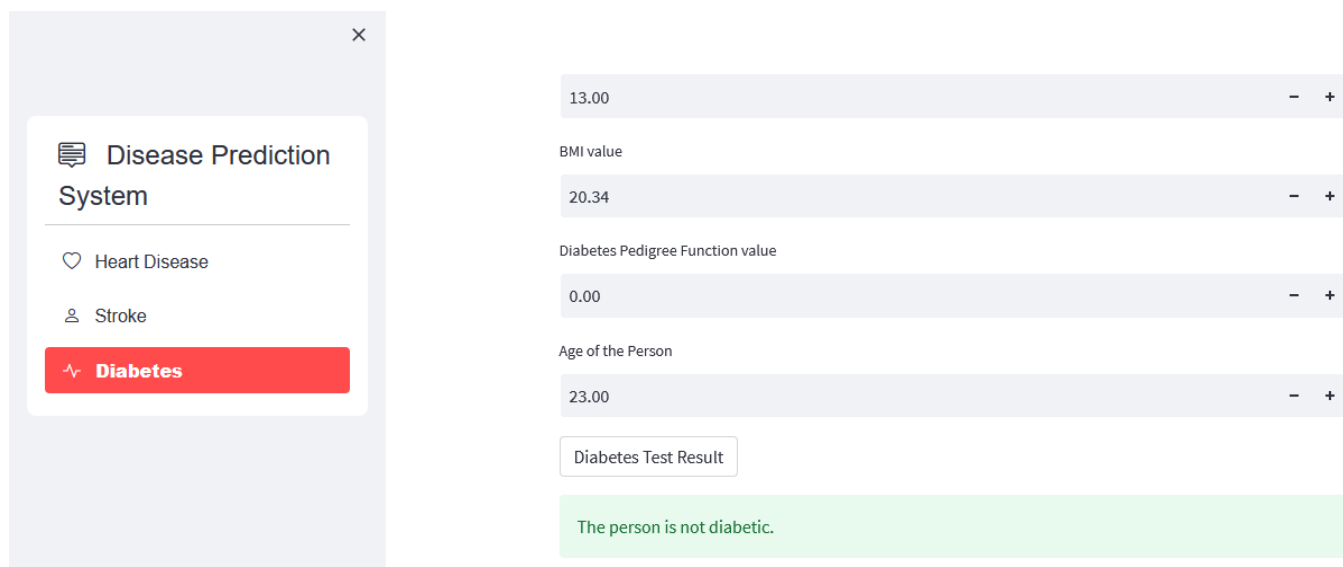
Heart Disease Test Result

Activate Windows  
Go to Settings to activate Windows

Slika 26. Izgled aplikacije

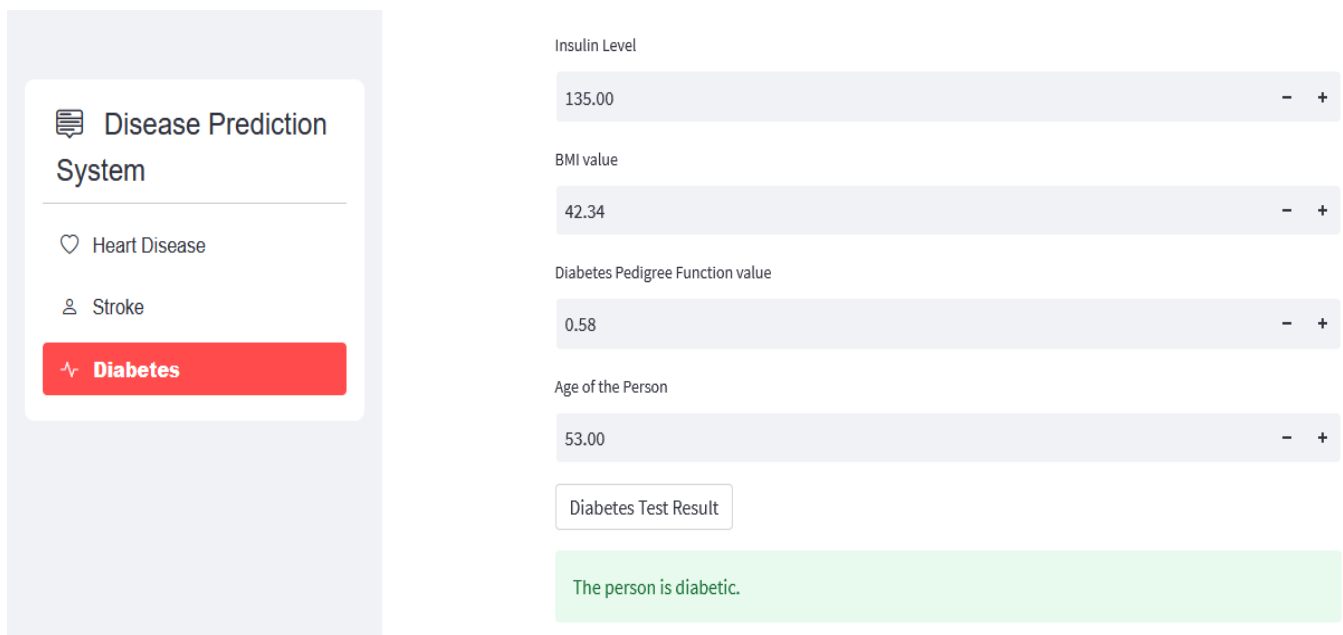
Isprobala sam rad aplikacije za testiranje da li je pacijent dijabetičar ili ima velike šanse za sticanje te bolesti, odnosno da li pacijent nije dijabetičar ili nema velike šanse za sticanje te bolesti.

Na prvoj slici prikazan je rezultat nakon unošenja parametra koji odgovaraju potpuno zdravoj osobi, dok je na drugoj prikazan rezultat nakon unošenja parametara koji su karakteristični za dijabetičare.



The screenshot shows a web application titled "Disease Prediction System". On the left, there is a sidebar with three options: "Heart Disease", "Stroke", and "Diabetes" (highlighted in red). The main area displays input fields for "Age of the Person" (13.00), "BMI value" (20.34), and "Diabetes Pedigree Function value" (0.00). Below these is a "Diabetes Test Result" button. The output area shows a green box with the text "The person is not diabetic."

Slika 27. Izgled aplikacije nakon izvršene predikcije



The screenshot shows the same web application as the previous one, but with different input values. The sidebar remains the same. The main area displays input fields for "Age of the Person" (53.00), "BMI value" (42.34), and "Diabetes Pedigree Function value" (0.58). Below these is a "Diabetes Test Result" button. The output area shows a green box with the text "The person is diabetic."

Slika 27. Izgled aplikacije nakon izvršene predikcije

## 7. Zaključak

Ovaj rad ima za cilj da pokaže kako upotreba tehnika web mining-a za predviđanje bolesti ima potencijal da revolucionirše oblast zdravstvene zaštite i lečenja pacijenata. Predviđanje bolesti u web mining-u se odnosi na proces analize i modeliranja velikih količina zdravstvenih podataka kako bi se identifikovali obrasci i odnosi koji se mogu koristiti za predviđanje verovatnoće razvoja bolesti. Ovaj pristup se može koristiti da pomogne zdravstvenim radnicima u postavljanju tačnijih i pravovremenih dijagnoza, kao i da pomogne pacijentima da preduzmu preventivne mere kako bi izbegli ili upravljali potencijalnim zdravstvenim rizicima. Neverovatan napredak tehnologije omogućava da se ovako složeni proces, za koje se ljudi pripremaju i uče godinama može izvršiti pomoću koda koji je prethodno opisan. Iskorišćeno je svega tri različita algoritma, ali je ukupan broj algoritama koje Python programski jezik nudi daleko veća. Sa poboljšanjem skupova podataka njihova tačnost biće i daleko veća.

Međutim, važno je napomenuti i da korišćenje veb mining-a za predviđanje bolesti takođe izaziva zabrinutost oko privatnosti i bezbednosti podataka. Kao takav, razvoj ove tehnologije mora da bude praćen čvrstim etičkim smernicama i propisima kako bi se osiguralo da su informacije o pacijentima zaštićene. Sve u svemu, potencijalne prednosti ove aplikacije čine je uzbudljivom pojavom u zdravstvenoj industriji koja ima moć da transformiše način na koji pristupamo prevenciji i upravljanju bolestima.

Sve u svemu, predviđanje bolesti u web mining-u je polje koje se brzo razvija sa ogromnim potencijalom za poboljšanje ishoda pacijenata i smanjenje troškova zdravstvene zaštite. Uz kontinuirani rast zdravstvenih podataka na vebu, primena tehnika veb rudarenja za predviđanje bolesti će verovatno postati sve važnija u narednim godinama.

## 8. Literatura

- [1] [https://www.matec-conferences.org/articles/mateconf/pdf/2018/35/mateconf\\_ifid2018\\_01033.pdf](https://www.matec-conferences.org/articles/mateconf/pdf/2018/35/mateconf_ifid2018_01033.pdf)
- [2] <https://www.taylorfrancis.com/books/mono/10.4324/9780203491683/analysis-time-series-chris-chatfield>
- [3] <http://mines.humanoriented.com/classes/2010/fall/csci568/papers/ann.pdf>
- [4] [https://www.cc.gatech.edu/fac/Charles.Isbell/classes/2008/cs7641\\_spring/handouts/yor12-introsvm.pdf](https://www.cc.gatech.edu/fac/Charles.Isbell/classes/2008/cs7641_spring/handouts/yor12-introsvm.pdf)
- [5] <https://towardsdatascience.com/top-machine-learning-algorithms-for-classification-2197870ff501>
- [6] <https://bebafarm.rs/sluzbe/kardiologija/>
- [7] <https://www.analyticsvidhya.com/blog/2021/07/diabetes-prediction-with-pycaret/>
- [8] [https://numpy.org/doc/stable/user/absolute\\_beginners.html](https://numpy.org/doc/stable/user/absolute_beginners.html)
- [9] <https://www.activestate.com/resources/quick-reads/what-is-pandas-in-python-everything-you-need-to-know/>
- [10] <https://en.wikipedia.org/wiki/Scikit-learn>
- [11] <https://towardsdatascience.com/introduction-to-data-preprocessing-in-machine-learning-a9fa83a5dc9d>
- [12] <https://cloud.google.com/architecture/data-preprocessing-for-ml-with-tf-transform-pt1>
- [13] [https://en.wikipedia.org/wiki/Random\\_forest](https://en.wikipedia.org/wiki/Random_forest)
- [14] <https://towardsdatascience.com/understanding-random-forest-58381e0602d2>
- [15] <https://medium.com/@kohlshivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397>