Klaster analiza nad skupom podataka fizičko-hemijskih osobina vina

Seminarski rad u okviru kursa Analiza velikih podataka
Univerzitet u Novom Sadu
Prirodno-matematički fakultet

Nikola Pujaz - 21m/19

18. septembar 2020.

Sažetak

Koristeći skup podataka fizičko-hemijskih osobina vina u ovom radu je urađena klaster analiza pomenutog skupa sa ciljem odabira adekvatne vrednosti k, kao i razumevanja odnosa gustine vina i ostalih fizičko-hemijskih osobina u perspektivi segregacije podataka na k-klastera. Uz prethodno navedeno prikazane su i dve tehnike vizualizacije n-dimenzionalnih rezultata klasterovanja, i kreiran je model koji se može koristiti u daljem istraživanju.

Sadržaj

1 Uvod

Analiza velikih podataka, zajedno sa mašinskim učenjem, omogućava pristup različitim tehnikama koje se mogu koristiti za automatsku procenu kvaliteta hrane i pića, razumevanja podataka koji opisuju fizičko-hemijske osobine hrane i pića, kao i pronalaženja međusobnih sličnosti između podataka i grupisanje podataka na osnovu tih sličnosti.

Čovek usavršava tehnike produkcije vina od početka civilizacije pa sve do danas. Napretkom nauke i tehnologije omogućeno je bolje razumevanje fizičko-hemijskih osobina supstance, pa samim tim i dublje razumevanje ovih osobina, kao i kakav je njihov međusobni odnos i u kojoj meri one čine vino kvalitetnim. Ovaj pristup je sigurniji i pruža više mogućnosti od kvalitativne ocene vinskih stručnjaka, čiji se metodi zasnivaju na njihovom iskustvu i sposobnosti čula vida, mirisa i ukusa.

Skup podataka koji je korišćen u ovom radu je izdat od strane *University of California Irvine* i može se pronaći na [3].

U ovom radu je izvršena klaster analiza prethodno navedenog skupa podataka sa ciljem adekvatnog izbora broja klastera i razumevanjem odnosa gustine vina (*'density'*) i ostalih fizičko-hemijskih osobina u perspektivi segregacije podataka na klastere.

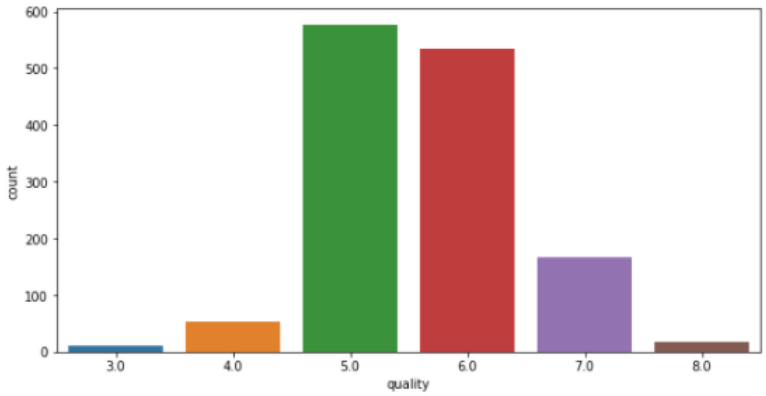
2 Analiza i pretprocesiranje podataka

Za realizaciju ovog rada korišćeno je *Google Colab* okruženje, kao i *Apache Spark* okruženje, programski jezik *Python* i njegove konstrukcije i druge biblioteke za analizu podataka i njihovu vizualizaciju. Implementacija ovog rešenja se nalazi u *Jupyter Notebook* datoteci u prilogu ovog rada.

Skup podataka koji je analiziran u ovom radu sadrži podatke o fizičko-hemijskim osobinama crvenog vina koje definiše 12 atributa sa sledećim nazivima: *'fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density', 'pH', 'sulphates', 'alcohol', 'quality'*, a poslednji atribut se smatra klasnim atributom, koji se u problemima klasifikacije koristi kao ciljni atribut - atribut koji određuje klasu instance. Ukupan broj instanci skupa iznosi 1599.

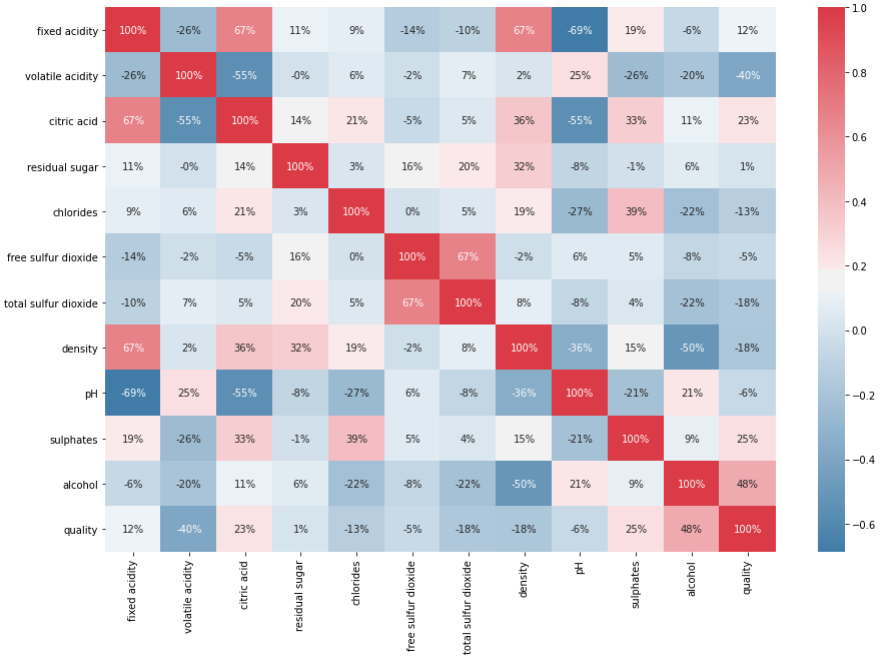
U narednom koraku je ispitana čistoća podataka, pri čemu su uklonjenji duplikati, dok instanci sa nedostajućim vrednostima atributa nije bilo. Ukupan broj instanci skupa nakon uklanjanja 240 duplikata iznosi 1359.

Pregledom tipova atributa ustanovljeno je da su sve vrednosti atributa u *DataFrame* strukturi tipa *string*, pa je bilo potrebno pretvoriti vrednosti atributa u odgovarajući *double* tip koristeći *pyspark.sql.types* biblioteku, nakon čega je izvršen pregled broja instanci prema pripadajućim klasama prikazan na slici 1.



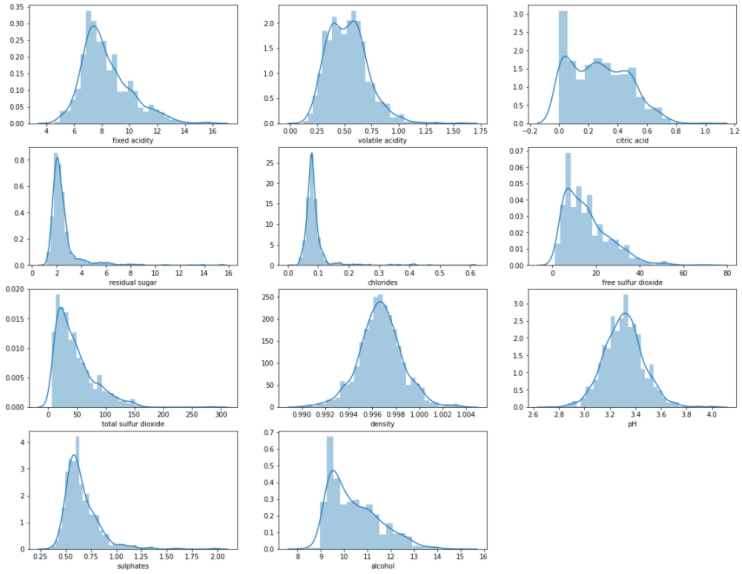
Slika 1: Pregled broja instanci po pripadajućim klasama

U sledećoj fazi razumevanja skupa podataka napravljen je pregled statističkih korelacija između atributa, korišćenjem Pirsonovog koeficijenta korelacije (eng. *Pearson correlation coefficient*) koji se kreće u intervalu između -1 i +1 i zahteva linearnu povezanost i neprekidnu normalnu distribuciju između promenljivih.[1] Korelacija je prikazana toplotnom kartom na slici 2, sa koje se može primetiti da postoje atributi koji imaju visoku međusobnu korelaciju.

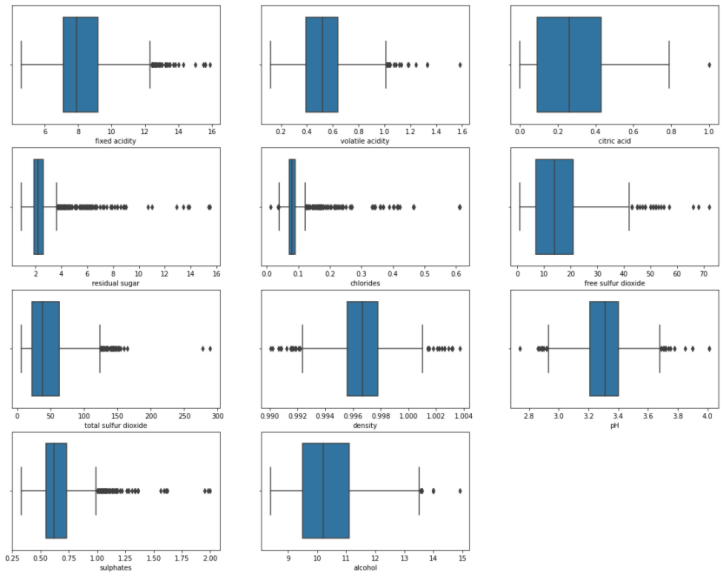


Slika 2: Pregled međusobnih zavisnosti između atributa

Na slici 3 su prikazane distribucije vrednosti atributa za svaki atribut, pri čemu se može primetiti potencijalno prisustvo *outlier* vrednosti na krajevima skupova vrednosti atributa, tj. vrednosti koje odstupaju od ostalih podataka i mogu u velikoj meri da utiču na rezultate učenja i evaluacije modela ukoliko se njihovo prisustvo jednostavno ignoriše. Bolja identifikacija *outlier* vrednosti omogućena je primenom *box-and-whisker* dijagrama, prikazanog na slici 4. Ovaj dijagram prikazuje podatke od donjeg do gornjeg kvartila. Iako se gornja i donja granica mogu različito definisati, najčešće predstavljaju najmanju i najveću vrednost koja se nalazi unutar 1.5 umnoženog interkvartilnog raspona gledajući od donjeg, odnosno gornjeg kvartila. Sve tačke izvan tih granica se smatraju *outlier* vrednostima.[2]



Slika 3: Pregled distribucija vrednosti po atributima

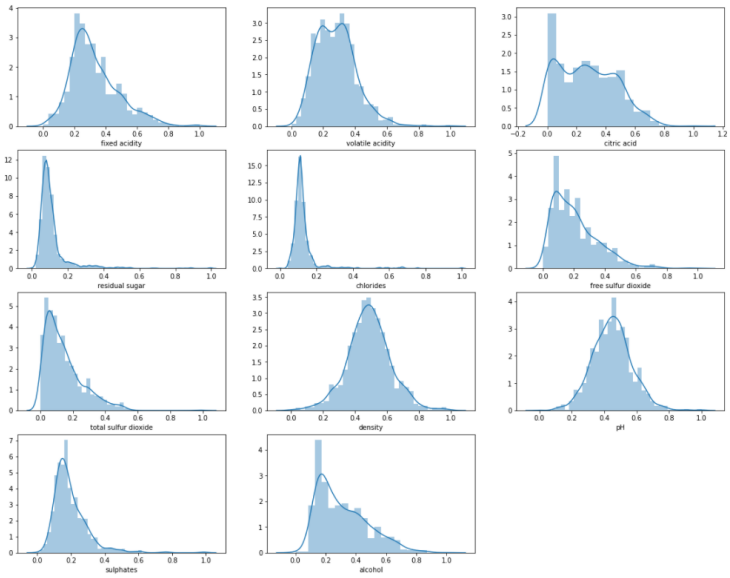


Slika 4: *Box-and-whisker* dijagram

Box-and-whisker dijagram jasno prikazuje visoko prisustvo *outlier* vrednosti za svaki atribut skupa podataka. Prisustvo *outlier* vrednosti može biti rezultat loše čistoće ili čuvanja originalnih podataka, kao i ljudske greške, pa je veličinu ovih podataka teško tumačiti ukoliko čitalac nije domenski ekspert. Iako se u praksi *outlier* vrednosti uklanjaju iz skupa podataka, prisustvo velike količine ovih vrednosti može da pokazuje značaj ovih podataka, pa ih ne treba uklanjati već primeniti druge tehnike rešavanja problema prisustva *outlier* vrednosti.

U nastavku rada je predstavljena primena K-Means klasterovanja i evaluacija rezultata istog, pa je zbog različitih mernih jedinica vrednosti atributa potrebno normalizovati podatke određenom metodom, pri čemu treba napomenuti da se normalizacija u praksi koristi i kao tehnika za rešavanje problema prisustva *outlier* vrednosti.

Apache Spark okruženje i biblioteka *pyspark* pružaju razne mogućnosti skaliranja-normalizacije podataka. U slučaju prisustva *outlier* vrednosti najčešće se koristi *RobustScaler*, jer svoja izračunavanja bazira na interkvartilnom rasponu, što ga čini robustnim za *outlier* vrednosti. Međutim, zbog različitih mernih jedinica vrednosti atributa, izabran je *MinMaxScaler* koji skalira vrednosti na interval [0,1], pri čemu je bitno napomenuti da u tom slučaju se *inliner* vrednosti grupišu u uskom intervalu oko nule, što ga čini osetljivim za *outlier* vrednosti. Pre samog skaliranja, bilo je potrebno vektorizovati podatke korišćenjem *VectorAssembler* klase koju pruža *pyspark*. Vizualizacija nakon primene *MinMaxScaler* skaliranja je prikazana na slici 5.



Slika 5: Pregled distribucija vrednosti nakon skaliranja

Kako je cilj rada klaster analiza nad skupom podataka fizičko--hemijskih osobina, u poslednjem koraku faze pretprocesiranja izdvojen je atribut *'quality'* iz skupa podataka, koji opisuje vrednosti kvalitativne ocene vina.

3 K-Means klasterovanje

Tehnike klasterovanja podataka spadaju u grupu nenadgledanih metoda mašinskog učenja koje dele podatke na "prirodne"grupe. K-Means predstavlja algoritam klasterovanja koji funkcioniše po principu minimizacije euklidske kvadratne distance između instanci i njenih dodeljenih centara klastera. Prednost ovog algoritma se ogleda u mogućnosti izbora broja klastera definisanjem parametra K, pri čemu kvadratna distanca monotono opada sa povećavanjem vrednosti K.[4]

U praksi postoje različite tehnike izbora K parametra, od kojih će tehnika korišćenja *Silhouette* koeficijenta biti prikazana u nastavku.

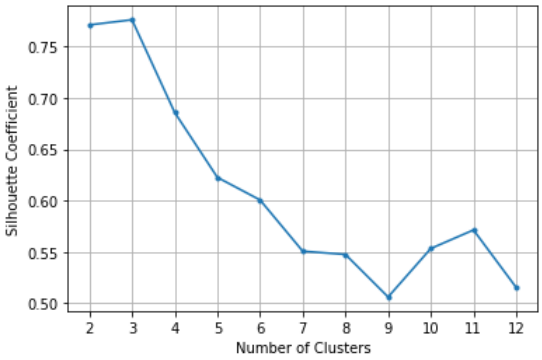
3.1 Izbor parametra K

Klasom *ClusteringEvaluator* koja se nalazi u *pyspark.ml.evaluation* biblioteci, izvršena je evaluacija klasterovanja korišćenjem *Silhouette* koeficijenta. Vrednost ovog koeficijenta se kreće u intervalu [-1,1], pri čemu vrednost bliža +1 označava da su instance bilske instancama u istom klasteru, a udaljene od instanci drugih klastera, tj. da su klasteri adekvatno međusobno razdvojeni, dok je vrednost bliža -1 označava pogrešno dodeljivanje instanci klasterima. Vrednost 0 označava da distanca između klastera nije značajna. [1]

Narednom sekcijom koda je izvršena analiza *Silhouette* koeficijenta:

In [1]:
```
silhouetteList = []
for k in range(2,13):
 kmeans = KMeans().setK(k).setSeed(1) \
                     .setFeaturesCol("features
                         ") \
                     .setPredictionCol("
                         prediction")
 model = kmeans.fit(scaledVectors)
 predictions = model.transform(scaledVectors)
 evaluator = ClusteringEvaluator()
 silhouette = evaluator.evaluate(predictions)
 silhouetteList.append(silhouette)
```

Vizualizacija rezultata prikazana na slici 6 jasno prikazuje da je *Silhouette* koeficijent najviši kada K vrednosti iznosi 3, stoga je ova vrednost i izabrana kao vrednost K.



Slika 6: *Silhouette* koeficijent u odnosu na broj klastera

3.2 Pipeline

Nakon izbora parametra K, kreiran je *pipeline* koji sadrži tri faze: vektorizaciju, skaliranje i primenu K-Means klasterovanja, komandama u nastavku:

In [2]:
```
kmeans = KMeans(k=3, featuresCol="
    scaledFeatures", predictionCol='
    prediction')
pipeline = Pipeline(stages=[assembler, scaler
    , kmeans])
```

Korišćenjem klase *PipelineModel* kreiran je model koji će u nastavku biti korišćen za evaluaciju rezultata:

In [3]:
```
model = pipeline.fit(wineDataNew)
cluster = model.transform(wineDataNew)
```

Nakon kreiranja modela, koristeći poslednju fazu *pipeline*-a, izdvojeni su centroidi klastera:

In [4]:
```
centers = model.stages[2].clusterCenters()
for center in centers:
  print(center)
```

```
Out[4]:  _[0.31584306 0.27625571 0.30380531 0.13869358
              0.13132507 0.35562342
           0.28787121 0.52478786 0.42175923 0.1940897
              0.22117843]
          [0.48067951 0.18906862 0.47867612 0.11573399
             0.1431227  0.14071188
           0.08438797 0.54202206 0.36311684 0.24168684
              0.37189792]
          [0.22756852 0.34761134 0.10825796 0.09235676
             0.11331281 0.1758676
           0.10513699 0.4275612  0.52452551 0.16660147
              0.32264313]
```

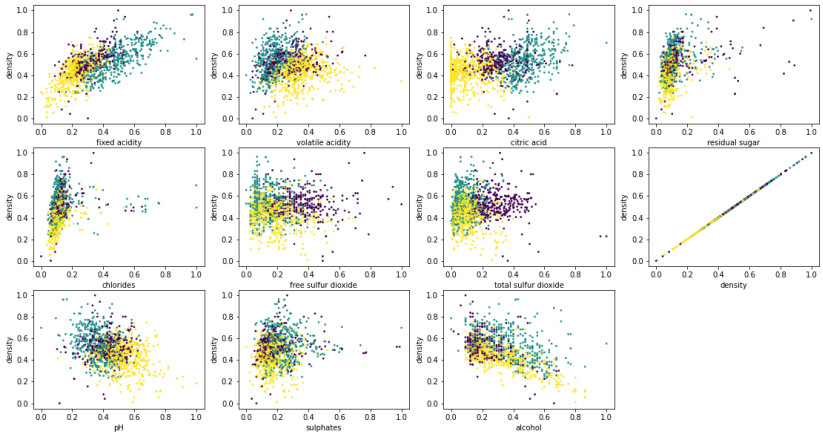
3.3 Vizualizacija

Tipičan primer prikazivanja rezultata klasterovanja predstavlja *Scatter* dijagram. *Scatter* dijagram je tip matematičkog dijagrama koji se koristi da bi prikazao tipične vrednosti dve promenljive u koordinatnom sistemu, kreiran od strane britanskog statističara Francis Galtona 1888. godine, sa ciljem prikazivanja korelacije između dve promenljive. Podaci su prikazani kao kolekcija tačaka u koordinatnom sistemu gde su vrednosti promenljivih prikazane na x i y osi.[2]

Intuitivno je jasno da je prikaz rezultata klasterovanja trivijalan ukoliko je skup podataka dvodimenzionalan ili trodimenzionalan, tj. sastoji se iz dva ili tri atributa, ne računajući rezultate klasterovanja. Postavlja se pitanje na koji način se mogu prikazati klasteri podataka n-dimenzija?

Kao naivno rešenje se nameće smanjivanje dimenzionalnosti grupisanjem atributa u grupe po 2 ili 3 člana, radi lakšeg prikaza na grafikonima. Ovo rešenje može biti pogodno za razmatranje segregacije podataka po klasterima u odnosu na odabrane atribute. Na slici 7 su prikazane dvodimenzionalne projekcije klastera, pri čemu se jedna dimenzija odnosi na vrednost atributa *'density'*, dok se druga dimenzija odnosi na sve ostale atribute.

Sa većine dijagrama se može uočiti da većina instanci sadrži vrednost gustine (*'density'*) iz intervala [0.3, 0.7], nezavisno od količine druge supstance. Sa druge strane, na svim dijagramima se primećuje disperzija podataka kod klastera čije su instance obojene ljubičastom bojoj, pa bi u budućnosti bilo poželjno ispitati da li je to efekat prisustva *outlier* vrednosti, uzimajući u obzir da je izvršeno skaliranje osetljivo na iste.



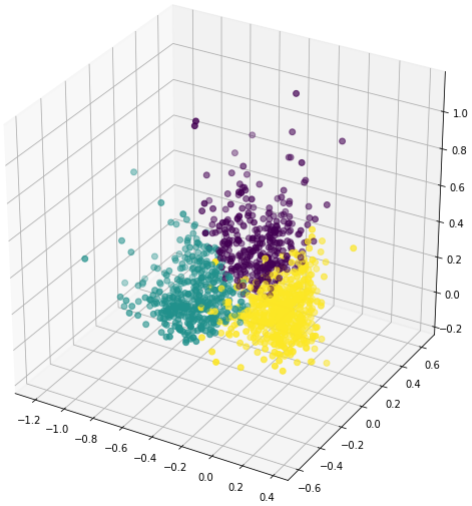
Slika 7: 2D *Scatter* dijagram klaterovanja

U nastavku će biti prikazana alternativna tehnika vizualizacije rezultata klasterovanja, koja je pogodna za trodimenzionalni prikaz i lakšu identifikaciju klastera.

3.4 Vizualizacija pomoću PCA

Analiza glavnih komponenti (eng. *PCA - (Principal Component Analysis)*) predstavlja metodu redukcije dimenzionalnosti skupa podataka. Ova nenadgledana metoda transformacije podataka funkcioniše po principu identifikacije smera varijanse podataka, tj. smera disperzije podataka. Primenom PCA dobijeni su novi atributi, koji se u kontekstu PCA nazivaju glavne komponente, dobijene linearnom kombinacijom prvobitnog skupa podataka. Prva nova komponenta obuhvata najveći deo disperzije skupa podataka, a svaka sledeća obuhvata manji deo disperzije, koji nije pokriven prethodnom komponentom.[4]

Sa ciljem vizualizacije rezultata klasterovanja PCA metodom je redukovana dimenzija skupa sa 11 atributa na 3 glavne komponente, koje predstavljaju dimenzije *scatter* dijagrama na kome se vide instance grupisane u klastere obojene različitom bojom na slici 8. Sa iste slike se može primetiti da su centri klastera međusobno bliži jedan drugom, kao i da je disperzija podataka veća kod klastera čije su instance obojene ljubičastom bojom, nego što je to slučaj kod klastera čije su instance obojene tirkiznom i žutom bojom.



Slika 8: 3D *Scatter* dijagram klaterovanja

4 Čuvanje *model* i *pipeline* objekata

Na samom kraju, *model* i *pipeline* je moguće sačuvati u željenom direktorijumu navodeći odgovarajuće adresne niske karaktera u sledećim komandama:

In [5]:
```
model.save("/content/gdrive/My Drive/Colab
    Notebooks/model")
pipeline.save("/content/gdrive/My Drive/Colab
     Notebooks/pipeline")
```

Model i *pipeline* se mogu učitati izvršavanjem sledećih komandi, takođe navodeći odgovarajuće adresne niske karaktera kao parametre za lokaciju direktorijuma.

In [6]:
```
loadModel = PipelineModel.load("/content/
    drive/My Drive/Colab Notebooks/model")
loadPipeline = Pipeline.load("/content/drive/
    My Drive/Colab Notebooks/model")
```

5 Zaključak

U prethodnim sekcijama je prikazana klaster analiza skupa podataka koji sadrži fizičko-hemijske osobine crvenog vina. Na osnovu *Silhouette* koeficijenta može se zaključiti da je segregacija podataka na klastere na zadovoljavajućem nivou u slučaju kada je za K - broj klastera izabrana vrednost 3. U budućnosti bi bilo poželjno analizirati da li dolazi do promene rezultata klasterovanja pri promeni vrednosti *seed* parametra *KMeans* algoritma.

Iznesen je adekvatan zaključak o odnosu gustine vina (*'density'*) i ostalih fizičko-hemijskih osobina u perspektivi segregacije podataka na klastere. Sačuvani *model* i *pipeline* se mogu iskoristiti i za analizu međusobnog odnosa vrednosti drugih atributa u istoj perspektivi.

Literatura

[1] C.C. Aggarwal. *Data Mining: The Textbook*. Springer International Publishing, 2015.

[2] Krstić Dijana. Istraživačka analiza podataka (EDA) uz upotrebu statističkog softvera R. Master rad, Univerzitet u Novom Sadu, Prirodno-matematički fakultet, Trg Dositeja Obradovića 3, Novi Sad, Srbija, 6 2016.

[3] UCI Machine Learning. Uci red wine quality dataset, 2017. na stranici: https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009/.

[4] I.H. Witten, E. Frank, M.A. Hall, and C.J. Pal. *Data Mining: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2016.